G K S - 3 0 0

PASCAL-Implementierung des Grafischen Kernsystems (GKS)
auf dem SIEMENS-Grafikarbeitsplatz SIGRIS

Karl-Heinz Klein, Günther Schmitgen

Gesellschaft für Mathematik und Datenverarbeitung mbH Bonn
Schloß Birlinghoven, Postfach 1240
D-5205 St. Augustin 1

Das Grafische Kernsystem GKS

Bei der Standardisierung grafischer Grundsoftware wurden in den letzten Jahren entscheidende Fortschritte erzielt. Seit 1976 ist das Grafische Kernsystem GKS als Normvorschlag in einem Unterausschuß des Deutschen Instituts für Normung (DIN) erarbeitet worden. Dieser Entwurf wurde mit einigen Modifikationen auch von der International Organization for Standardization (ISO) akzeptiert; bemerkenswert ist, daß er dem amerikanischen Konkurrenzvorschlag ACM SIGGRAPH's Core vorgezogen wurde. Ende 1982 erschien GKS als DIN-Normentwurf (Rosadruck) /1/; die Veröffentlichung als ISO Draft International Standard steht unmittelbar bevor.

Der Bedarf nach Normung resultierte aus dem Wirrwarr auf dem Grafiksektor; da für verschiedene Hardware-Konfigurationen die Grafikpakete von Struktur und Leistungsumfang sehr unterschiedlich waren, ließ sich bei Umstellungen meist ein Neuprogrammieren der Anwendungssoftware nicht vermeiden. Der Anspruch von GKS ist es, hardware-, software- und anwendungsunabhängig einen Satz von Grundfunktionen - mittlerweile sind es etwa 180 - zu definieren, der es gestattet,alle grafischen Geräte in einheitlicher Weise anzusprechen und als Werkzeug für die Programmierung beliebiger Aufgaben aus dem Bereich 2-dimensionaler Linien- und Rastergrafik eingesetzt werden kann; GKS ist jedoch kein "Turn Key" System für fertige Problemlösungen etwa in der Kartographie oder im CAD-Bereich; solche Systeme sollen vielmehr "schalenförmig" auf dem "grafischen Betriebssystem" GKS aufsetzen.

Von Beginn der Normungsbestrebungen an haben Mitarbeiter der Gesellschaft für Mathematik und Datenverarbeitung mbH Bonn (GMD) in dem DIN-Ausschuß NI-UA 5.9 "Verarbeitung graphischer Daten" mitgewirkt. Außerdem wurde begleitend eine Pilotimplementierung von GKS auf Siemens - Großrechnern (BS-2000) durchgezogen. Diese wurde als vorläufige GKS-Version in verschiedenen grafischen Anwendungsprojekten des Instituts erfolgreich als Grundsoftware eingesetzt. Durch die Kombination von Theorie und Praxis konnten entscheidende Beiträge für die Ausgestaltung der GKS-Norm geliefert werden.

Die GKS-Realisierung G K S - 3 0 0

1980 schloß die Gesellschaft für Mathematik und Datenverarbeitung (GMD) mit der Fa. Siemens, Karlsruhe, einen Kooperationsvertrag über die Implementierung des Graphischen Kernsystem (GKS) auf dem Grafikarbeitsplatz SIEMENS-SIGRIS. Die Entwicklungsarbeiten konnten Ende 1982 mit der Übergabe des Systems an Siemens erfolgreich abgeschlossen werden. Das Produkt GKS-300 entspricht

- GKS-Version 7.0
- GKS-Level 2 c

Der Grafikarbeitsplatz SIGRIS besteht aus

- Siemens-Prozeßrechner der Serie 300
- Grafik-Sichtgerät Tektronix 102A (oder verwandte Typen)
- alphanumerischem Sichtgerät
- grafischem Tablett
- vorgesehen sind außerdem Plotter, Digitalisierer, hochauflösende Rastersichtgeräte und Spracheingabe

Der Arbeitsplatz ist geeignet für Aufgaben aus der Karthographie, CAD / CAM, Prozeßsteuerung, Businesgraphics oder Grafik im Lehr- und Forschungsbereich. Er wird teils als Grafiksatellit an BS-2000 Rechnern mit dem Grafiksystem SICAD, teils "stand alone" eingesetzt. Ziel der GKS-Implementierung war es, dem Benutzer des Arbeitsplatzes ein komfortables Grafik-Grundsystem für die Erstellung von Anwendungssoftware zu bieten.

Die Rechner der Serie 300 haben eine 16 bit Wortstruktur; daraus ergibt sich eine Adressierungsschranke von 64 K Worten pro Task; bei entsprechendem Ausbau können jedoch mehrere Programme paralell ablaufen. Für den Betrieb von GKS-300 ist als Systemausbau erforderlich

- Hauptspeicher ca. 200 K Worte (16 bit)
- Plattenspeicher ca. 30 M Byte
- Betriebssystem ORG-PV (virtuelles Betriebssystem)

Das Grafiksichtgerät ist eine Speicherröhre (Auflösung 4096 x 3072 Punkte) mit begrenzter Refreshmöglichkeit (ca. 1000 Vektoren flickerfrei). Der Bildwiederholspeicher für den Refresh befindet sich im Hauptspeicher; die Datenübertragung erfolgt über Selektorkanal. Zum Ansteuern der Grafikgeräte existiert ein in Assembler geschriebenes Treiberpogramm SIGRIS, das Code-Generierung und I/O abwickelt und den Refresh Puffer (8K Worte im Hauptspeicher) verwaltet. Darüber hinaus kann SIGRIS höhere grafische Funktionen wie Transformationen, Clipping und Bildteilmanipulation ausführen; jedoch sind viele Funktionen sehr hardwarenah spezifiziert, wichtige wie z.B. Segmenttransformationen, und die Eingabearten Choice und Pick fehlen.

Die Bereitstellung von GKS auf dem Arbeitsplatz dient dem Zweck, ein standardisiertes grafisches Basissystem zu bieten, um weitgehende Geräteunabhängigkeit und Portabilität bei Anwendungssoftware zu erreichen; wichtig waren dabei insbesondere das volle Spektrum der Eingabefunktionen, die Möglichkeit von Segmentmanipulationen und Metafile als Instrument zum Archivieren und Transportieren von Grafiken.

Ziel der GMD bei der Entwicklung von GKS-300 war es, das "know how" über Grafik-Systeme in ein Softwareprodukt zu transferieren, das bezüglich Reife und Wartbarkeit industriellen Ansprüchen genügt. Besonderes Interesse bei der Implementierung galt der Frage, wieweit sich trotz der Kleinrechnerrestriktionen der volle GKS-Funktionsumfang so realisieren läßt, daß sich bei Anwendungssystemen ein akzeptables Laufzeitverhalten ergibt.

Die Aufgabenstellung von GKS-300 beinhaltete, das volle in GKS vorgesehene Funktionsspektrum sowohl bei der grafischen Ausgabe, Strukturierung und Archivierung als auch bei der Eingabe zu realisieren, soweit es bei der gegebenen Hardware machbar ist. Hinzu kamen weitere Anforderungen, die beim Design zu berücksichtigen waren:

- Sprachinterface für verschiedene höhere Programmiersprachen;
- gleichzeitige Benutzung durch mehrere Anwender;
- effiziente Bearbeitung umfangreicher Programme;
- volle Interaktivität mit akzeptablem Dialogverhalten;
- Wartbarkeit, Modifizierbarkeit und Erweiterbarkeit des Systems.

Um Anwendungsprogramme realistischer Größe betreiben zu können, ist es auf Grund der beschränkten Adressierbarkeit nötig, GKS nicht als Unterprogrammpaket zum Anwendungsprogramm zu binden, sondern als separate Task in einem eigenen Laufbereich unterzubringen. Das Anwendungsprogramm bindet nur einen relativ kleinen Interfacemodul zur Abwicklung der Inter-Prozeß-Kommunikation. Mehrere Programme können gleichzeitig GKS-300 benutzen, ohne daß der gesamte Code mehrfach geladen werden muß. Für die variablen Datenteile wurde ein Swapping eingebaut.

Die Modulstukturierung für die gegebene Konfiguration ist in Abb. 1 dargestellt. Die Anwendungsprogramme können in verschieden Sprachen programmiert sein. Sie binden einen für diese Sprache passenden GKS Interface Modul, der die Kommunikation mit dem zentralen GKS übernimmt. Da Grafikprogramme meist umfangreiche Datenmengen speichern müssen, ist es wesentlich, daß der GKS-Anteil am Adreßraum klein gehalten werden kann. Wegen des Umfangs von GKS ist eine Overlay-Strukturierung des zentralen GKS-300 Bausteins unvermeidlich. Die Treiberbausteine können Unterprogramme oder eigene Tasks sein oder aus einer Kombination von beidem bestehen; für die Bearbeitung von asynchronem Input sind auf jeden Fall eigene Prozesse erforderlich. Der Austausch von Parametern (Kommandos, grafische Daten) erfolgt über globale COMMON-Bereiche (COM-RECORD); für die Koordinierung der Prozesse untereinander (Aktivieren,

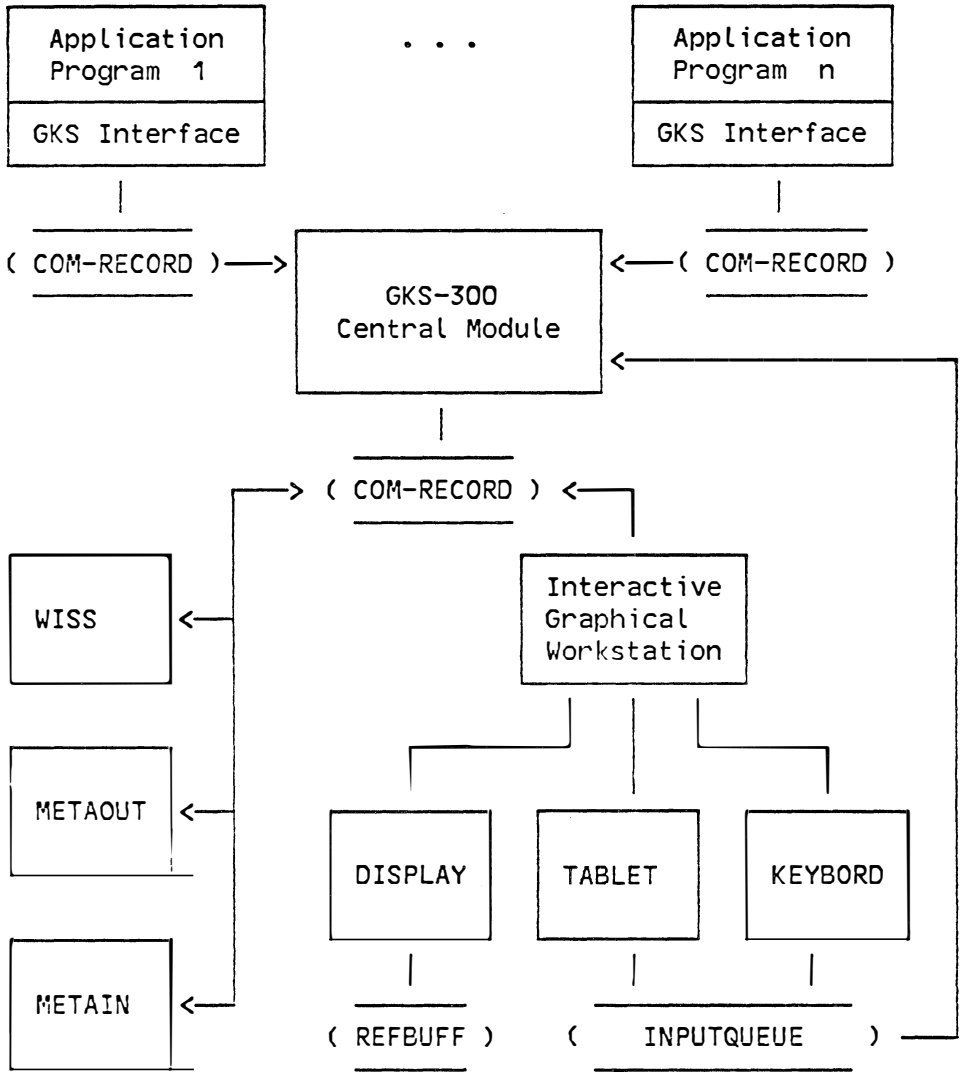

Abb. 1 Module, Kontrolle und Datenfluß in GKS-300

Warten) bieten Koordinierungszähler, die vom Betriebssystem verwaltet werden, ein sehr leistungsfähiges Werkzeug. Tablett- und Tastatureingaben werden als Event Report in einem Ringpuffer abgelegt und über Koordinierungszählerbetätigung dem Zentralmodul gemeldet. Um Doppelarbeit bei der Entwicklung, Wartung und Anpassung an neue Hardware zu vermeiden, werden die hardwarebezogenen Treiberfunktionen über die Hersteller-Software SIGRIS /4/ abgewickelt.

Größe der einzelnen Bausteine:

GKS - Interfacemodul zum Anwendungsprogramm	4 - 12 KW
COM-RECORD (globaler Datenbereich)	2 KW
Gerätetreiber	14 KW
Refresh Puffer	8 KW
Eingabe Ringpuffer	2 KW
zentraler GKS Baustein (mit Overlay)	28 KW
Platz für GKS-Listen	9 KW

Programmiersprachen

Da es sich bei GKS-300 um selbstständige Moduln handelt, war die Implementierungssprache frei wählbar. Lediglich die Interfacemoduln sind angepaßt an die Srache der Anwendungsprogramme zu erstellen. Da es sich hierbei im wesentlichen nur um die Bereitstellung der ENTRY-Namen und Transport der Parameter handelt, läßt sich, wenn erst einmal die nötigen Vorarbeiten gemacht sind, die Generierung durch Einsatz eines Makro-Prozessors zu einem Großteil automatisieren. Da auch die Schnittstellen in den anderen Moduln automatisch generiert werden, ist garantiert, daß sie deckungsgleich sind. Ausgangsmaterial sind die Funktionsdefinitionen und Tabellen in der GKS-Norm /1/, die mit Hilfe von Editierprogrammen in Prozedur-Deklarationen mit zugehörigen Parametern und in Definitionsteile für interne Listen überführt werden. Für Testzwecke und einfache Anwendungen "per Hand" wurde mit einem ähnlichen Generierungsmechanismus ein interaktives Testprogramm erstellt, das es gestattet, im Dialog alle Funktionen von GKS anzusprechen. Diese Methode erwies sich als sehr nützlich, als während der Entwicklungsarbeiten der Übergang von GKS Version 6.2 zu Version 7.0 erhebliche Veränderungen brachte.

Die Hauptimplementierungssprache ist Pascal. Die Vorzüge brauchen hier nicht im einzelnen erwähnt zu werden; bemerkt sei, daß auch in der GKS-Beschreibung eine Pascal-ähnliche Diktion verwendet wird. Durch den Einsatz von Pascal werden die Programme "lesbar"; dadurch kann der Dokumentationsaufwand geringer gehalten werden und die Wartungsfreundlichkeit ist höher. Nicht unerheblich war es auch, daß mit PASCAL 300 ein leistungsfähiger Compiler verfügbar war.

Allerdings fehlen in Pascal Srachmittel für Inter-Prozeß-Kommunikation und das Ansprechen von globalen Datenbereichen. Hierzu wurden Prozeduren in Assembler geschrieben. Aus sprachtechnischen Gründen und um das Laufzeitverhalten zu verbessern, wurden auch Teile der Datenverwaltung in Assembler codiert. Ferner war Assembler nötig, um binäre Daten bitweise zu packen und bearbeiten und so Seicherplatz zu sparen.

Wenn man die Assemblerroutinen auf einem anderen Rechner nachbildet, ist der geräteunabhängige Teil von GKS-300 portierbar, vorausgesetzt, die Kapazität des Kompilers ist ausreichend. Von den Möglichkeiten von PASCAL 300 wurden getrenntes Übersetzen von Prozeduren und OVERLAY Strukturierung genutzt; damit das System in sich konsistent bleibt, werden alle nicht lokalen Deklarationsteile separat gehalten und erst zum Kompilieren in die einzelnen Programmteile hinein kopiert. Hierfür wurde ein Dienstprogramm erstellt.

Sprachschnittstellen wurden für FORTRAN und Pascal realisiert, da FORTRAN die am weitesten verbreitete Sprache für grafische Anwendungen ist, daneben jedoch Pascal immer mehr an Bedeutung gewinnt. Die Festlegung der Funktionsnamen und Parameter für FORTRAN folgt einer DIN-Empfehlung /2/. In Pascal wurde entsprechend wie in FORTRAN das Prozedur - Konzept gewählt, die Funktionsnamen sind gleich, die Parameter entsprechen sich weitgehend, wobei ausgenutzt

wurde, daß sich in Pascal die in der GKS-Norm gebrauchten Typen in der Sprache direkt definieren lassen. Dem Anwender werden die erforderlichenn TYPE- und PROCEDURE- Deklarationen auf Datei zur Verfügung gestellt, um sie in sein Programm einzukopieren. Da Standard-Pascal keine ARRAY's variabler Länge als PROCEDURE- Parameter zuläßt, ergeben sich Schwierigkeiten bei Koordinatenbereichen und Listen. Um sprachkonform zu bleiben, muß man a priori feste Längen definieren; bei PASCAL 300 wäre es möglich, statt der ARRAY's als Parameter Pointer auf ARRAY's zu benutzen.

Einbettung von GKS in die Systemumgebung

GKS-300 unterstützt folgende logischen Geräte (workstations):

- Segmentspeicher (WISS)
- Metafile Input
- Metafile Output
- Interaktiver grafischer Arbeitsplatz
  mit dem Ausgabegerät Speicherröhre Tektronix GMA
  und den logischen Eingabegeräten

Locator	realisiert	über	Tablett
Pick	"	"	Tablett
Valuator	"	"	Tablett
Choice 1	"	"	Tablett
Choice 2	"	"	alphanum. Sichtgerät
Text	"	"	alphanum. Sichtgerät

Um bei Bedarf ein Neuzeichnen des Bildes durchführen zu können, müssen alle Segmente abgespeichert werden. Da für die anfallenden Datenmengen der Hauptspeicherplatz nicht reichen würde, kommt nur Plattenspeicher in Frage. Andererseits ist für interaktives Arbeiten ein schneller Zugriff auf auf die grafischen Segmente erforderlich. Eine in der Praxis brauchbare Lösung besteht darin, für die Segmente im Hauptspeicher in Form einer verketteten Liste Informationsblöcke abzulegen, die Name, Attribute und einen Verweis enthalten, wo die grafischen Primitive auf Platte gespeichert sind (s.Abb.2). Plattenzuriffe sind abgesehen von Picken nur nötig, wenn neu gezeichnet wird. Allerdings ergibt sich eine Limitierung der Anzahl der Segmente durch den Hauptspeicherplatz, der zur Verfügung steht.

Da von der Hardware her die Pick-Operation nicht unterstützt wird, ist eine softwaremäßige Simulation notwendig. Dazu wird um die Position, die mit dem Pen eingegeben wird, ein kleiner Viewport gelegt. Durch Neuzeichnen aller pickbaren Segmente wird geprüft, ob Teile eines Segments in diesen Viewport fallen; ein solches Segment gilt als getroffen. Der Aufwand für das Picken wird dadurch wesentlich reduziert, daß in einer Vorselektion solche Segmente ausgeschieden werden, die als Kandidaten nicht in Frage kommen. Dazu werden während des Generierens eines Segments die maximalen und minimalen Koordinatenwerte bestimmt und in dem Segment-Informationsblock gespeichert. Als mögliche Treffer kommen

nur solche Segmente in Frage, deren so bestimmte Box mit dem Pick-Viewport einen nicht leeren Durchschnitt hat. Da keine Plattenzugriffe erforderlich sind, ist dieses "Boxing" /3/ sehr schnell.

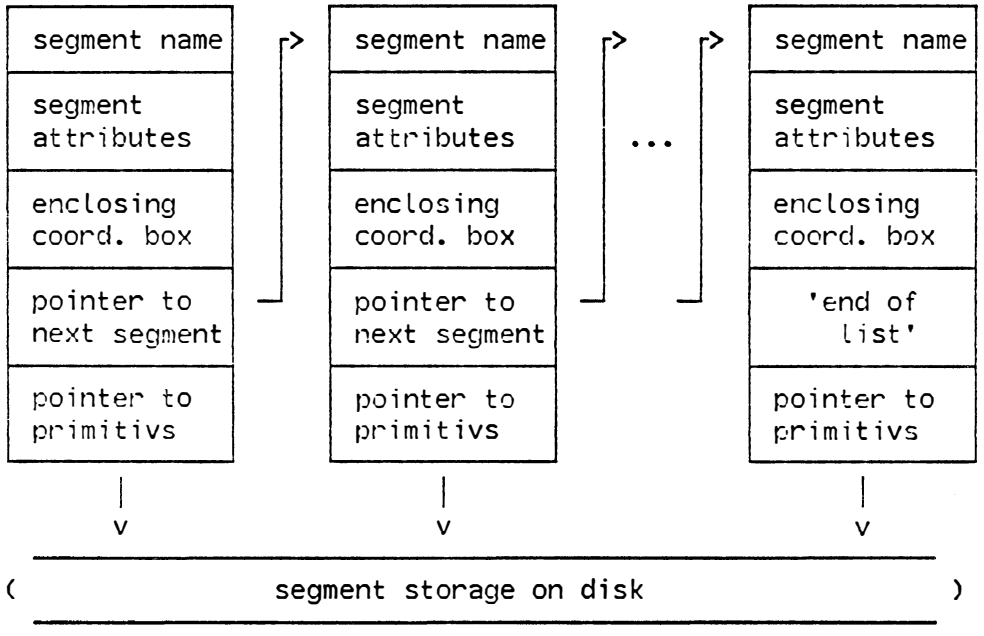

Abb. 2 Struktur des Segmentspeichers

GKS hat kein direktes Sprachmittel, um zwischen Storage Mode und Refresh Mode der Röhre zu unterscheiden. Für Anwendungen ist es jedoch sehr wichtig, den passenden Modus auswählen und ändern zu können. So werden solche Teile einer Grafik, die interaktiv bearbeitet werden, im Refresh dargestellt und nach erfolgreicher Beendigung in Storage Mode überführt. Da das wesentliche bei Refresh die Möglichkeit zum selektiven Löschen ist, ist es einleuchtend, Store und Refresh als Segmenteigenschaften zu behandeln. Da bei einer Speicherröhre das GKS Attribut "highlighted" keine direkte Bedeutung hat, wird es in GKS-300 benutzt, um Refresh zu kennzeichnen.

Von seiner Konzeption her übernimmt der Metafile zwei Aufgaben: Archivieren von Bildern an einer Anlage und Transport von Bildern zu fremden Anlagen. Zum Archivieren ist ein internes binäres Datenformat am geeignetsten, da keine Konvertierungen nötig sind; erst für den Transport ist ein genormtes ASCCI-Format erforderlich. Der Zentralmodul von GKS-300 kennt nur ein internes Metafile-Format; für die Konvertierung stehen Hilfsprogramme zur Verfügung. In GKS-300 gibt es keine Plotter Workstation; das Konzept sieht vor, daß Plotter über spezielle Spool-Programme mittels Metafile angesprochen werden.

Wichtigstes Ziel der Entwicklung von GKS-300 ist es, die Portabilität grafischer Anwendungsprogramme zu fördern; für die Akzeptanz

ist es wesentlich, daß das Laufzeitverhalten für interaktives Arbeiten annehmbar und das Gesamtsystem leicht zu handhaben und zu warten ist. An Hand von Testprogrammen zeigte sich, daß in den meisten Fällen der GKS Overhead vernachläßigbar ist gegenüber den zeitintensiven Hardwareoperationen wie Zeichengeschwindigkeit der Röhre, Datenrate des Tabletts und Platten - I/O. Bei grafischer Ausgabe, selbst mit Segmentstrukturierung, überwiegt deutlich der Zeitbedarf des Anwendungsprogramms; bei wechselnden Funktionen (z.B. LOCATOR und POLYLINE in einem "Montagsmaler"-Programm) kann sich OVERLAY-Wechsel unangenehm bemerkbar machen.

Um GKS-300 speziellen Benutzeranforderungen gemäß konfigurieren zu können, sind die dafür relevanten Daten (z.B. Länge des COMMON-Bereichs, Dimensionierung von Feldern, Tablett-Maße) als Pascal-CONST festgelegt und damit leicht änderbar. Vorbereitete Prozeduren zum Übersetzen, Binden und Laden sowie Dienstprogramme und Anwendungsbeispiele bieten dem Benutzer Unterstützung, um mit dem System ververtraut zu werden und es bequem handhaben zu können.

Literatur

/1/ DIN Deutsches Institut für Normung e.V.:
Informationsverarbeitung - Graphisches Kernsystem (GKS)
Funktionale Beschreibung, DIN ISO 7942 (in Englisch)
Beuth-Verlag, Berlin (1982)

/2/ DIN Deutsches Institut für Normung e.V.:
GKS FORTRAN- und DI/DD-Interface, NI-5.9 / 34-80

/3/ W.M. Newman, R.F. Sproull:
Principles of Interactive Computer Graphics
McGraw-Hill, New York (1979)

/4/ SIGRIS, Technische Beschreibung, ORG-PV Version
Siemens AG, München (1981)

/5/ G. Schmitgen: GKS-300 - eine GKS-Implementierung auf dem grafischen Arbeitsplatz SIEMENS R30-SIGRIS
IIG-INFO, GMD, St. Augustin (1981)

/6/ H. Jungblut, K.-H. Klein, G. Schmitgen:
GKS-300 - GMD-Implementierung des grafischen Kernsystems GKS auf SIEMENS Arbeitsplatzrechnern
Übersicht und Benutzeranleitung
Arbeitsberichte der GMD Nr.12, St. Augustin (1983)