MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-3720-TDL/201

"PADRONIZAÇÕES GRÁFICAS"

Carlos Alberto Felgueiras

Trabalho de Integrado realizado no Instituto de Pesquisas Espaciais, em março de 1985.

INPE
São José dos Campos
1985

À Debora pela força sempre presente.

AGRADECIMENTOS

A todos aqueles que demonstraram interesse pela nova filosofia do Exame Integrado em Computação Aplicada e contribuiram, de alguma forma, para que este fosse concretizado já neste ano de 1985.

Em especial à Comissão para Exame Integrado em Computação Aplicada.

Às Chefias de Departamento e Divisão pela compreensão e pelo tempo de expediente cedido, sem os quais este trabalho não teria sido terminado em tempo hábil.

ABSTRACT

The increasing use of graphic systems, specially those developed in aid of design and manufacture, has caused a quick growing of an amount of different graphic softwares implemented in various manufactures' graphic fittings. The graphic systems standardization process stands in need of harmonizing application softwares, I/O devices, graphic data formats and process of information exchange involved in the graphic computation area. The aim of this work is to give the reader an idea of the present situation of the recent works that are being developed in order to obtain a graphic systems standardization.

SUMÁRIO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS

ACM : "Association for Computing Machinery"
ANSI : "American National Standard Institute"
ASCII : "American Standard Code for Information Interchange"
CAD : "Computer Aided Design"
CAM : "Computer Aided Manufacturing"
DIN : "Deutch Institute fur Normating"
E/S : "Entrada e Saída"
GKS : "Graphical Kernel System"
GSPC : "Graphics Standard Planning Committee"
IEEE : "Institute of Electrical and Electronics Engineers"
IFIP : "International Federation of Information Processing"
IGES : "Initial Graphics Exchange Standards"
ISO : "International Standards Organization"
NAPLPS : "North American Presentation Level Protocol Syntax"
NDC : "Normalized Device Coordinate"
PHIGS : "Programmer's Hierarquical Interative Graphics Standard"
PMIG : "Programmer's Minimal Interface to Graphics"
SIGGRAPH: "Special Interest Group on Graphics"
VDI : "Virtual Device Interface"
VDM : "Virtual Device Metafile"
X3H3 : "Comitê Técnico para Computação Gráfica no ANSI"
WC : "World Coordinates"
WDSS : "Workstation Dependent Segment Storage"
WG2 : "Grupo de Trabalho de Computação Gráfica da ISO"
WG5.2 : "Grupo de Trabalho de Computação Gráfica da IFIP"
WISS : "Workstation Independent Segment Storage"

CAPÍTULO 1

INTRODUÇÃO

A necessidade de padronização de sistemas gráficos começou a ser sentida a partir de 1974. Desde então, muito trabalho foi realizado na tentativa de oferecer aos fabricantes e usuários de sistemas gráficos padrões gráficos compatíveis com os interesses de ambas as partes.

O presente trabalho objetiva fornecer uma visão geral dos principais padrões gráficos, nos seus diferentes níveis de atuação, que já foram aprovados ou estão em fase de adoção ou desenvolvimento nos principais órgãos de padronização, tais como o ANSI, a ISO e o DIN.

No primeiro capítulo é introduzido o conceito de padronização gráfica e delimitados os níveis de atuação das principais interfaces que deveriam compor um sistema gráfico para que o objetivo de padronização deste sistema fosse alcançado.

No segundo capítulo é apresentada uma descrição sucinta dos vários padrões gráficos, adotados ou em fase de adoção, ressaltando, principalmente, os níveis de atuação de cada um, bem como suas principais características funcionais.

No terceiro capítulo apresenta-se um descrição histórica, sucinta, do trabalho que vem sendo desenvolvido na padronização de sistemas gráficos. Relatam-se os principais encontros científicos realizados para apresentar e estudar os padrões gráficos, com seus objetivos e resultados relevantes.

No capítulo seguinte descrevem-se e discutem-se as principais vantagens da criação e adoção de padrões gráficos nos seus diferentes níveis de atuação dentro do sistema gráfico como um todo.

O quinto capítulo fornece uma descrição mais detalhada, ao nível de utilização, do padrão GKS (Graphical Kernel System), uma interface ao nível de aplicação, que está se tornando um padrão mundial devido, principalmente, a adoção de alguns conceitos básicos que o diferem do seu pseudo padrão *raíz*, o CORE, e o torna bastante potente ao nível de aplicação.

No sexto e último capítulo apresenta-se uma visão geral do estágio atual de desenvolvimento dos diversos padrões gráficos existentes, bem como as tendências futuras nesta área. São inseridas, ainda, algumas conclusões importantes em relação ao processo de padronização em geral.

1.1 - INTERFACES DE SISTEMAS GRÁFICOS

A idéia da criação de padrões gráficos resume-se no desenvolvimento de interfaces de "software" gráfico que sejam capazes de estabelecer uma relação usuário/sistema, tal que, o usuário possa visualizar todos sistemas gráficos, independente do "hardware" e "software" básicos que os compõem, como se tivessem estruturas básicas idênticas. Mais informalmente isto significa que se um usuário desenvolve "software" em um sistema gráfico fabricado pela indústria XSG(X Sistemas Gráficos) ele poderá utilizar seu "software" já desenvolvido e/ou continuar desenvolvendo novos programas, em outro sistema fabricado pela indústria YSG (Y Sistemas Gráficos), sem que ele sofra qualquer tipo de *trauma* para adaptar-se ao novo equipamento. Na prática, alguma modificação sempre existe, mas a padronização tende a minimizar este efeito de forma que o "software" se torne bastante portável.

A partir dessa idéia pode-se começar a delimitar os diversos níveis de atuação de cada interface dentro de um sistema gráfico global. A Figura 1 mostra um esquema dos limites de atuação de cada interface.

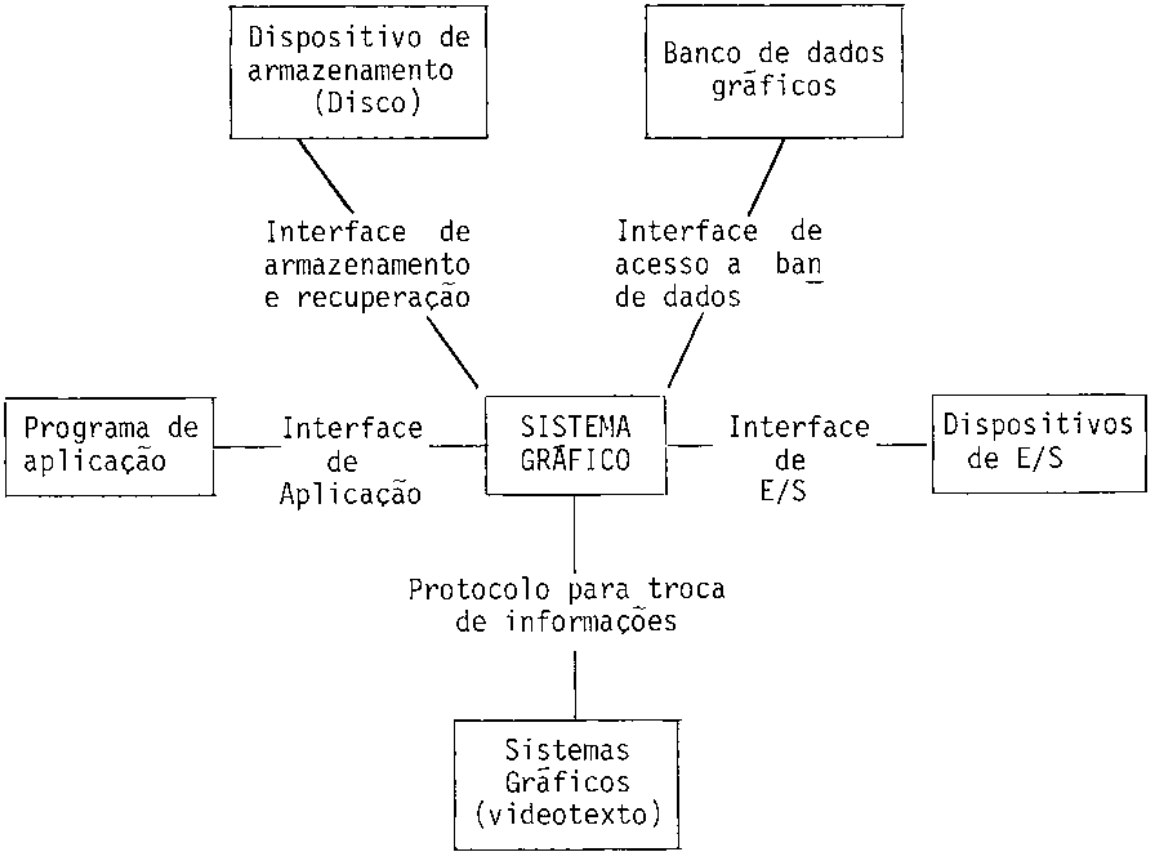

Fig. 1 - Níveis de atuação de um sistema gráfico.

A primeira idéia é que se deve ter uma interface ao nível de aplicação, a qual deve fornecer ao usuário um conjunto de rotinas gráficas básicas padrões que serão utilizadas no desenvolvimento de "software" aplicativo.

Um outro conjunto de rotinas básicas deve possibilitar ao usuário acessar qualquer tipo de dispositivo de entrada/saída de uma forma padrão única, sem que exista preocupação com as características particulares de cada dispositivo.

Deve existir, ainda, uma interface ao nível de banco de dados, que possibilite o acesso a banco de dados gráficos, de uma forma padrão, através de sistemas gráficos diversos. Neste nível inclui-se,

também, a definição do formato padrão de representação de dados e figuras gráficas.

Para aplicações que necessitam de troca de informações através de canais de comunicação diversos, deve-se criar um protocolo padrão de comunicação para troca ou envio de informações gráficas e textuais entre computadores e periféricos. Este nível é particularmente importante nas aplicações de videotexto.

Por último, deve-se criar uma interface que possibilite o armazenamento e recuperação rápidos de informações gráficas, em disco, de tal forma que não exista dependência do tipo ou estrutura de disco utilizada e que esse armazenamento e recuperação sejam otimizados.

A Figura 1 apresenta um esquema dos níveis de atuação de um sistema gráfico e as respectivas interfaces de ligações com esses níveis.

CAPÍTULO 2

OS PADRÕES GRÁFICOS

2.1 - INTRODUÇÃO

Atualmente, existem pelo menos 5 padrões funcionando nos vários níveis de um sistema gráfico como um todo, que são oficiais ou estão em desenvolvimento nas principais organizações de padronizações. Esses padrões são:

1) GKS e CORE - "Graphical Kernel System" e "GSPC CORE System";
2) VDI - "Virtual Device Interface";
3) IGES - "Inicial Graphics Exchange System";
4) VDM - "Virtual Device Metafile";
5) NAPLPS - "North American Presentation Level Protocol Syntax".

O objetivo deste capítulo é dar ao leitor uma visão geral de cada um desses padrões, sem a preocupação com seus detalhes estruturais e de utilização deles. A idéia é apresentar uma definição de cada padrão, bem como suas características funcionais principais. Faz-se ainda uma abordagem dos níveis de atuação de cada padrão dentro do sistema gráfico como um todo, pois ela é de primordial importância para o entendimento de cada uma dessas interfaces.

2.2 - OS SISTEMAS GKS e CORE

O Sistema GKS e o sistema GSPC (Graphics Standard Planning Committee") CORE, ambos, atuam ao *nível de aplicação*. São conjuntos de rotinas gráficas básicas que auxiliam o programador no desenvolvimento de seus aplicativos e permitem a portabilidade de código fonte.

Conceitualmente, as abordagens dos sistemas gráficos CORE e GKS não apresentam divergências básicas. Ambos fundamentam-se nos 4 temas metodológicos emanados do primeiro encontro, com o objetivo de pa

dronizar os Sistemas Gráficos, o "Seillac I", principalmente no primeiro tema, o da transportabilidade de "softwares" aplicativos através da padronização. Esta *transportabilidade* é a capacidade de transportar uma aplicação gráfica de uma instalação para a outra com um mínimo de mudanças no programa. Outro benefício potencial dessa padronização é a transportabilidade de programador, isto é, a capacidade de um programador de aplicação mudar de uma instalação para outra com um mínimo de treinamento adicional.

Cronologicamente, o CORE foi a primeira proposta de pacote gráfico publicada e colocada em uso. Por isto ele é a base sobre a qual se desenvolveu grande parte da computação gráfica e, em particular o GKS.

Porém, existem duas diferenças básicas entre os dois sistemas que distanciaram os seus destinos:

- A *dimensão*, que no CORE é 2-D ou 3-D e no GKS é 2-D apenas.
- A *inclusão do conceito de Estação de Trabalho* ("Workstation")na estrutura do sistema GKS.

Atualmente existe um grande esforço por parte da ISO no sentido de definir uma versão 3-D para o GKS. Os adeptos deste padrão garantem que isto é possível, uma vez que a raiz do GKS é também 3-D. O conceito de Estação de Trabalho será amplamente discutido no Capítulo 5, que trata especificamente do padrão GKS.

Uma diferença mais relacionada com o aspecto formal dos dois documentos é que a especificação funcional do GKS é mais rica e melhor estruturada devido ao *conceito de estação de trabalho, além de apresentar padrão "binding" com linguagens de programação.*

Neste nível, de aplicação, é que tem ocorrido as maiores polêmicas, já que se trata do núcleo gráfico sobre o qual se assenta toda a parte do "software" gráfico visível ao programador de aplicação.

Além do CORE e do GKS, dois outros esforços recentes de padronização merecem ser citados:

O PHIGS - "Programmer's Hierarchical Interactive Graphics Standard" é uma espécie de extensão do CORE, desenvolvido no ANSI, destinada a suportar dados gráficos em estruturas hierárquicas de três dimensões, usadas nas construções de modelagem para sistemas de CAD. No âmbito do ANSI, espera-se um anteprojeto da proposta do PHIGS em 1985, enquanto na ISO ainda não é considerado um item de trabalho. O ponto mais obscuro da proposta é a compatibilidade com o sistema gráfico padrão GKS.

O PMIGS - "Programmer's Minimal Interface to Graphics" é o resultado do trabalho de um subcomitê constituído pelo ANSI para incluir um nível de saída gráfica mínima no padrão GKS, para permitir a utilização do sistema de uma forma mais simples nos casos de aplicações que requerem recursos gráficos mínimos. O PMIGS está incorporado ao GKS na versão do ANSI; aguarda-se a sua aprovação como padrão internacional juntamente com a versão da ISO.

2.3 - O PADRÃO VDI

O VDI ("Virtual Device Interface") é um *conjunto de funções que fornecem uma interface uniforme entre aplicações gráficas de alto nível e dispositivos de E/S*. Isto significa que o VDI possibilita que os "softwares" de aplicação se tornem *independentes dos dispositivos de E/S* que compõem o sistema gráfico. Desta forma, os programas de aplicação podem ignorar as limitações dos dispositivos físicos de entrada e saída, e dados gráficos podem ser mostrados com a mais alta qualidade disponível, em cada dispositivo. Por outro lado, o VDI possibilita que o usuário misture e "case" periféricos de diferentes fornecedores.

De todos os padrões gráficos, o VDI foi o que causou mais impacto no setor de mercado. A implementação do VDI é capaz de alterar a relação entre os produtores de circuitos integrados, os vendedores de periféricos, os projetistas de "software" e os fabricantes de compu

tadores. Atualmente estes quatro grupos estão ligados por uma relação que dificulta inovações. Para exemplificar esta relação, suponha-se que um produtor de circuitos integrados consiga desenvolver uma nova arquitetura mais rápida; neste caso os projetistas de "software" teriam que reescrever todos os "softwares" gráficos de alto nível já existentes; os fabricantes de periféricos e computadores teriam que preparar uma nova linha de produtos para o qual provavelmente não deve ainda existir "software" desenvolvido.

Com o VDI, entretanto, uma nova arquitetura não seria tão ameaçadora. Os projetistas de "software" desenvolveriam "software"para o VDI ao invés de escrevê-los para cada tipo de periféricos; os fabricantes de periféricos teriam assegurados que novos dispositivos funcionariam tanto com os pacotes de "software" já existentes, quanto com "softwares" em desenvolvimento, mesmo que em instalações de diferentes fabricantes; e os fabricantes de computadores teriam maior liberdade para inovações porque o "software" e os periféricos já existentes no mercado, com certeza, seriam compatíveis com seu computador.

2.4 - O PADRÃO IGES

No nível da estrutura de dados da aplicação, o IGES - "Initial Graphics Exchange Standards" - é um padrão ANSI herdado da indústria aeronáutica americana, que *permite a comunicação de dados entre o programa de aplicação de CAD/CAM e o banco de dados gráfico.*

Trata-se de uma formatação de dados que definem um produto, isto é, a descrição das propriedades geométricas dos objetos resultantes de um processo de CAD/CAM.

Um sistema de CAD, para poder compartilhar dados armazenados no padrão IGES, deve possuir um *pré-processador* para traduzir o formato padrão e um *pós-processador* para formatar seus dados de saída segundo esse padrão.

O ANSI já tem um documento final adotado, enquanto, internacionalmente, o DIN estuda para a ISO uma forma de adequar o IGES ao padrão GKS.

Vários fornecedores de grandes sistemas ou já estão oferecendo, ou brevemente oferecerão, o IGES Versão 1.0 para aplicações CAD/CAM mecânicas. Versões estendidas do IGES, para aplicações elétricas e novas aplicações mecânicas, estão atualmente em desenvolvimento.

Em março de 1983, cinco fornecedores de sistemas gráficos CAD/CAM, incluindo a Computer Vision, a Control Data, a GERBER e a IBM, conjuntamente com a BOEING e a "National Bureau of Standards",juntaram-se para demonstrar como arquivos de dados gráficos poderiam ser transferidos para diferentes sistemas sistemas CAD/CAM usando o formato de dados IGES. A demonstração aconteceu na Conferência e Exposição AUTOFACT 4 na Philadelphia.

A demonstração iniciou com um arquivo de dados desenvolvido pela "IGES Test, Evaluation and Suport Committee" e gerado na Boeing. Os dados descreviam a geometria de uma parte mecânica e incluía toda a faixa de dimensionamento necessário em desenhos de engenharia. Os dados foram lidos por cada um dos fabricantes, em seus estandes, armazenados num sistema CAD/CAM, mostrados em vídeo e então armazenados novamente em uma nova fita IGES para ser transportada para um novo sistema.

Resumindo, o IGES fornece um passo intermediário na transferência de informações de projeto entre 2 sistemas CAD/CAM que podem até ser incompatíveis. Tudo o que é necessário é que o fabricante de cada sistema forneça um pré e um pós-processador para transladar arquivos de dados de um sistema para o formato IGES e vice-versa.

Em termos de conteúdo é bom que fique claro que os arquivos IGES podem conter não somente dados de desenhos, mas também toda a estrutura geométrica 3-D deles.

O trabalho de desenvolvimento de novas versões do IGES procura manter a compatibilidade das antigas versões, ou seja, novas versões do IGES deverão ser capazes de trabalhar com arquivos criados com as versões anteriores do IGES.

2.5 - O PADRÃO VDM

A função do "software" padrão VDM ("Virtual Device Metafile") é *especificar formatos de arquivos para armazenar e transmitir figuras*. O VDM usa coordenadas de dispositivos normalizadas e primitivas para realizar essas tarefas.

A tendência atual, em relação à criação de uma interface padrão do tipo VDM, é adaptar a estrutura do "metafile" existente no padrão GKS como ponto de partida para padronização do VDM. Nas seções seguintes serão feitas algumas considerações sobre os "metafiles" implementados com o sistema GKS. Para entender melhor os conceitos de segmento e estação de trabalho que serão citados nas considerações a seguir, consulte o Capítulo 5.

Além de ser independente do dispositivo, pode-se reprocessar e modificar a informação gráfica em um "Metafile" sem a necessidade da restauração da imagem armazenada.

Comparando o VDM com segmentos, quando uma sessão do GKS é fechada, todas as informações armazenadas nos segmentos dessa sessão são perdidas. Portanto, os segmentos fornecem um mecanismo para armazenamento de informações gráficas apenas dentro de uma invocação simples do GKS. O "Metafile" do GKS fornece um mecanismo para armazenamento de informações gráficas por longo período. Isto significa que um "metafile" pode ser *criado* por uma invocação do GKS e reusado por subsequentes invocações que podem ser: do mesmo programa; de um programa diferente, mas do mesmo usuário; de um programa diferente de um usuário diferente; ou mesmo de um programa de um local diferente.

O GKS contém uma especificação de como os "metafiles" são lidos e escritos sugerindo, ainda, um método para armazenamento do "metafile". Os "metafiles" são tratados, pelo GKS, como categorias especiais de estações de trabalho. Eles são abertos e fechados pelas mesmas funções como as outras estações de trabalho. Mais de um "metafile" pode estar ativo simultaneamente para possibilitar que formatos diferentes de "metafiles" possam ser criados simultaneamente.Similarmente, mais de um "metafile" de entrada pode ser aberto simultaneamente para leitura. A instalação define se os tipos particulares de estações pertencem a categorias de "metafiles" de entrada ou "metafiles" de saída.

Vários estudos têm sido realizados no âmbito da ISO para compatibilizar a intersecção existente entre o IGES e o VDM, levando em conta a padronização do GKS.

2.6 - O PADRÃO NAPLPS

O "North American Presentation Level Protocol Syntax" - NAPLPS - aprovado pelas maiores organizações de padronização em 1983, *fornece um padrão de comunicação para troca de informações gráficas e textuais entre computadores*. Ele é particularmente bem sucedido para codificação de dados gráficos numa forma altamente portável e independente de resolução.

O "ANSI Standards X3.110-1983" descreve formatos, regras e procedimentos para codificação de texto alfanumérico e informação pictorial que pode então ser transmitida através do meio mais apropriado para uma aplicação (telefone, modem, rede de computadores, mecanismos terminais, rádio FM de banda lateral ou transmissão de TV). Essa independência do meio de transmissão é uma das características mais importantes do NAPLPS. Ele possibilitará migrações mais simples para novas tecnologias quando estas forem introduzidas. Portanto, não existe obsolência embutida no padrão.

O protocolo de comunicação ASCII é o que existe de mais restrito a um padrão universal para o envio de dados via linha telefônica. O protocolo ASCII é adequado para muitas aplicações que se utilizam apenas de texto, mas ele é inaceitável para computadores gráficos porque seu formato incorpora somente códigos alfanuméricos.

O NAPLPS, entretanto, que foi voltado na sua forma atual pelo ANSI e pelo "Canadian Standards Association" em novembro de 1983, engloba o padrão ASCII e oferece um modo rápido e efetivo de tratar com comunicações baseadas em gráficos.

Em termos simplificados, o NAPLPS é um *superconjunto do ASCII (ANSI X3.4 - 1977)*; ele aumenta a capacidade do padrão ASCII através do uso de extensões. Isto permite que sejam incorporadas capacidades mais complexas de display na mesma imagem visual, de forma que palavras e figuras possam aparecer na mesma tela. As extensões são acompanhadas, primariamente, de *opcode* que informam ao terminal: qual dos conjuntos de caracteres prédefinidos ou variáveis (dinamicamente redefiníveis) usar; qual a forma gráfica de um repertório de primitivas geométricas simples (ponto, linha, arco, polígono, etc.) desenhar; e qual o modo do meio a ser usado para pintar os gráficos e textos na tela (cor, cintilação, largura de traço, textura, etc.), e onde desenhá-los.

A Figura 2 mostra um esquema do relacionamento de um sistema gráfico com os padrões discutidos nesta seção.

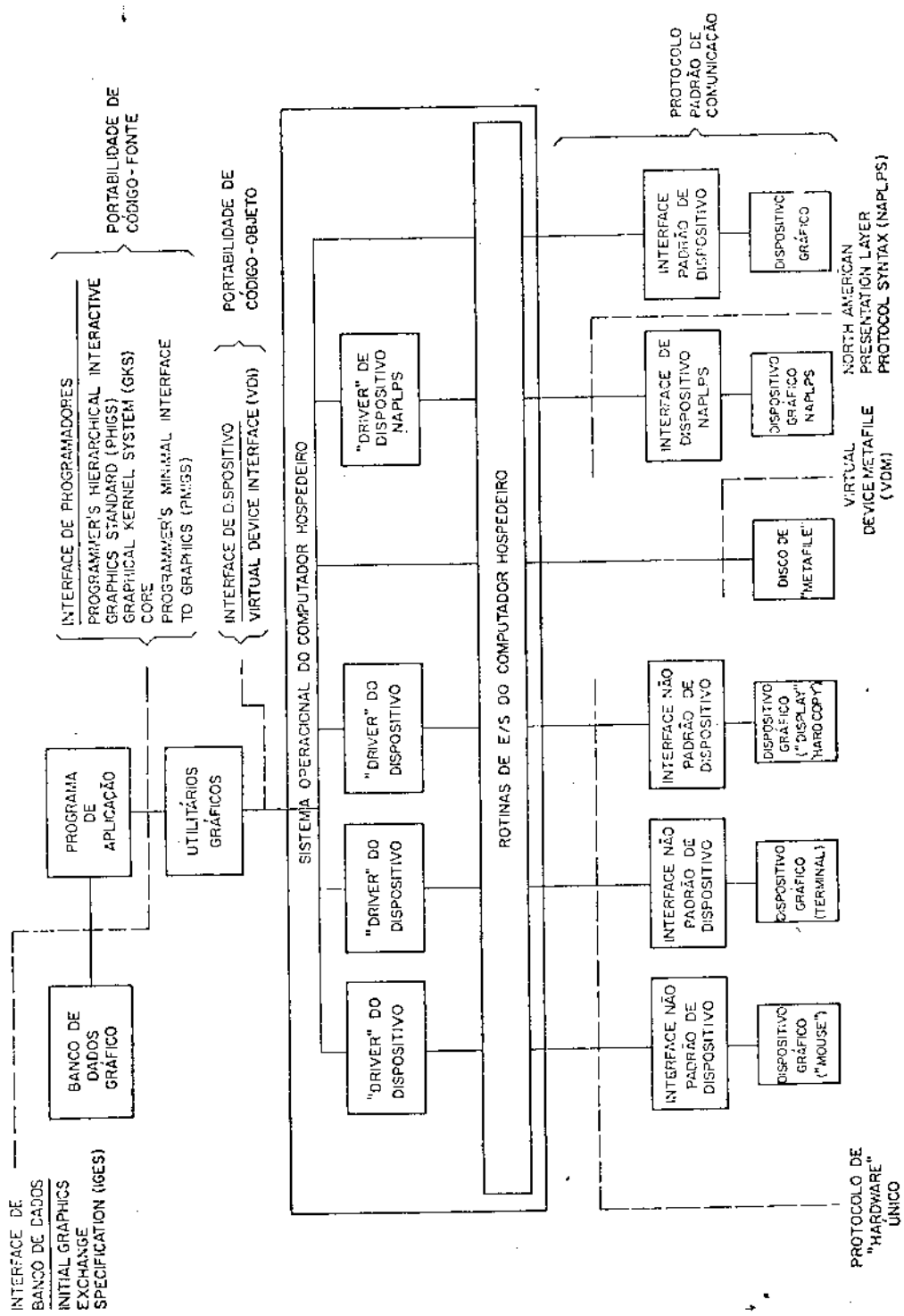

Fig. 2 - As relações entre os padrões gráficos.

CAPÍTULO 3

HISTÓRICO DAS PADRONIZAÇÕES EM COMPUTAÇÃO GRÁFICA

De acordo com Hatfield e Herzog numa breve história de "A Busca para Padrões", *o período de 1963 a 1974 foi caracterizado por muitos desenvolvimentos e projetos isolados no campo da computação gráfica*. Em 1974, Newman e Sproull escreveram: *Virtualmente cada vez que um terminal de display gráfico é ligado a um computador em uma configuração própria, um novo sistema de "software" gráfico deve ser escrito para suportá-lo.*

Com essa consciência, *a necessidade de padronização* tornou-se evidente e muitos comitês vêm trabalhando com a finalidade de compatibilizar sistemas gráficos, desde o primeiro "ACM Siggraph Workshop on Machine Independente Graphics" ocorrido em abril de 1974 na "National Bureau of Standards".

Em maio de 1976 realizou-se a "International Workshop Graphics Standards Methodology", o "Seillac I Workshop" em Castelo de Seillac no Vale do Loire, na França, com o objetivo de desenvolver uma metodologia para obter portabilidade de programa, bem como uma especificação funcional para o sistema gráfico CORE. Este encontro possibilitou uma revisão dos conceitos de computação gráfica que foram mal entendidos no encontro anterior. Foram, ainda, estudados tópicos diversos desde as razões para padronização até o corpo e os requisitos de um padrão gráfico. Participaram deste encontro especialistas em computação gráfica dos Estados Unidos e da Alemanha Ocidental. Alguns desses americanos eram membros do Comitê de Planejamento de Padrões Gráficos ("GSPC - Graphics Standards Planning Committee"), formado 2 anos antes sob o patrocínio do "ACM Special Interest Group on Computer Graphics (SIGGRAPH)", com o objetivo de especificar um sistema gráfico central. Enquanto se iniciava a definição do sistema GSPC CORE pelo grupo americano, o Instituto de Padronizações da Alemanha Ocidental, DIN, criou um grupo com este mesmo objetivo. O grupo chefiado por José Encarnação pro

duziu diversas versões do seu sistema gráfico central, o "Graphical Kernel System" - GKS. Uma das maiores diferenças entre as duas propostas era que o GKS, um sistema 2-D, era significativamente mais pobre em funcionalidade do que o sistema GSPC CORE.

Após grande quantidade de trabalho realizada pelo grupo GSPC, foi publicado em 1977 um primeiro projeto público ("public draft") de um sistema gráfico central. Esta proposta endereçava apenas terminais gráficos vetoriais. No encontro de Toronto, em agosto de 1977, o principal item em discussão foi a recente publicação sobre o GSPC CORE. Em 1978, uma edição inteira da "ACM Computer Surveys" foi devotada à descrição do GSPC CORE e exemplificação de seu uso.

O encontro de fevereiro de 1979, em Amsterdã, objetivou a comparação das duas propostas, o GKS versão 4 e o GSPC Core 77, para identificar as maiores diferenças entre eles. A diferença fundamental era que o GKS não incorporava o conceito de Posição Corrente ("Current Position"). Outra diferença marcante era o conceito adotado pelo GKS de Estação de Trabalho que, sobre o controle do programa de aplicação, possibilitaria que as características particulares de um "display" de saída fossem usadas de modo a aproveitar o máximo de vantagens deste dispositivo. Neste encontro, recomendou-se um número de mudanças em ambos os sistemas, de modo a compatibilizar as duas propostas.

Ainda em 1979 o SIGGRAPHS publica uma nova versão do CORE que endereçava terminais "raster", "metafiles" e gráficos distribuídos. Foi então recomendado que o trabalho do GSPC deveria passar para o corpo formal de padronização americano, o ANSI. O grupo de trabalhos gráficos, X3H3, do ANSI teve seu primeiro encontro em setembro de 1979. Neste, o grupo adotou como ponto de partida para seu trabalho a última versão do GSPC CORE.

No encontro seguinte do grupo de trabalho da ISO em Budapest, em outubro de 1979, a versão 5.1 do GKS incorporava várias recomendações do último encontro. Além disto, as facilidades de entrada fo

ram aumentadas e a capacidade de uso de múltiplos dispositivos de saída simultaneamente foram introduzidas. Nesta oportunidade o subcomitê ISO/TC97/SC5/WG2 resolveu iniciar um processo para adoção do GKS como um padrão internacional.

Em 1981 o subcomitê X3H3 começa trabalhar na interface de dispositivo Virtual, VDI ("Virtual Device Interface"), com o suporte de 15 fabricantes de dispositivos gráficos, representando interesses de "software" e "hardware".

No ano de 1982 o "Graphical Kernel System", GKS, é adotado pela ISO como um *"Draft International Standard"*, o penúltimo passo para a padronização, e o ANSI inicia um processo para a adoção do GKS como padrão.

Em 1983 o GKS e o VDM são liberados para "public review". No mês de novembro deste mesmo ano, a "North American Presentation Level Protocol Syntax", NAPLPS, é aprovada pelas organizações de padronização ANSI e "Canadian Standards Association" como um padrão de comunicação para troca de informações gráficas e textuais entre computadores e dispositivos de E/S.

Em 1984, um quadro de controle é formado pela ISO para responder questões sobre a implementação atual do GKS.

A Tabela 1 mostra a situação até maio de 1984, dos padrões gráficos chaves que estão em estudo no ANSI.

Como se pode notar, a padronização é um processo longo e trabalhoso. Por exemplo, a adoção do GKS como um padrão levou aproximadamente 7 anos desde os primeiros trabalhos neste sentido. Contudo, este trabalho torna-se necessário à medida que os sistemas vão se diversificando e suas consequências são extremamente vantajosas, tanto para os fabricantes como para os usuários que estão envolvidos com sistemas gráficos.

TABELA 1

ESTADO ATUAL (MAIO/84) DOS TRABALHOS NO ANSI

SITUAÇÃO DOS PADRÕES CHAVES NO ANSI			
PADRÃO (Mn)	PROJETO PROPOSTO PARA O COMITÊ X3H3	REVISÃO PÚBLICA	PROPOSTA FINAL PARA O COMITÊ X3H3
GKS 2-D	Outubro 1983	Março a junho 1984	Depende das observações dos revisores
VDM	Outubro 1983	Fevereiro a março 1984	Depende das observações dos revisores
VDI	Trabalho em progresso	Verão ou Outono 1985	-
GKS 3-D	Depende do trabalho da ISO (Outono/1984)	-	-
IGES	Documento final já adotado, mas em revisão	-	-
NAPLPS	Documento final já adotado	-	-

CAPÍTULO 4

VANTAGENS DA PADRONIZAÇÃO EM SISTEMAS GRÁFICOS

A utilização de padrões em sistemas gráficos produz numerosos benefícios. Os principais são:

- Portabilidade de "software": A implementação de padrões gráficos permite o transporte de "software" de aplicação de uma instalação gráfica para outra, sem que sejam necessárias mudanças bruscas nesse "software".

- Facilidade de desenvolvimento de programas de aplicação: No desenvolvimento de padrões existe uma preocupação notória na adoção de uma metodologia de programação, ou seja, as funções das diversas interfaces padrões são definidas de modo a facilitar o trabalho de desenvolvimento de "software" pelo usuário de aplicação. Além disto, o programador se torna livre para concentrar seus esforços no "software" de aplicação sem se preocupar com detalhes de cada dispositivo gráfico e com "software" de baixo nível.

- Portabilidade de programadores: Um dos principais problemas de operação e utilização de sistemas gráficos é a necessidade de treinamento de programadores para entenderem a complexidade de cada nova instalação gráfica que eles se defrontam. A padronização oferece uma oportunidade de reduzir esse problema de treinamento e alcançar, assim, a portabilidade de programadores.

- Portabilidade de representação de dados gráficos: O conteúdo de figuras e/ou dados gráficos podem ser representados como itens de um banco de dados em um determinado formato padrão. Desta forma essas figuras e dados podem ser transportados de uma instalação para outra, podendo ser mostrados por computadores diferentes e ainda resultarem nas mesmas figuras (ou dados).

- Uso de equipamentos gráficos de diferentes fornecedores: O uso de interfaces comuns em todos os sistemas gráficos permite que o usuário selecione produtos gráficos de diversos vendedores sem a necessidade, que existia anteriormente, de ficar amarrado ao fornecedor do "núcleo do sistema".

- Aumento da comerciabilidade de pacotes de "software": Os consumidores de pacotes de "software" de aplicação têm assegurado que poderão mudar suas instalações gráficas sem que seu produto de "software" se torne obsoleto. Consequentemente o comprador de "software" tem garantia de uma vida útil bastante longa para sua aquisição. Portanto, o investimento em "software" cresce.

CAPÍTULO 5

O PADRÃO GKS

5.1 - DEFINIÇÃO

O GKS, iniciais de "Graphical Kernel System", é uma interface padrão ao nível de aplicação, desenvolvida na Alemanha Ocidental a partir de 1976 por membros da Organização de Padronizações da Alemanha Ocidental (DIN). Suas raízes estão contidas nas primeiras propostas de padrões gráficos, incluindo-se aí, principalmente, a proposta do "ANSI - Special Interest Group on Computer Graphics" (SIGGRAPH), o CORE. Durante seu desenvolvimento o GKS tem sido influenciado por importantes organizações de padronização como o ANSI, a ISO e o DIN.

O padrão GKS possibilita a portabilidade de código fonte de "software" de aplicação, ou seja, ele possibilita o transporte de programas de aplicação entre diferentes sistemas através da definição de uma interface consistente. Isto é obtido com a criação de uma "linguagem" de ligação de alto nível, a qual define a sintaxe exata de chamada de cada função gráfica que compõe a interface. A definição dessa sintaxe de chamada assegura que os programadores gráficos serão capazes de trabalhar em implementações de aplicações diferentes, sem precisarem caminhar através de uma linha de aprendizado difícil.

Informalmente, o objetivo do GKS é produzir e manipular as figuras de maneira independente do computador ou dispositivo gráfico a ser usado. Essas figuras variam de uma simples linha gráfica (que servem para ilustrar resultados experimentais, por exemplo), a desenhos de engenharia para "lay-outs" de circuitos integrados (usando cores para diferenciar as diferentes ligações), até imagens que representam dados médicos ("scanners" de tomógrafos computadorizados) em escala de cinza ou em cores.

Para resumir, o GKS define uma interface entre um programa de aplicação e um modelo gráfico, protegendo a aplicação das diferenças entre os vários computadores e dispositivos gráficos. Um programador de aplicação aprende essa linguagem gráfica simples, que gerencia os detalhes de entrada e saída de imagens gráficas e desenvolve todo seu "software" baseado nessa linguagem, tornado-se livre de preocupação com o "software" de baixo nível.

5.2 - CARACTERÍSTICAS FUNCIONAIS DO GKS

5.2.1 - PRIMITIVAS DE SAÍDA

O GKS fornece seis primitivas básicas de saída, mostradas na Figura 3, que são:

- "Polyline": desenha uma sequência de segmentos de linhas conectados.
- "Polymarker": marca uma sequência de pontos da tela com um símbolo prèestabelecido.
- "Fill Area": preenche uma área (polígono) especificada.
- "Text": desenha uma cadeia de caracteres.
- "Cell Array": permite especificar uma matriz de cores.
- "Generalized Drawing": permite o desenho de formas geométricas comuns especificadas de maneira dependente da implementação. Pode usar capacidade de "hardware", tais como: geradores de círculos, de elipses, etc.

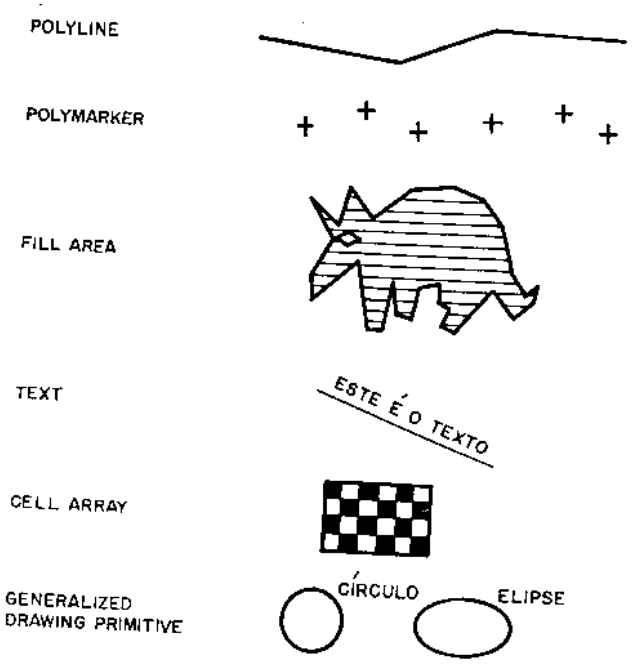

Fig. 3 - Primitivas de saída no GKS.

5.2.2 - ATRIBUTOS DA PRIMITIVAS

A cada primitiva de saída está associada uma série de atributos da primitiva que podem ser modificados de acordo com a necessidade da aplicação.

Os principais atributos de cada primitiva de saída são:

- Atributos da primitiva *"Polyline"*:
  - tipo de linha (contínua ou pontilhada),
  - largura da linha,
  - cor da linha.

- Atributos da primitiva *"Polymarker"*:
  - tipo de marca (*, +, #, ., o, etc.),
  - tamanho da marca,
  - cor da marca.

- Atributos da primitiva *"Fill Area"*:
  - tipo de preenchimento (em branco, totalmente pintado, hachurado, etc.),
  - cor de preenchimento.

- Atributos da primitiva *"Text"*:
  - fator de expansão,
  - espaçamento entre letras,
  - cor,
  - altura,
  - orientação,
  - direção,
  - alinhamento.

A primitiva de saída "Text", no GKS, possibilita facilidades de texto em três precisões diferentes: *Precisão "String"*, *Precisão "Character"* e *Precisão "Stroke"*. As precisões "String" e "Character" utilizam geradores de caractere por "hardware".

A *precisão "String"* não necessita utilizar todos os atributos da primitiva "text". É aplicada em textos utilizados para processos interativos, onde se requer a produção rápida de textos.

A *precisão "Character"* é mais elaborada que a precisão "String". Nesta a posição dos caracteres são precisas, mas a orientação dos caracteres individualmente é fixada para cima e para a direita.

A *precisão "Stroke"* é a mais alta precisão com a qual se pode trabalhar na primitiva "Text". Ela é usada para obter uma saída de alta qualidade. Ela suporta a faixa completa de atributos da primitiva "Text".

O GKS especifica atributos de primitivas de 3 maneiras diferentes: *Individual ("Individual")*, *Agrupada ("Bundled")* e *Misturada ("Mixed")*.

No *modo Individual* os atributos das primitivas de saída, relacionados acima, são definidos individualmente no programa de aplicação. Por exemplo, os atributos individuais de uma linha incluem cor, tipo de linha e largura da linha. Estes atributos são modificados individualmente por funções específicas de cada atributo.

No *modo Agrupado* define-se um pacote de atributos válidos para aquela primitiva de saída. Este pacote ou grupo de atributos contêm uma definição específica da primitiva. No caso da primitiva "poly line", por exemplo, o pacote deve conter uma definição específica da cor, tipo e largura da linha. Após a definição do grupo, pode-se referenciá-lo por sua identificação de grupo. As implementações GKS fornecem grupos prédefinidos para cada primitiva de saída, porém o usuário pode definir seus próprios grupos.

No *modo Misturado*, também conhecido como modo Chaveado,o usuário tem a possibilidade de utilizar os dois modos anteriores, mas não ambos ao mesmo tempo. O programa chaveia entre o modo individual e o modo agrupado de acordo com as necessidades.

5.2.3 - DISPOSITIVOS LÓGICOS DE ENTRADA GRÁFICA

No GKS, os dados que servem de entrada para programas de aplicação, através do operador, são divididos em seis diferentes tipos. Assim, seis classes diferentes de dispositivos lógicos de entrada são definidos correspondendo a esses tipos de dados. Da mesma forma, quando se descreveu as primitivas de saída e seus atributos, não se preciso preocupar com os dispositivos físicos de saída e suas características particulares, aqui também não é necessária a discussão das características físicas detalhadas de cada dispositivo de entrada, mas concentrar-se-á nos dispositivos lógicos de entrada. O mapeamento dos dispositivos físicos de entrada para os dispositivos lógicos de entrada será entendido melhor na Seção 5.2.4 que discute o conceito de estação de trabalho.

As seis classes de dispositivos lógicos de entrada no GKS são:

1) "Locator": dispositivo que permite apontar para uma posição na tela e receber a coordenada (x, y) dessa posição. Os principais dispositivos físicos pertencentes à classe "locator" são: "light pen", "joystic", "mouse", "crosshair" e "thumbwheel".

2) "Valuator": este dispositivo lógico de entrada fornece um valor real para o programa de aplicação. Os dispositivos físicos típicos desta classe são: potenciômetros e teclado do terminal de vídeo.

3) "Stroke": o dispositivo lógico de entrada "stroke" fornece uma sequência de pontos para o programa de aplicação como se fosse formado de múltiplos "locators". O dispositivo físico típico desta classe é a mesa digitalizadora.

4) "Choice": este dispositivo lógico de entrada fornece ao programa de aplicação um valor inteiro, especificando qual, de um número de possibilidades, foi escolhida. Os dispositivos físicos desta classe são: "buttom box", "light pen" e teclados do terminal, todos eles usados para seleção em "menu".

5) "String": o dispositivo lógico de entrada "string" fornece ao programa de aplicação uma cadeia de caracteres. Em muitos casos a cadeia é reproduzida caractere por caractere em um dispositivo de saída. Tipicamente o dispositivo físico de entrada dessa classe é o teclado do terminal de vídeo.

6) "Pick": o dispositivo lógico de entrada "pick" fornece ao usuário do programa de aplicação o nome de um segmento que identifica o objeto (ou objetos) que está(ão) sendo indicado(os) pelo dispositivo. Tipicamente o dispositivo físico desta classe é a "light pen".

5.2.4 - O CONCEITO DE ESTAÇÃO DE TRABALHO

5.2.4.1 - DEFINIÇÕES

Uma das maiores forças do GKS é a sua formalização do conceito de *Estação de Trabalho*. Este conceito identifica uma estação de trabalho como uma unidade que consiste de *uma superfície de display e dispositivos de entradas gráficas*, tais como teclados de vídeo, mesas digitalizadoras, penas luminosas, etc. Este conceito vai além dos primeiros pseudo-padrões gráficos que trabalham principalmente com dispositivos apenas de saída.

O GKS permite ainda uma transferência de processamento para estações de trabalho com inteligência local.

A Estação de Trabalho como um todo é tratada, no GKS, como uma unidade lógica.

Uma Estação de Trabalho totalmente equipada

- tem uma superfície de display endereçável;
- permite o uso de menores espaços de display que o máximo,enquanto garante que não será gerado um display fora do espaço físico especificado;
- suporta vários tipos de linhas, cores, fontes de texto,tamanhos de caracteres, etc.;
- tem um ou mais dispositivos de entrada para cada classe de primitiva de entrada;
- permite entradas do tipo "request", "event", e "sample". Ver Seção 5.2.8, modos de interação.

Uma instalação real da estação de trabalho pode ou não estar equipada com todas essas capacidades. O programa de aplicação deve perguntar, via GKS (veja Seção 5.2.10 - funções de consulta) quais são as capacidades disponíveis na estação e adaptar seu comportamento

de acordo com elas. Caso haja a requisição de capacidades não-disponíveis, um erro padrão é definido.

As Estações de Trabalho são identificadas pelo programa de aplicação através do uso de um *Identificador de Estação*. Sempre que um programa de aplicação quiser usar uma estação, ele deve primeiro requisitar a *abertura* desta estação pelo GKS. A ação de abertura associa a estação ao terminal gráfico correspondente e dá à aplicação acesso a todas as estações ativas. A entrada e manipulação de segmentos podem ser realizadas por qualquer estação aberta.

Os dispositivos de entrada de uma estação de trabalho particular são identificados pelo identificador de estação de trabalho;pela classe do dispositivo de entrada ("stroke", "valuator", "choice", etc.);epelo número do dispositivo de entrada.

5.2.4.2 - A "WORKSTATION" DO GKS EM COMPARAÇÃO COM O SISTEMA CORE

O aparecimento de primitivas de saída gráficas variarão significativamente se os dispositivos de saída gráfica apresentarem diferentes características. No Sistema GSPC Core o aparecimento de uma primitiva é associado com a própria primitiva. No GKS, o aparecimento de uma primitiva é definida por dois estágios.

No primeiro estágio, um atributo simbólico é associado à primitiva, enquanto no segundo estágio este atributo é mapeado nas capacidades individuais de cada estação de trabalho, determinando assim a aparência final no terminal gráfico. Com o GKS, ambos os estágios estão sobre o controle total do programa de aplicação. Usando este mecanismo o programa de aplicação pode, por exemplo, desenhar a mesma primitiva "polyline" como uma linha vermelha em uma "plotter", ou como uma linha pontilhada em um display de raios catódicos sem haver a necessidade de regeneração da primitiva.

No GKS isto significa que conceitualmente, num primeiro passo, uma *figura independente da estação de trabalho é desenhada em uma superfície abstrata* e, num segundo passo, *para cada estação de trabalho ativa*, esse dado gráfico é combinado com dados dependentes da estação de trabalho, com a figura sendo então *enviada para a superfície de visão física associada a cada uma dessas estações*. A Figura 4 ilustra este conceito.

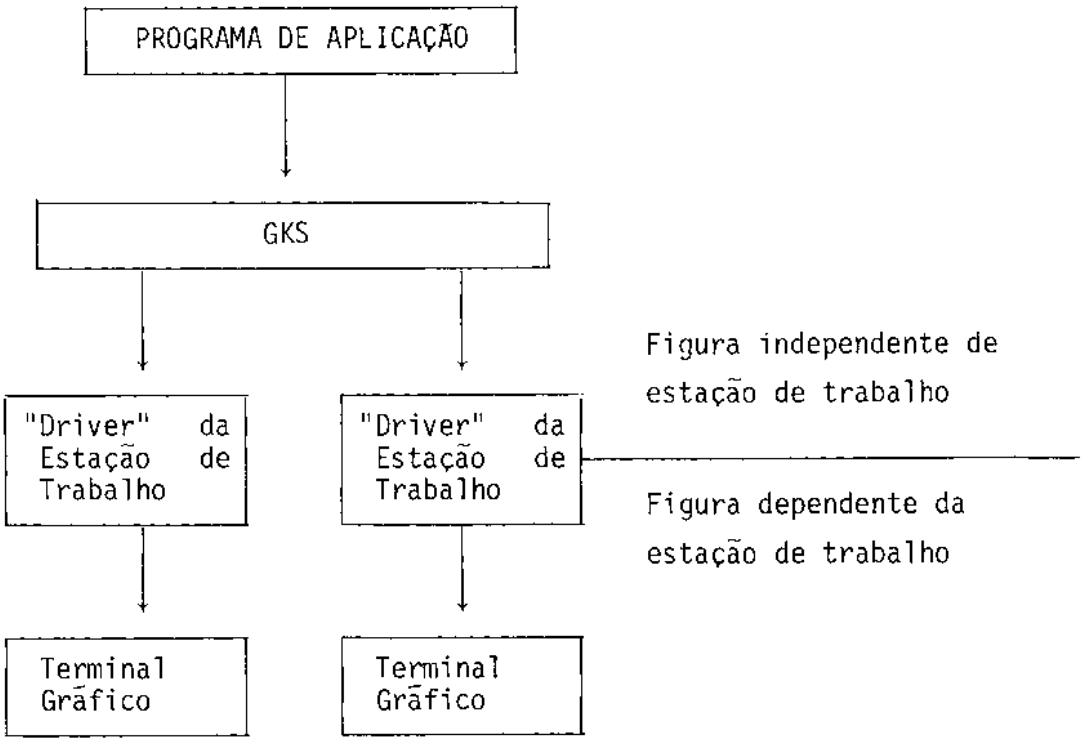

Fig. 4 - Percurso de uma figura de entrada do GKS para a estação de trabalho.

O conjunto de características dependentes da estação de trabalho, mostrados na Figura 5, que podem ser definidas separadamente sobre o controle total do programa de aplicação para cada estação, contêm:

- mapeamento da coordenada de dispositivo normalizada, NDC, para coordenada de dispositivo. Veja Seção 5.2.7 (O Sistema de Coordenadas);

- definição de representação de pena, i.e., correlacionamento de um número de pena lógica com uma pena física predefinida ou com um tipo de linha, largura de linha, cor, etc.);
- definição da representação de texto, ou seja, correlacionamento de um número de texto lógico com um gerador de caractere ("hardware") predefinido ou com um conjunto de atributos de texto, tais como fonte de texto, tamanho de caractere, espaçamento entre caracteres, etc.;
- escolha de uma qualidade de espaços de display, isto é, qualidade de papel para uma "plotter", ou cor de fundo para um dispositivo de "display raster".

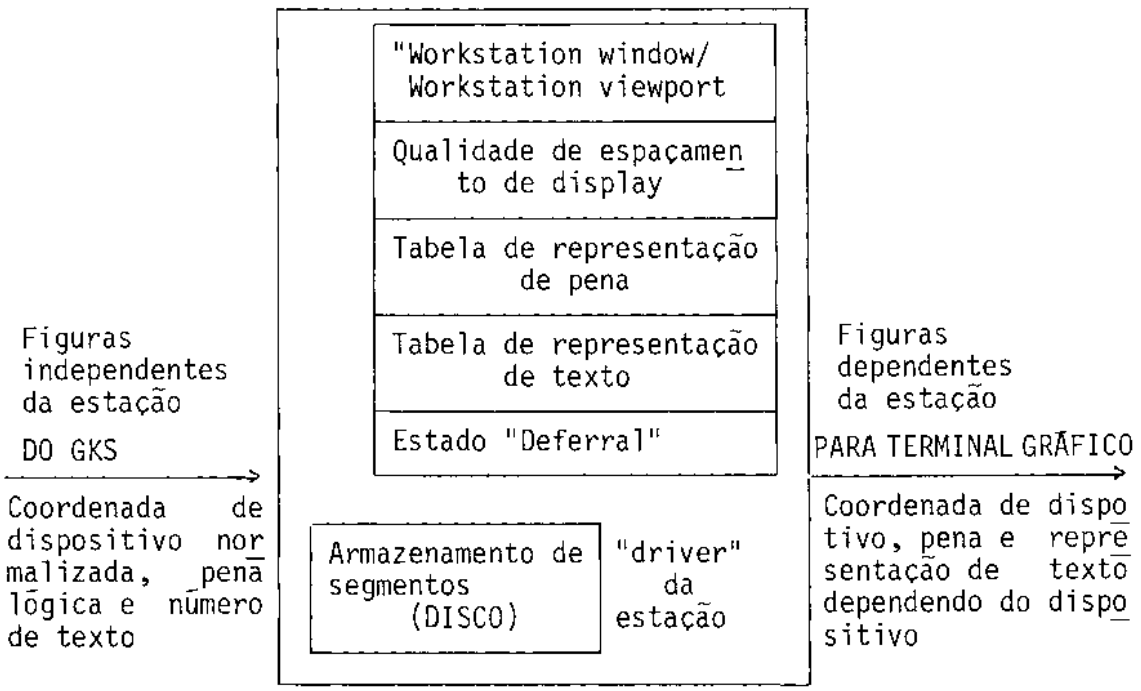

Fig. 5 - Características dependentes da estação de trabalho.

5.2.4.3 - CONCLUSÃO

Ao invés de trabalhar com dispostivos gráficos de entrada e saída, por exemplo, "plotter", displays gráficos, "mouse", etc., separadamente, como dispositivos lógicos de E/S, o GKS trata coleções des

ses dispositivos como uma unidade funcional, a estação de trabalho.

Esse conceito adapta-se melhor à tendência atual de desenvolvimento de sistemas gráficos inteligentes: um operador manipula uma combinação de dispositivos gráficos de entrada e saída como uma unidade operacional.

5.2.5 - SEGMENTOS E SEUS ATRIBUTOS

Define-se um *segmento gráfico* como um local onde saídas gráficas podem ser armazenadas e reutilizadas mais tarde durante a execução de um programa.

O GKS permite o agrupamento de um conjunto de primitivas gráficas de saída para forma um segmento gráfico. Relaciona-se a cada segmento diferente um nome de segmento ou um indicador de segmento,que possibilita assim que ele seja manipulado como uma entidade única.

Cada segmento tem uma quantidade de *atributos de segmentos* associados a ele, os quais permitem ao usuário modificar a aparência do segmento quando ele é reusado.

No GKS, os atributos associados a cada segmento gráfico são:

- Transformação de Segmento: é uma operação de transformação nas coordenadas de definição do segmento. As possíveis transformações são: *translação*, *ângulo de rotação*, *posição absoluta* e *fator de escala*. Este atributo é muito usado quando se necessita mover objetos ou parte deles na tela em resposta a alguma computação ou decisão do operador.
- Corte de Segmento: o corte de um segmento é realizado sempre a posteriore a uma transformação. Uma primitiva de saída em um segmento é transformada por normalização e por transformações de segmento e, então, se o indicador de corte estiver definido pa

ra cortar, ela é cortada segundo o "viewport" (da transformação de normalização) que estava ativo quando a primitiva foi colocada no segmento. Em cada segmento está armazenado um retângulo de corte para cada primitiva que ele contém.

- Visibilidade do Segmento: o atributo de segmento visibilidade determina se um segmento será mostrado (visível) no dispositivo de saída ou não.
- Realçamento de Segmento: este atributo permite que uma ou mais primitivas do segmento sejam mostradas na tela com um realce maior que as demais. Por exemplo, o segmento aparece piscando ou em cores mais brilhantes na tela.
- Prioridade de Segmento: este atributo permite que se defina uma prioridade para cada primitiva que compõe o segmento. No caso de decisão entre dois segmentos como, por exemplo, de superposição de segmentos na tela, o de maior prioridade terá predominância.

Os segmentos são muito usados quando, durante a execução de um programa deseja-se duplicar parte de uma figura. Por exemplo, quando a roda de um carro em uma figura é criada, deve-se colocá-la em um segmento com uma determinada identificação. Quando se vai desenhar as outras três rodas do carro basta proceder a uma transformação nesse segmento.

5.2.6 - O ARMAZENAMENTO DE SEGMENTOS - WISS e WDSS

Já foi mostrado como os segmentos são criados e manipulados, porém não foi dada qualquer indicação do lugar onde esses segmentos são armazenados.

Conceitualmente, os segmentos são armazenados nas estações de trabalho ativas, no momento em que eles são criados.

Um implementação particular do GKS poderia armazenar centralmente segmentos, mas, para estações inteligentes com armazenamento

local, o ideal seria armazenar os segmentos em estações de trabalho. Isto aumentaria o desempenho visto pelo usuário em termos de manipulação de segmento. Mesmo que o armazenamento fosse central, a implementação deveria ser de tal forma que o usuário pudesse imaginar que os segmentos estivessem armazenados na sua estação de trabalho.

Seguindo essas idéias, o GKS fornece dois mecanismos diferentes de armazenamento de segmentos: o *WISS "Workstation Independent Segment Storage"* e o *WDSS "Workstation Dependent Segment Storage"*.

No WISS o armazenamento é central e independente das estações de trabalho existentes. Porém, o WISS é tratado pelo GKS como uma estação de trabalho.

O WDSS permite armazenar segmentos naquelas estações que estão ativas quando os segmentos são criados. Assim, se o WISS está ativo quando um determinado segmento é criado, este segmento será armazenado no WISS e também no WDSS de cada estação de trabalho ativa.

Os segmentos armazenados no WISS têm exatamente os mesmos atributos de segmentos que os segmentos armazenados no WDSS e são manipulados pelas mesmas funções do GKS. O GKS possibilita ainda a transferência de segmentos do WISS para qualquer outra estação de trabalho.

5.2.7 - O SISTEMA DE COORDENADAS

Três sistemas de coordenadas, no GKS, permitem a manipulação de dados gráficos.

- Coordenada Universal - WC ("World Coordinate"): é usada pelos programas de aplicação e define objetos gráficos em termos de dimensões reais. O termo coordenada universal é usado para definir um sistema de coordenadas utilizado para apresentar as saídas gráficas para o GKS.

- Coordenada de Dispositivo Normalizada - NDC ("Normalized Device Coordinate"): define-se um dispositivo virtual ou normalizado como aquele que tem uma superfície de display visível na faixa de 0 a 1 em ambas as direções X e Y. As coordenadas destes dispositivos são chamadas coordenadas de dispositivo normalizadas-NDC. Para produzir uma saída visível, as coordenadas universais que definem a saída devem ser mapeadas em posições de coordenadas dentro da unidade quadrada de 0 a 1 da coordenada de dispositivo normalizado. Um processo chamado *transformação de normalização* converte as coordenadas universais, WC, para coordenadas de dispositivo normalizadas, NDC. Múltiplas transformações de normalização podem compor uma figura no espaço NDC a partir de diferentes seções de um espaço WC, ou mesmo de diferentes espaços WC.

- Coordenadas de Dispositivos ou Coordenadas de Estação de Trabalho ("Workstation Coordinates"): uma transformação adicional, chamada *transformação de estação de trabalho* ("Workstation transformation"), possibilita selecionar qualquer porção de uma figura no espaço de coordenada de dispositivo normalizada e posicioná-la em diferentes setores de diferentes dispositivos. Transformações de estação de trabalho separadas podem existir para diferentes estações, o que permite que cada uma mostre diferentes aspectos de uma figura.

A Figura 6 ilustra os conceitos relativos ao sistema de coordenadas e as possíveis transformações no GKS.

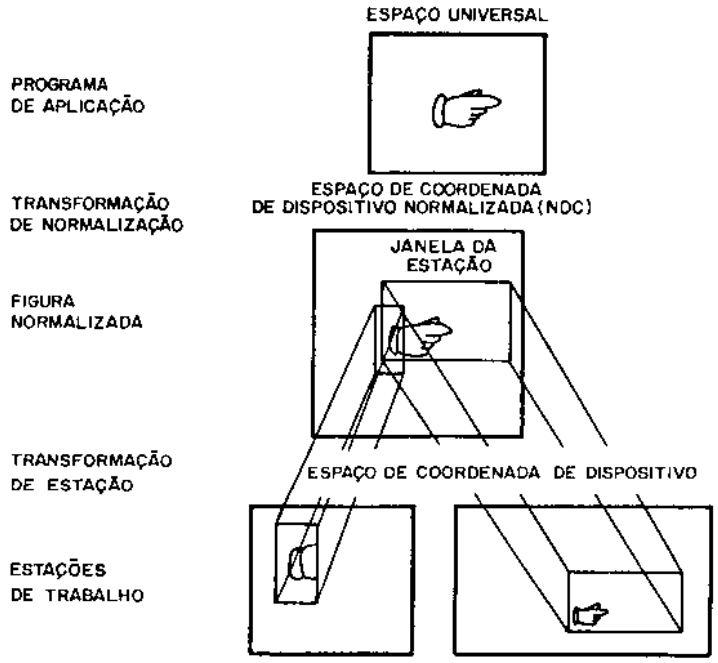

Fig. 6 - O sistema de coordenadas e as transformações no GKS.

5.2.8 - MODOS DE INTERAÇÃO

O GKS possibilita que cada dispositivo de entrada opere em 3 modos diferentes: o *modo requisição* ("request mode"), o *modo amostragem* ("sample mode") e o *modo evento* ("event mode").

Os três modos de operação dos dispositivos lógicos de entrada especificam quem, o operador ou o programa de aplicação, é dono da iniciativa durante a execução do programa. Uma entrada no modo amostragem é adquirida diretamente pelo programa de aplicação; uma entrada no modo requisição é produzida pelo operador em resposta direta ao programa de aplicação; e uma entrada no modo evento é gerada assincronamente pelo operador.

O funcionamento desses três modos de interação, no GKS, são descritos a seguir:

1) Modo Requisição: neste, o programa de aplicação e o processo de entrada trabalham alternadamente. Primeiro o programa de aplicação requisita uma entrada e então aguarda uma resposta. O processo de entrada é ativado pela requisição, libera o dado para o programa de aplicação e retorna para um estado de espera. Um ou outro processo está ativo mas não ambos ao mesmo tempo.

2) Modo Amostragem: neste modo o programa de aplicação e processo de entrada estão ambos ativos ao mesmo tempo, porém o programa de aplicação é o parceiro dominante. O processo de entrada trabalha em "background" fornecendo sempre o último dado de entrada do dispositivo, que pode ou não ser utilizado pelo programa de aplicação. Ao mesmo tempo o programa de aplicação continua em execução, fazendo amostras e utilizando o dado de entrada corrente do dispositivo, sempre que ele precisar.

3) Modo Evento: neste modo o programa de aplicação e o processo de entrada estão novamente ativos, mas o parceiro dominante é o processo de entrada. Ele libera dados de entrada para o programa de aplicação e aguarda uma ação dele que dependerá desse dado recebido. O operador controla quando o dado de entrada está disponível e, efetivamente, dirige a interação. Quando um dispositivo de entrada é colocado no modo evento, todos os valores de entrada são colocados numa fila de entrada para o programa de aplicação ler. Este, lê cada valor em ordem de entrada e trata-o antes de manipular o próximo.

5.2.9 - NÍVEIS DE IMPLEMENTAÇÃO

O GKS fornece vários níveis de implementação e a funcionalidade fornecida para cada nível está estritamente definida, veja Tabela 2.

A definição de níveis de implementação possibilita a escolha de capacidade do sistema de acordo com as necessidades do progra

mador. Desta forma, se o programador necessita uma funcionalidade mínima, ele pode conservar a independência de dispositivo e compatibilidade de "software" sem suportar a carga requerida pelos recursos de alto nível.

TABELA 2

NÍVEIS DE IMPLEMENTAÇÃO DO GKS

NÍVEL DE SAÍDA \ NÍVEL DE ENTRADA	A	B	C
M	Nenhuma entrada, controle mínino, só atributos individuais, subconjunto de funções de saída.	Entrada no modo requisição, definição de modo e inicialização de dispositivos, sem entrada "Pick"	Entradas no modo amostragem e evento, sem entrada "Pick"
0	Nenhuma entrada, controle mínimo, grupos de atributos predefinidos	Entrada no modo requisição, definição de modo e inicialização de dispositivo	Entradas no modo amostragem e evento
1	Todo tipo de saída, suporte de segmento gráfico e armazenamento em "metafile"	Igual a 0B acima mais entrada "Pick" com requisição, definição e inicialização para entrada "Pick"	Entradas no modo amostragem e evento para dispositivo lógico "Pick"
2	Igual a 1A acima mais armazenamento de segmento independente da estação	Igual a 1B acima	Igual a 1C acima

5.2.10 - FUNÇÕES DE CONSULTA

Para auxiliar o programador na escrita de programas que sejam independentes do dispositivo, o GKS fornece um conjunto de funções de consulta.

Essas funções auxiliam o programador na consulta sobre as características individuais de cada estação de trabalho disponível e sobre a implementação GKS que está sendo usada.

Dessa forma, o programa pode fazer escolhas inteligentes baseadas nas facilidades disponíveis. Por exemplo, se não se tem disponibilidade de cores, o programa deve usar tipos de linhas diferentes para discriminar objetos de cores diferentes.

5.2.11 - LISTAS DE ESTADO NO GKS

Internamente ao GKS e transparente para o programador de aplicação existem listas de estado que contêm informações relevantes sobre o estado corrente do GKS.

O programador de aplicação, na realidade, não precisa saber que elas existem. Entretanto, é usual fazer-se algumas considerações sobre essas estruturas.

As principais listas de estado no GKS são:

- Estado de operação: contém um único valor que fornece o estado atual do GKS.
- Tabela de descrição do GKS: é uma tabela que dá informação sobre uma implementação particular do GKS. Por exemplo, ela contém informações de quantas estações de trabalho podem estar abertas ao mesmo tempo, o número permitido de diferentes transformações de normalização, e os tipos de estações de trabalho disponíveis.

- Lista de estado do GKS: esta é a lista principal para o lado virtual do GKS. Ela contém informações sobre todos os conjuntos de atributos correntes, as transformações de normalização, qual segmento está aberto, etc. A fila de entrada é parte desta lista.

- Tabelas de descrição de estação de trabalho: estas tabelas são definidas pelo gerenciador de instalação local e contêm informações sobre as características das estações de trabalho locais. Existe uma tabela para cada tipo de estação disponível.

- Lista de estado da estação: esta contém informações sobre as estações que estão abertas. À cada estação, aberta ou ativa, é associada uma lista de estado da estação. Esta é inicializada a partir de valores contidos na tabela de descrição de estação como, por exemplo, o tamanho de tela do dispositivo de saída da estação.

Essas tabelas contêm todo o conhecimento relevante atual do GKS e são acessadas, pelo programa de aplicação, através do conjunto de funções de consulta.

CAPÍTULO 6

CONCLUSÃO

A tendência atual na área de padronização de sistemas de computação gráfica, é o desenvolvimento e adoção dos padrões GKS, VDI, VDM, IGES e NAPLPS, cada qual no seu próprio nível de atuação dentro do sistema gráfico. Estes padrões são frutos de incessantes trabalhos dos órgãos de padronização em conjunto com fabricantes e usuários desses sistemas.

Esses 5 padrões já se encontram em fase adiantada no processo de adoção como padrões internacionais. Em particular, o GKS foi o primeiro que obteve o título de *"draft international standard"*, pela ISO, que é o penúltimo degrau do processo de padronização.

Numa comparação estrutural com o sistema GSPC CORE, o GKS cujas bases estão contidas nesse mesmo CORE, tem uma certa vantagem devido, principalmente, aos seus conceitos de *estação de trabalho* e de *padrão "Binding" com linguagens de programação* que garantem uma maior independência de dispositivos de E/S e aumentam o grau de transportabilidade de "softwares" aplicativos.

Além disso, o processo de desenvolvimento do GKS sempre esteve diretamente ligado a órgãos tradicionalmente produtores de padrões, os quais conduziram seus esforços de desenvolvimento sempre voltados para esse aspecto, o que tornou o GKS um padrão mais bem estruturado.

Apesar dessas razões, o sistema CORE, que foi o primeiro a ser desenvolvido e a englobar dados gráficos de 2 e 3 dimensões, tornou-se, através dos anos, um padrão informal devido à sua alta taxa de implementação nos principais sistemas gráficos existentes. Estima-se em cerca de 200.000 os usuários de sistemas implementados com base nos conceitos do CORE. Por outro lado, o CORE já é considerado um padrão

ACM, que é um órgão paralelo de padronização tal qual o IEEE.

Esses fatos demonstram que as batalhas travadas para padronização deste ou daquele sistema de "software" gráfico ainda não terminaram. Nesse contexto estão envolvidos, não só interesses tecnológicos de usuários e fabricantes, mas também, implicações políticas e econômicas que estão fora do controle puramente científico.

A motivação maior que mantém todo esse processo de desenvolvimento de padrões é realimentada pelo fato de que, apesar de todas as batalhas e da grande quantidade de esforços que devem ser dispendidos para definir, desenvolver e adotar um padrão, esse trabalho é necessário e tem consequências extremamente benéficas para os fabricantes e usuários envolvidos com os diversos tipos de sistemas gráficos existentes. A própria *definição de padronização* pode nos dar uma idéia dos benefícios que ela proporciona.

PADRONIZAÇÃO: Redução dos objetos do mesmo gênero a um só tipo, unificado e simplificado, segundo um modelo preestabelecido (Aurélio Buarque de Holanda).

BIBLIOGRAFIA

BONO, P.R.; ENCARNAÇÃO, J.L.; HOPGOOD, F.R.; HAGEN, P.J.W. GKS the first graphics standard. *IEEE Computer Graphics*, *2*(5): 9-23, July, 1982.

DAVIS, A. Proprietary/standard graphics software mix to give more. *Computer Design*, *23* (5): 167-234, May 1984.

ENCARNAÇÃO, J.L.; ENDERLE, G.; KANSY, K.; NEES, G.; SCHLECHTENDAHL, E. G.; WEISS, J.; WIBKIRCHEN, P. The workstation concept of GKS and resulting conceptual differences to the GSPC core system. *Computer Graphics*, *14*(3): 226-230, July, 1980.

HINDIN, H.J. Graphics standards finally start to sort themselves out. *Computer Design*, *23*(5): 167-180, May 1984.

HOLLAND, G.L. NAPLPS standard defines graphics and text communication. *EDN*, *30*(1): 179-192, January 1985.

HOPGOOD, F.R.A.; DUCE, D.A.; GALLOP, J.R.; SUTCLIFFE, D.C. *Introduction to the Graphical Kernel System - GKS*, London, Academic, 1983.

LANGHORST, F. VDI promises graphics software portability. *Computer Design*, *23*(5): 197-203, May 1984.

MACHOVER, C.; MAYERS, W. Interactive computer graphics. *Computer*, *17* (10): 145-161, Oct. 1984.

MICHENER, J.C.; DAM, A.V. A functional overview of the core system with glossary. *ACM Computing Surveys*, *10*(4): 381-388, Dec. 1978.

NEWMAN, W.M.; DAM, A.V. Recent efforts towards graphics standardization. *ACM Computing Surveys*, *10*(4): 365-380, Dec. 1978.

ROSENTHAL, D.S.H.; MICHENER, J.C.; PFAFF, G.; KESSENER, R.; SABIN, M. The detailed semantics of graphics input devices. *Computer Graphics* *16*(3): 33-38, July, 1982.

SANTOS, F.A.S. *Um sistema gráfico independente de dispositivo e interativo*. Dissertação de mestrado em Ciências de Computação. Rio de Janeiro, Pontifícia Universidade Católica, fev. 1985.

TANAKA, A.K. *Sistema gráfico de apoio a projeto de engenharia.* Dissertação de mestrado em Ciências de Engenharia de Sistemas. Rio de Janeiro, Instituto Militar de Engenharia, jan. 1985.

WAGGONER, N.C. Based graphics software adapts to changing technologies. *EDN*, *29*(11): 127-133, May 1984.

WATERBURY, R. Integration key to high level CAD/CAM. *Assembly Engeneering*, *26*(3): 40-44, Mar. 1983.