GC33-0100-0

Graphical Data Display Manager (GDDM) and Presentation Graphics Feature (PGF)

Program Product

General Information

Program Number 5748-XXH

IBM

GC33-0100-0

Graphical Data Display Manager (GDDM) and Presentation Graphics Feature (PGF)

Program Product

General Information

Program Number 5748-XXH

IBM

First Edition (September 1979)

| This edition, as amended by Technical Newsletter (TNL) SN33-6264,
| applies to Release 1.0 of the IBM Program Product: Graphical Data Display Manager (GDDM), Program Number 5748-XXH, and the Presentation Graphics Feature (PGF), Feature Number 6047 or 6048.

Information in this publication is subject to change. Changes will be published in new editions or technical newsletters. Before using this publication, consult the latest IBM System/370 and 4300 Processors Bibliography, GC20-0001, to learn which editions and technical newsletters are current and applicable.

It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

Publications are not stocked at the addresses given below; requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication; if the form has been removed, comments may be addressed either to:

International Business Machines Corporation,
Department 812HP,
1133 Westchester Avenue,
White Plains, New York 10604.

or to:

IBM United Kingdom Laboratories Limited,
Programming Publications, Mail Point 095,
Hursley Park,
Winchester, Hampshire SO21 2JN, England.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Preface

This book provides introductory information on an IBM program product which can add a versatile, easily-used pictorial display capability to existing IBM data-processing systems. The material is intended for executive and data-processing management, and application programmers. Chapter 1 discusses the advantages of graphical data display, provides a summary of product capabilities and typical applications, and illustrates some sample displays. The operating environment (operating systems and subsystems, programming languages, etc.) is described in Chapter 2, and the last chapter provides the application programmer with information on product use.

Contents

Figures

Chapter 1. Introduction

The use of numbers to convey information pervades modern life. This mode of communication is well suited to machines, but is often found wanting by people. Few of us can easily extract and fully comprehend the information content of purely numerical data. The task becomes much more difficult when we try to deal with large masses of numbers, or seek meaningful relationships among several sets of data.

Pictorial representation, one of the oldest forms of human expression, can provide an effective alternative. Unlike words and numbers, pictures are a universal language, and perhaps the most natural way for people to communicate. They can convey information directly, by stimulating the brain's unique abilities to distinguish shapes, patterns, colors, and shadings. Converting information to pictures can therefore assist the human use of machine-processed data.

Some of the inherent advantages of pictorial data presentation are shown in Figures 1-1 through 1-4. In the first example (Figures 1-1 and 1-2), note how the use of shading, color, and bar length creates visual patterns which clearly show the relationships in the data. Figures 1-3 and 1-4 illustrate the use of pictorial display to present the results of data analysis in an easily-assimilated form.

WEEK	DEPT A	DEPT B	DEPT C	TOTAL
1	17,103	6,227	2,696	26,026
2	2,889	3,210	3,851	9,950
3	963	8,282	4,815	14,060
4	13,996	14,124	3,916	32,036
5	17,912	14,065	1,990	33,967
6	12,904	1,284	899	15,087
7	14,894	14,058	4,045	32,997
8	14,766	8,346	2,054	25,166
9	19,902	8,089	3,978	31,969
10	17,912	5,265	3,786	26,963
11	4,622	15,410	5,008	25,040
12	6,805	8,282	2,119	17,206
13	7,832	4,173	2,055	14,060
14	10,914	2,311	4,879	18,104
15	11,877	15,023	3,916	30,816
16	16,820	2,440	3,788	23,048
17	8,988	8,217	3,981	21,186
18	16,949	6,099	1,027	24,075
19	5,842	4,943	4,045	14,830
20	9,052	14,188	4,815	28,055
21	10,850	8,206	2,086	21,142
22	9,951	7,190	3,082	20,223
23	5,906	15,248	3,820	24,974
23	3,853	3,017	1,284	8,154
25	3,916	12,391	1,862	18,169
26	11,745	12,242	1,926	25,913

Figure 1-1. Data for Figures 1-2, 1-6 and 1-8

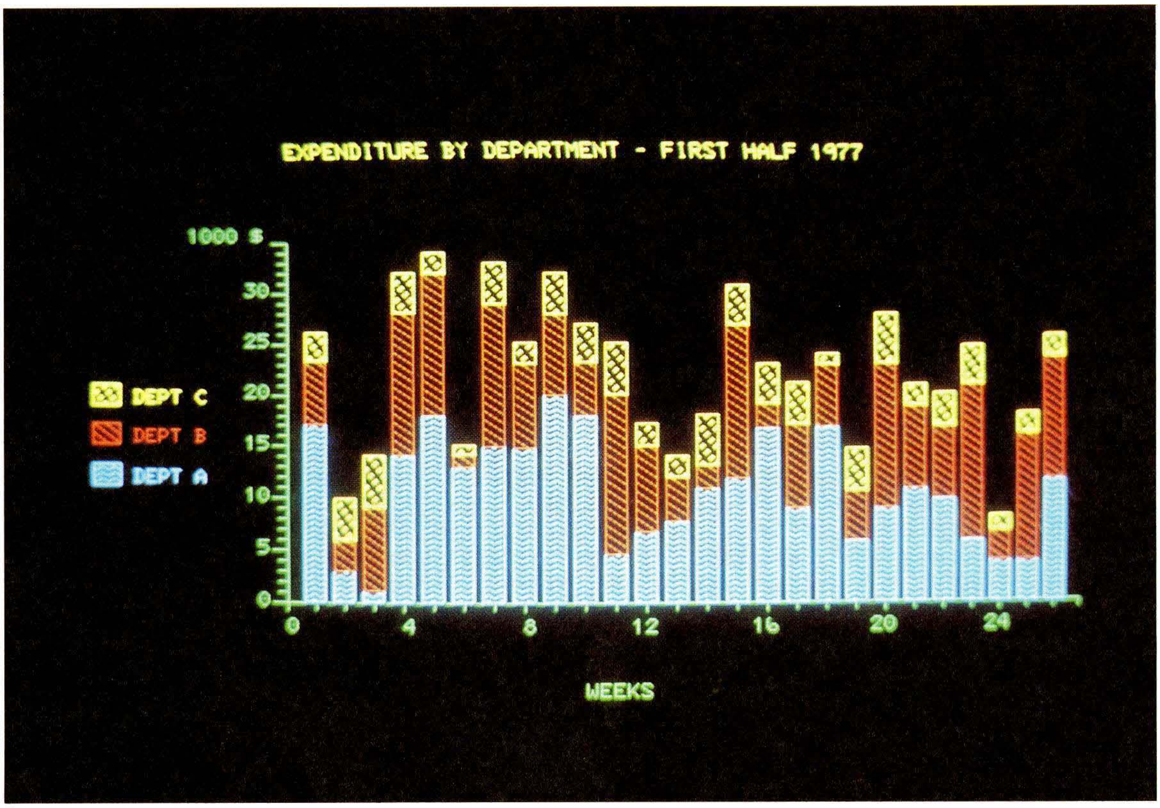

Figure 1-2. Composite Bar Chart

45	75	107	105	110	116	120
120	129	124	130	143	145	134
144	147	139	158	153	150	168
165	161	151	171	177	183	142
189	193	223	209	225	200	197
190	184	178	172	153	162	165
167	150	153	158	149	147	144
134	141	139	148	131	124	122
113	121	116	111	109	108	95
73	56	67	71	79	85	87
83	95	109	91	112	119	126
118	122	113	131	148	139	141
134	144	140	138	157	152	156
164	167	163	154	173	179	180
185	199	191	204	208	186	180
174	155	163	167	164	159	140
152	156	144	134	148	139	149
132	122	113	125	116	112	88
97	102	96	114	119	129	119
125	113	132	145	139	148	134
147	140	155	152	169	164	167
163	165	175	181	187	188	182
176	163	167	164	169	151	154
140	147	144	148	142	145	132
115	125	125	128	120	115	94
98	91	93	100	92	109	115
120	129	125	126	103	94	115
133	145	142	148	144	147	140
91	103	94	96			

Figure 1-3. Data for Figure 1-4

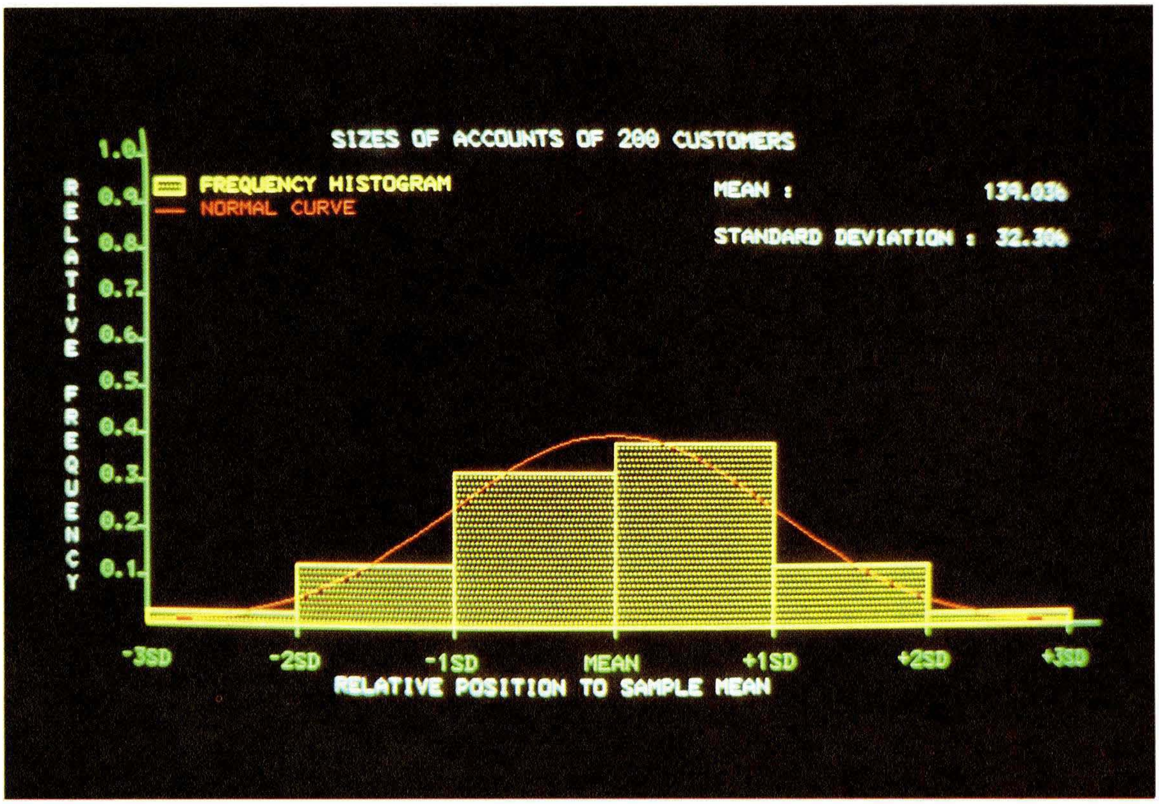

Figure 1-4. Statistical Analysis Chart

This manual describes the program product named Graphical Data Display Manager (GDDM) and the optional Presentation Graphics Feature (PGF). They are designed to offer a versatile, easily-used pictorial display capability to a wide range of users. By exploiting certain characteristics of display terminals and printers, information can be displayed in alphanumeric or graphical form, or in both forms simultaneously.

GDDM forms the foundation for graphics, by providing the functions needed to create pictorial displays compatible with the equipment available to the user. These functions are implemented either by a user-supplied application program or by PGF.

PGF can significantly reduce the programming effort needed to generate commonly-used graphs and charts. It can be used in two ways:

- In conjunction with a simple user application program
- Interactively, by the terminal user, with no need for an application program

While the interactive mode does not provide all of the features and flexibility of GDDM and PGF, it is especially useful because it requires only simple keyboard entries and can be used by non-technical personnel. This enables a user to begin basic graphics operations immediately upon installation of GDDM and PGF, without waiting for the development of application programs.

GDDM and PGF are compatible with a number of host operating systems and subsystems and several programming languages. Specific information on these is provided in Chapter 2.

Summary of Product Capabilities

GDDM and PGF contain many functions used for graphical display. These will be summarized in this section and discussed in more detail in Chapter 3.

The functions of both can be grouped into two general classes:

- Utilities, which can be used independently of application programs
- Basic functions, which are used only by an application program or a utility

UTILITIES

The Interactive Symbol Editor Utility is supplied with GDDM. Two others are provided with PGF: the Interactive Chart Utility and the Interactive Vector Symbol Utility.

Interactive Symbol Editor Utility (GDDM)

The Interactive Symbol Editor Utility enables the terminal operator to create and edit Image Symbol Sets (ISS). These can be used by GDDM as alphanumeric characters, graphical marker symbols, or shading patterns. Symbols may be monochrome or multicolored.

The utility can generate any symbol which can be depicted by a dot matrix of maximum dimensions 255 by 255 dots; a typical character matrix size is 9 by 16. All symbols in a set must have the same matrix dimensions. Symbol sets whose matrix size matches that of the display device using them may be loaded directly into the device by GDDM. Such symbols are called Programmed Symbols (PS). Larger symbols cannot be used as PS, but can be loaded into GDDM storage by the application program.

Each ISS created with the utility can contain up to 190 symbols. Sets are assigned names by the user and stored on files for later use. GDDM has functions for reading a set from the file and loading it into the display device or GDDM storage, as applicable.

Some examples of user-defined symbols are:

- Language symbols (Cyrillic, Farsi, Gothic, Greek, Kanji, etc.)
- Mathematical or other special symbols (exponents, summation or integral signs, chemical symbols, etc.)
- Footnote or reference marks, subscripts, superscripts, etc.
- Company logos

GDDM contains the following image symbol sets:

- One set of marker symbols
- One set of geometric shading patterns

These sets cannot be modified by the user.

In addition, the following symbol sets are provided with GDDM:

- A PS character set of standard keyboard symbols in three sizes, suitable for the IBM 3278 and 3279 Display Stations and the IBM 3287 Printer.
- An italic version of the above, in the same sizes.
- A set of 64 geometric shading patterns.
- A set of 64 color shading patterns

These can be edited by the user if desired, in the same way as sets created by using the utility.

Interactive Chart Utility (PGF)

The Interactive Chart Utility simplifies the production of certain commonly-used graphs and charts. It can be used independently, or by a call from an external program, which provides an easy way to add graphics functions to existing programs. In either case, it enables the user to create the following chart types:

- Line graph
- Surface chart
- Bar chart
- Histogram
- Pie chart
- Venn diagram

The presentation of a chart on the display screen involves three types of operations:

1. Chart format definition

   This specifies the type of chart, titles, labels, layout, and various chart attributes such as line type, color, shading, etc., which govern the appearance of the chart. Possible actions are:

   - Specify a new definition
   - Retrieve a definition from a file
   - Modify a definition (change titles, labels, attributes, etc.)
   - Save a definition on a file

2. Chart data specification

   - Keyboard entry from the terminal (prompted by a menu)
   - Retrieve data from a file
   - Modify data (change, insert, delete)
   - Save data on a file

3. Chart presentation

   - Display on the terminal

   - Redisplay after modification

   - Print

The utility communicates with the terminal user by displaying a sequence of <u>menus</u>, which contain prompting information and entry fields. Where applicable, default values are shown on the menus. The user responds by:

- Entering or changing the information in the entry fields

- Using keyboard program function (PF) keys to initiate actions. The actions related to these keys are displayed at the bottom of the menu, as applicable.

The menus are largely self-explanatory, so the user does not need detailed knowledge of the PGF functions used by the utility. Additional advisory information can be displayed on the terminal, if needed.

The simplest use of the utility involves only a basic chart format definition (type, titles, and labels) and data entry. This will produce a chart using the default values for chart attributes. If the chart uses more than one set of data values, the appropriate attributes (line type, color, shading, etc.) are changed automatically to distinguish the data on the chart.

Alternatively, the user can expand the format definition by specifying attribute values instead of accepting the defaults. Thus the chart appearance can be tailored as desired.

After the chart has been defined and the data entered, the resulting chart can be displayed on the terminal. The user can then use the modification options to change the chart appearance (or even select a different type), or modify the data, then redisplay the revised chart. This interactive process can continue until the user is satisfied with the result.

The save and retrieve functions can be used to minimize user input. For example, several chart formats can be predesigned and saved on external files. Subsequently, an application program can invoke the utility and pass data by a call statement, and the terminal user can retrieve a specific chart format. The data and/or format can be examined, and desired revisions can be made either before or after the chart is drawn. At the user's option, the revised data can be passed back to the application, or the revised data and/or format can be saved for later retrieval.

The Chart Utility has a few restrictions:

- The number of options available and the length of specific entry fields are limited.

- The terminal operator does not have direct control over the graphics display area.

- Only one chart with one set of axes may be displayed at a time.

In many cases, these restrictions will be unimportant, and user requirements will be fully satisfied by the Chart Utility.

Interactive Vector Symbol Utility (PGF)

The Interactive Vector Symbol Utility complements the Symbol Editor Utility by providing the capability to create and edit Vector Symbol Sets (VSS). Vector symbols differ from image symbols in the following ways:

- Vector symbols are defined by straight lines drawn from point to point in a rectangular space.
- Vector symbols can be used only by calls to GDDM from an application program or from PGF. They cannot be loaded directly into the display device as PS.
- The size and angular position of vector symbols can be controlled when used, by GDDM calls. Figure 3-1 shows some examples of such use of vector symbols.

Vector symbol sets generated by the utility can be assigned names by the user and stored on files for later use by GDDM or PGF. A standard vector symbol set is supplied with PGF.

BASIC FUNCTIONS

Both GDDM and PGF contain functions which are used to construct a picture. They are accessible by calls from application programs or invoked by the utilities as needed. These functions are described in some detail in Chapter 3.

GDDM

The basic functions of GDDM may be summarized as follows:

- Screen format control
- Alphanumeric input and display
- Graphics construction and display (lines, arcs, text, shaded areas, etc.)
- Control of attributes (color, line type, line width, marker symbols, shading patterns, etc.)
- Removal of parts of a picture lying outside a specified boundary
- Printing control

These functions allow the application programmer considerable flexibility in creating pictorial displays. For example, they can be used to design specialized charts, diagrams, maps, forms, etc.

The "basic" functions of PGF are actually a higher-level means of accessing GDDM in order to construct specific types of data charts. They are named Presentation Graphics Routines and are accessible by calls from an application program, which can be much less complex than one employing GDDM functions directly.

The Interactive Chart Utility also uses these routines. The primary advantage in using the Presentation Graphics Routines directly via an application program is the ability to create more sophisticated or more complex displays. For example, one could display several charts at different locations on the screen, or overlay two or more charts on the same set of axes.

The following chart types can be produced by the Presentation Graphics Routines:

- Line graph (with or without data point markers)
- "Scatter" plot (data markers only, no lines)
- Surface chart
- Histogram
- Bar chart (composite, multiple or floating)
- Pie chart (percentage or absolute values)
- Venn diagram

Each chart or graph may contain several sets of data (components) and these can be distinguished in various ways appropriate to the chart type. Functions are available for specifying:

- Position and size of the chart on the screen
- Chart title and its position
- Axes (linear or logarithmic, ranges, intercepts, scale marks, labels, titles)
- Datum lines (parallel to either axis, or to both)
- Grid lines
- Key symbols to identify chart components
- Attributes (line types, marker symbols, color, shading patterns, character set, etc.)

Typical Applications

Because pictorial display can so effectively enhance human perception, its range of application is almost limitless. Many requirements can be satisfied by the standard types of charts such as those produced by the Interactive Chart Utility. Others may require more complex displays designed by the user with GDDM. Some typical applications are listed below.

Planning/Analysis/Tracking
- Time trend analysis
- Data comparison
- Forecasting
- Sales analysis
- Modeling/simulation
- Critical path analysis
- Status reporting

Scheduling
- Transportation/distribution logistics
- Personnel/equipment schedules

Forms Creation/Completion
- Orders
- Payrolls
- Billing
- Reports

Industrial Process Monitoring/Control
- Network/pipeline maps
- On-line display of variables

Medical
- Cardiograms, scans, etc.
- Analysis of test/experimental data

Graphic Arts/Publishing
- Illustrations, diagrams
- Page layout and composition

Data Processing Installations
- Flow charts and diagrams
- Program documentation

Education
- Visual aids, programmed instruction

Illustrations

The following figures have been chosen to illustrate displays which might be relevant to some of the applications listed above. All illustrations were generated by the products described and printed from photographs of the display.

(See also Figure 1-2)

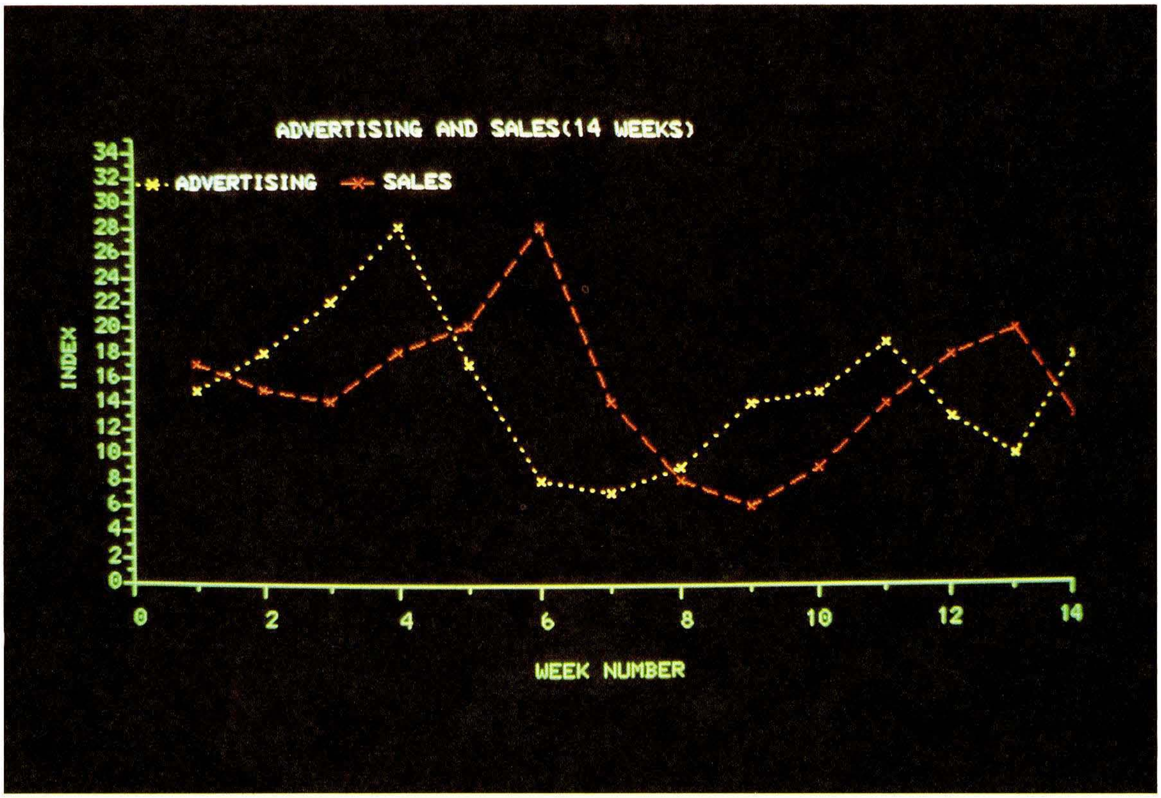

Figure 1-5. Line Graph

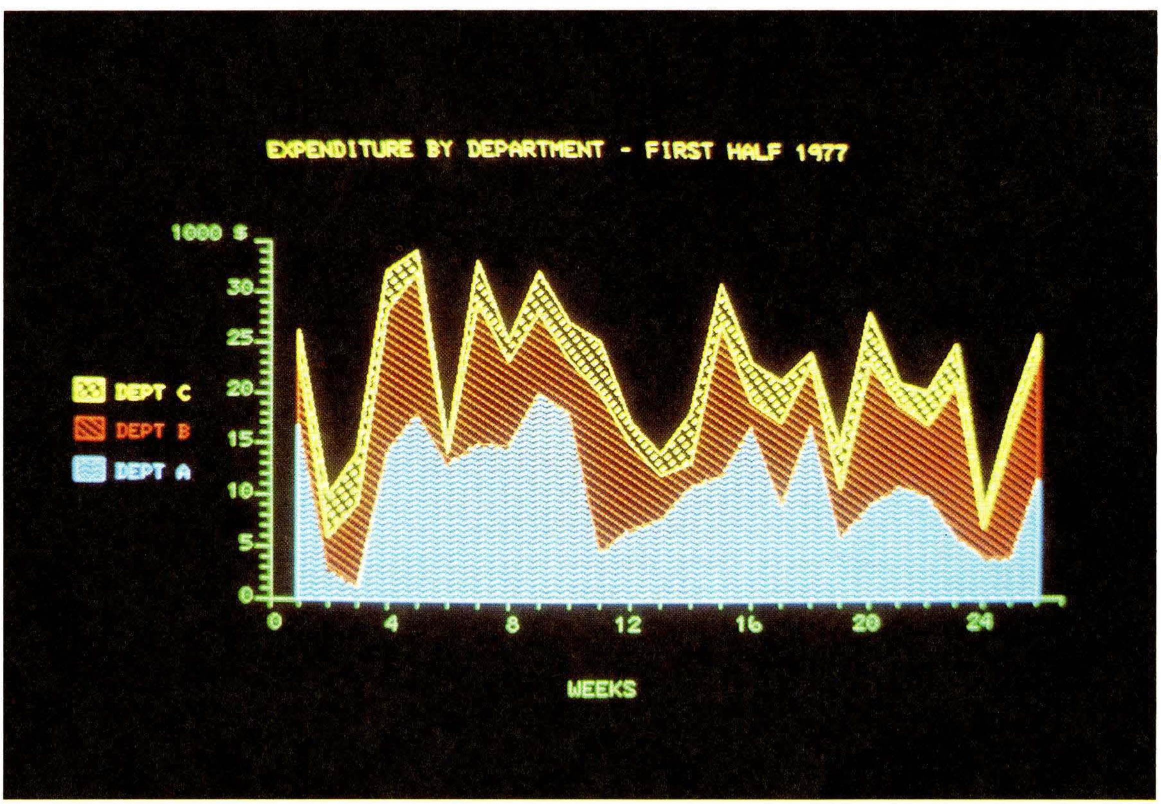

Figure 1-6. Surface Chart

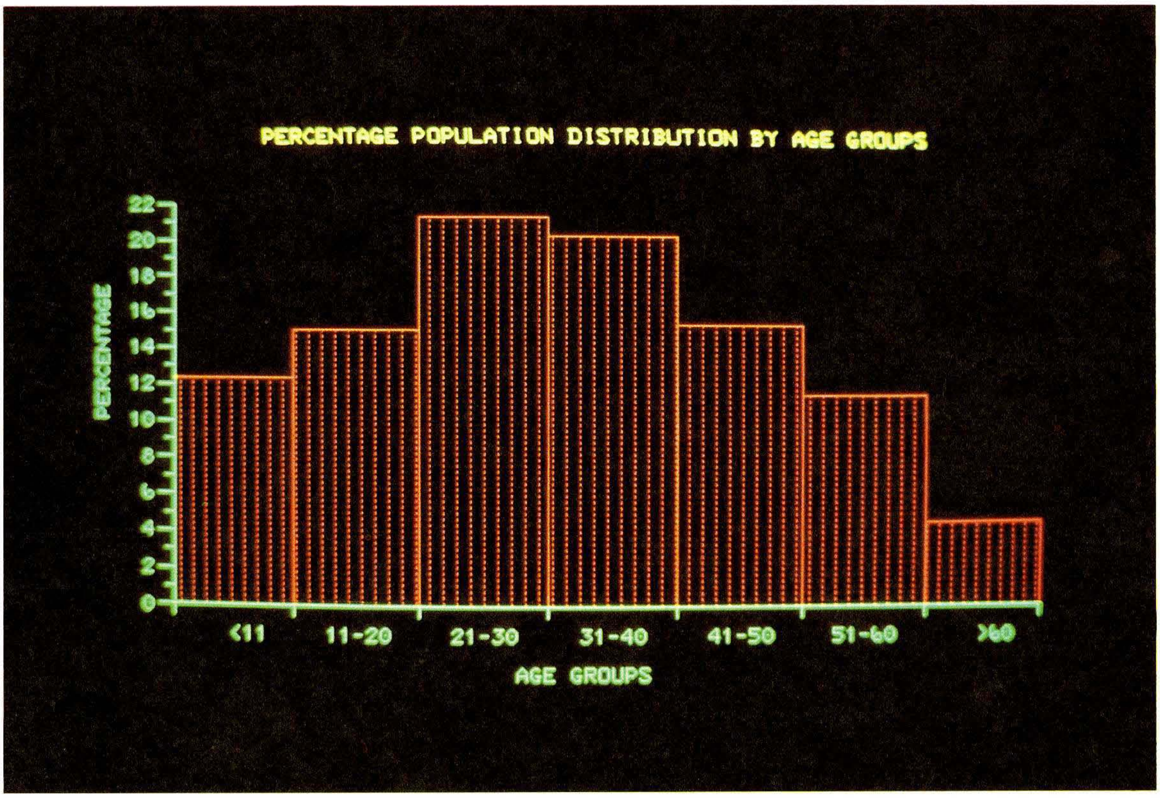

Figure 1-7. Histogram

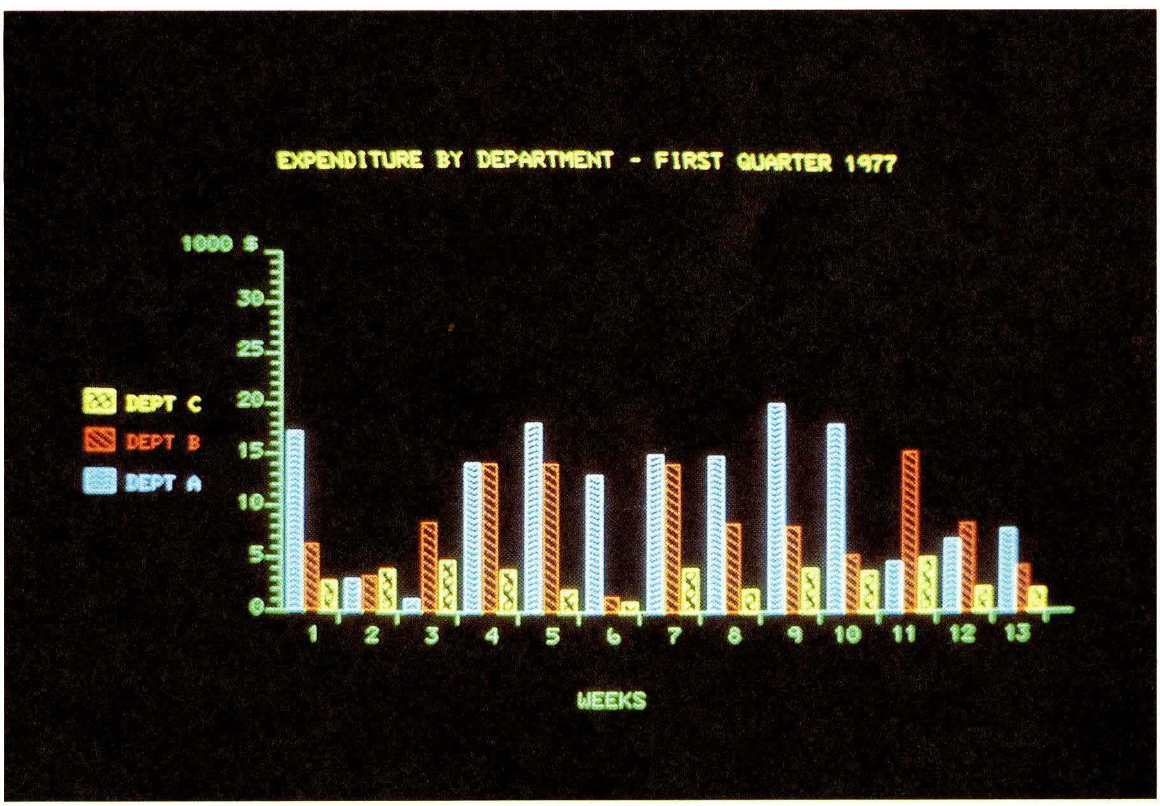

Figure 1-8. Multiple Bar Chart

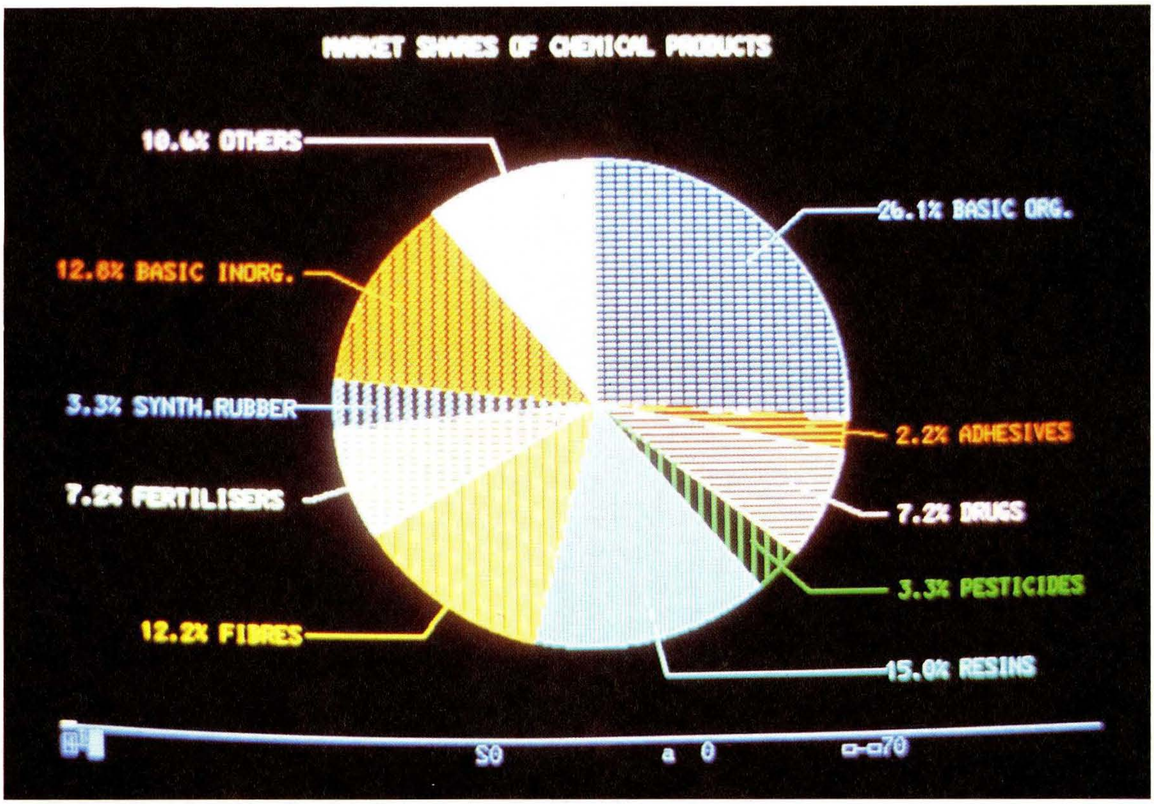

Figure 1-9. Pie Chart

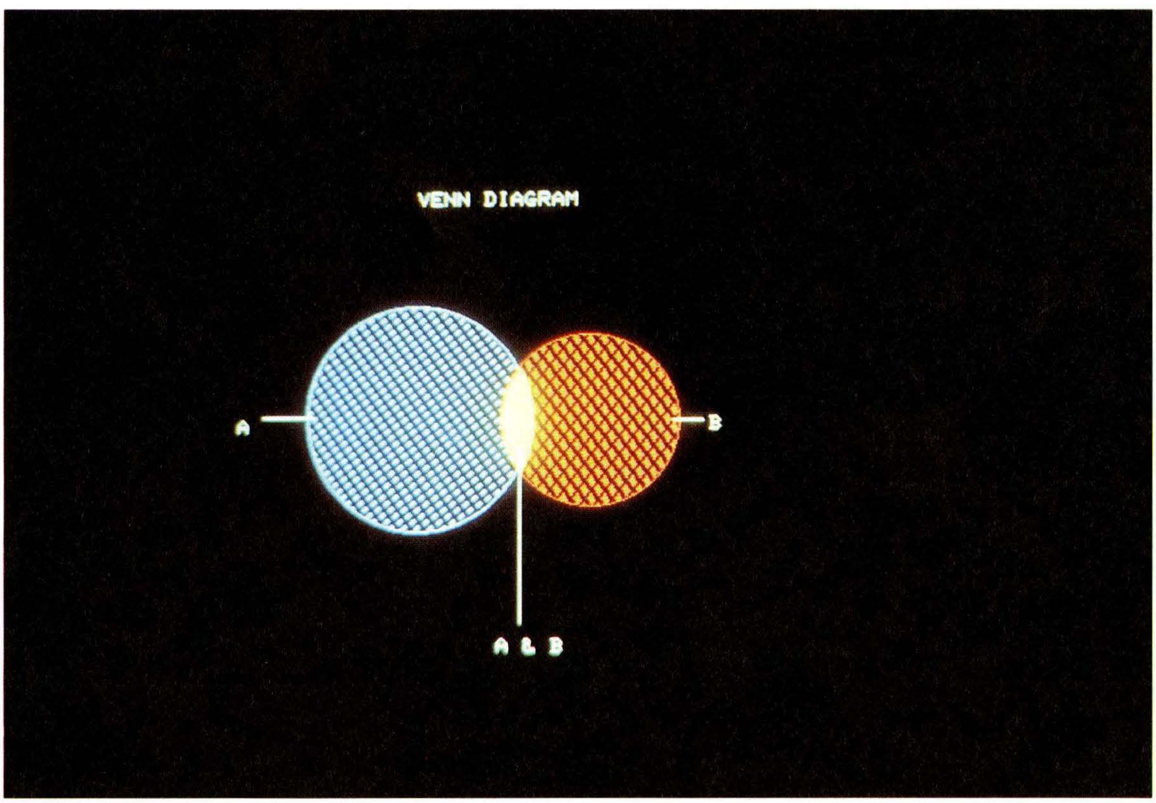

Figure 1-10. Venn Diagram

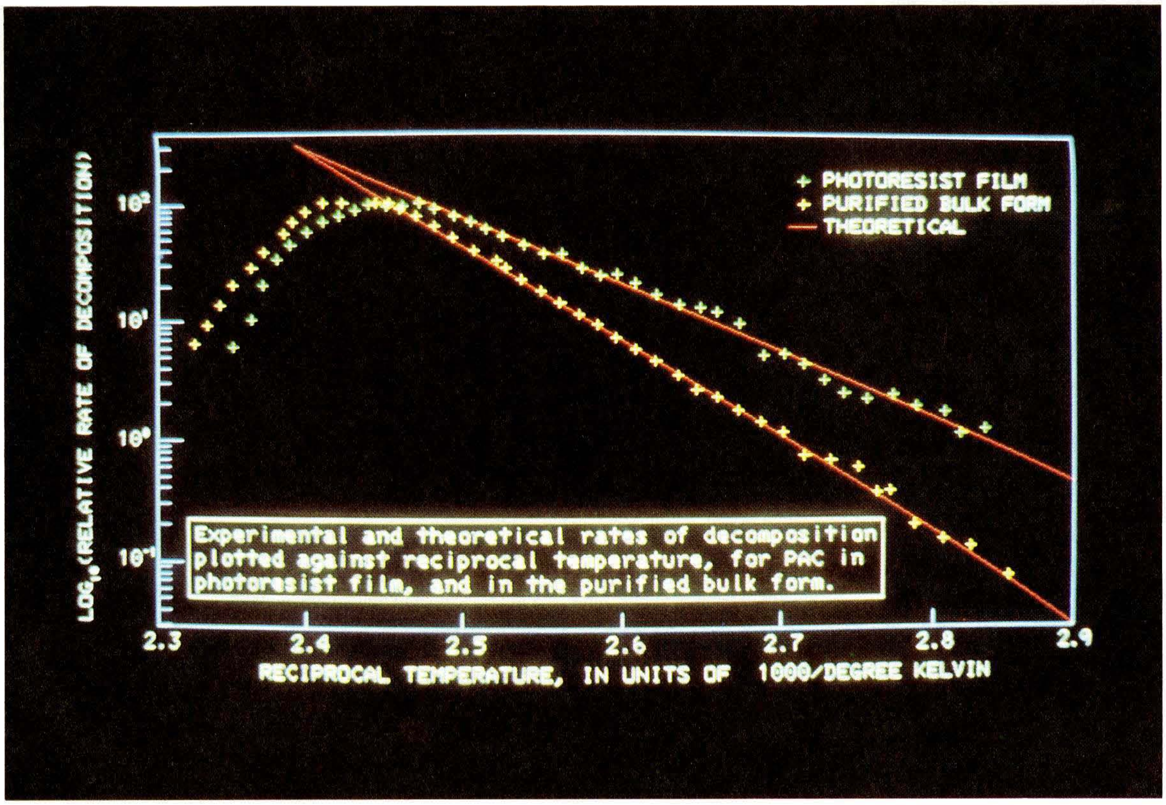

Figure 1-11. Technical Data Chart

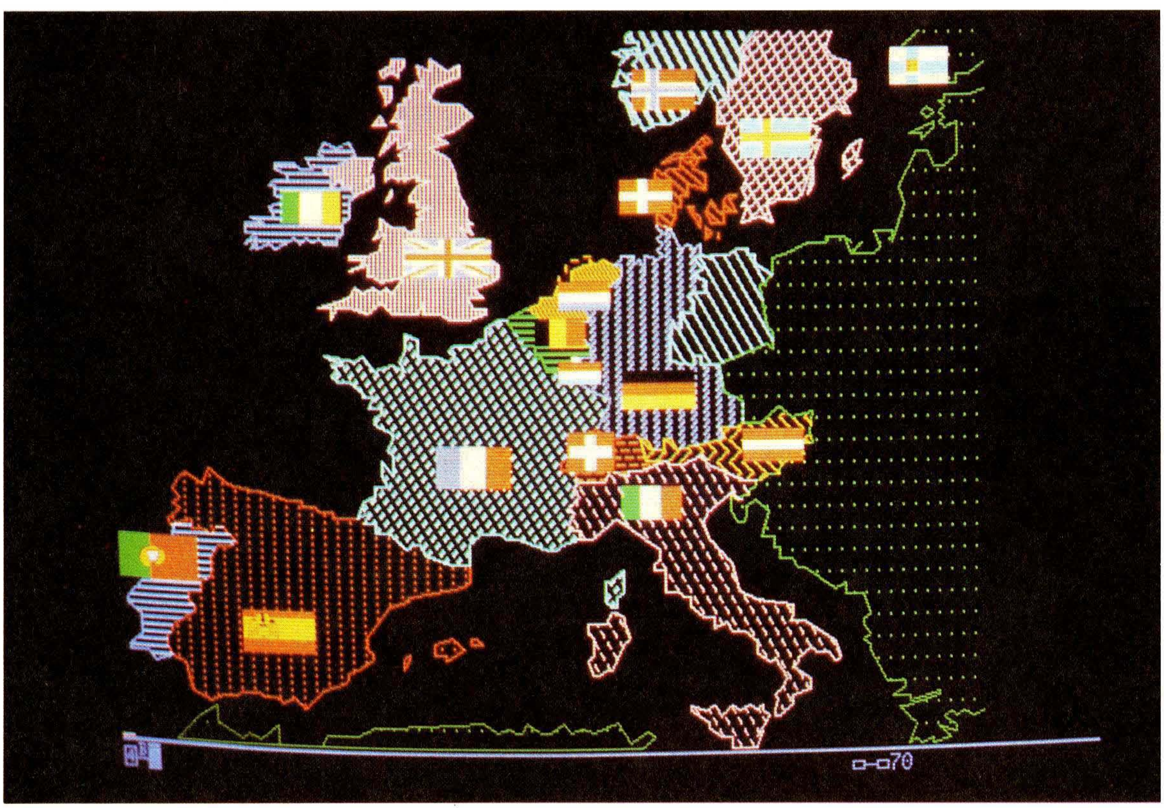

Figure 1-12. Map of Western Europe

Chapter 2. The Operating Environment

This chapter describes system hardware and software which will support GDDM and PGF. It also discusses coexistence and compatibility, storage requirements, performance, and security.

Processing Units

GDDM and PGF will run on any IBM System/370 or 4300 series processing unit with sufficient storage to meet the combined requirements of:

- The host operating system
- Access method
- DB/DC or interactive subsystem
- GDDM and PGF
- The user application program

Both GDDM and PGF require the floating point feature.

Operating Systems and Subsystems

The following table summarizes the IBM operating systems, subsystems, and access methods which will support GDDM and PGF:

Access Methods	DOS/VSE R36	OS/VS1 R6.7	OS/VS2 R3.7	VM/370
ACF/VTAME	CICS/VS V1 R5.0	–	–	–
ACF/VTAM V1 R2	CICS/VS V1 R5.0	CICS/VS V1 R5.0	CICS/VS V1 R5.0 TSO/VTAM	–
| ACF/TCAM V2 R1* | V2 R2	–	CICS/VS V1 R5.0	CICS/VS V1 R5.0 TSO/TCAM**	–
BTAM	–	CICS/VS V1 R5.0	CICS/VS V1 R5.0	–
BTAM-ES	CICS/VS V1 R5.0	–	–	–
VM/CMS	–	–	–	VM/CMS

| *Display support only
| **TSO/TCAM requires OS/VS2 R3.8

The levels of software listed in the table are required to allow GDDM to transmit the necessary extended data streams to local and remote terminals and printers.

Terminals, Printers, and Controllers

GDDM and PGF will provide graphics and alphanumerics on the following display terminals and printers:

- IBM 3278 Display - Models 2, 3 and 4 (monochrome)
- IBM 3279 Display - Models 2b and 3b (color)
- IBM 3287 Printer - Models 1 and 2 (monochrome)

  \- Models 1c and 2c (color)

These devices must be fitted with the Programmed Symbol (PS) feature and the Extended Character Set Adapter (ECSA). The IBM 3287 Printer also requires the Graphic Print feature.

The terminals and printers listed above must be attached to an IBM 3274 Controller which has the capability to support PS and color.

GDDM also supports full-screen alphanumeric operation on all models of the IBM 3270 series of display terminals. However, extended highlighting, color, and PS usage are not available on all of these devices.

The GDDM and PGF functions are also available to users of the above terminals (and the IBM 8775 Display) in conjunction with an IBM 8100 Processing Unit, via the 3270 Data Stream compatibility of DPPX or DPCX.

Programming Languages and Compilers

GDDM and PGF will support application programs using the following languages and compilers:

- System/370 Assembler Language
- PL/I (DOS and OS Optimizing Compilers and Libraries)
- COBOL (DOS/VS, OS/VS, ANS V2, V3, V4 Compilers and Libraries)
- FORTRAN (G and H Compilers)

CICS/VS cannot accept application programs written in FORTRAN.

Installation

Because GDDM and PGF run as subroutines to application programs and the utilities are stand-alone applications using the subroutines, no special system generation is required.

Installation is quite simple. A few data sets must be allocated and initialized:

- A data set containing PL/I Include members for subroutine declaration (PL/I only)
- A data set containing subroutine object code. This library is used during application link-edit.

- A data set containing GDDM and PGF load modules. This library contains shareable code loaded during application execution.
- Partitioned data set (TSO), set of sequential files (CMS), or VSAM key-sequenced data set (CICS), containing supplied symbol sets.

On CICS/VS, members of the load library must be added to the Program Processing Table, the PS data sets must be added to the File Control Table, and the Interactive Utilities (Symbol Editor, Vector Symbol and Chart) must be given transaction codes.

Coexistence/Compatibility

GDDM contains a function for restoring the previously-displayed picture to the screen. This allows GDDM to coexist with other full-screen processors:

CICS/VS BMS
: After GDDM has formatted the screen and displayed data, the picture can be replaced by a BMS-generated display. The terminal user then enters data using BMS; when the BMS operations are completed, the original GDDM display can be recalled or replaced by another.

TSO Line Output
: Line-by-line output generated by the application with TSO clears the screen and output appears from the top down. The GDDM picture can be restored or replaced by a new one.

CMS Line Output
: Line-by-line output on CMS clears the screen and output appears from the top down. The previous GDDM display may be restored by operator intervention or by an application call statement.

PRPQ PO9013
: GDDM and PGF may be used in conjunction with most of the Application Support routines from the IBM 3277 Graphics Attachment Support program (Program Number 5799-AXX PRPQ PO9013). In particular, the subroutines for analytic geometry, three-dimensional transformations, polygonal clipping, and curve fitting may be used. All functions are usable <u>except</u> those specifically concerned with the Dual Screen Manager and the PRPQ device support. The migration of applications written for the PRPQ to GDDM may require structural changes to the application code.

Storage Requirements

The main storage requirements for GDDM and PGF depend strongly on the complexity of the data being displayed, and the functions being used. The code is reentrant, so a single copy may be shared by a number of concurrent applications.

The following estimates of main virtual storage requirements, in bytes, are provided for general guidance. Note that modules are loaded only when required for execution.

	System/370 object code	+ per environment (11)	+ per application	+ per terminal (9)
GDDM	203K (1)	20K (8)	4K	10-43K
PGF	116K (2)	-	4K	8K (2)
Symbol Editor	79K (2,4)	-	-	15-24K
Vector Symbol Editor	79K (2,5)	-	-	129K
Chart Utility	80K (2,3,6)	-	-	72K
Print Utility	47K (7)	20K (8)	-	84K (10)

Notes: (1) Base services 110K. Alphanumerics 19K. Graphics 74K.
(2) Plus GDDM.
(3) Plus PGF.
(4) Plus 38K (shareable) for help panel, if required.
(5) Plus 114K (shareable) for help panel, if required.
(6) Plus 122K (shareable) for help panel, if required.
(7) Plus 160K (shareable) of GDDM services.
(8) 26K for TSO.
(9) Depending on the complexity of the process being performed. Minimum and typical values have been quoted.
(10) On TSO, this is per printer.
(11) CICS/VS, VM/CMS, or TSO.

"Shareable" applies to multiple users of the relevant help facility.

AUXILIARY STORAGE

Auxiliary (DASD) storage requirements for GDDM and PGF object code are 904K bytes for a single operating environment (maximum 1029K for all environments), plus up to 12K bytes per symbol set.

Performance

From the viewpoint of the terminal user, the significant performance criterion is the time required after data entry to begin and complete a picture on the screen. This is dependent on many factors, for example:

- Type of host processing unit
- Type and number of terminals in use
- Line speeds and utilization
- Channel utilization
- System activity and priority allocations
- Graphics picture complexity
- Application program design

Because of these dependencies, specific response times cannot be stated here. Several charts of different complexity have been selected as representative benchmarks. The lengths of the corresponding compressed data streams needed to construct them have been estimated, and are in the range of approximately 3000 to 7000 bytes. GDDM constructs these data streams so that the picture appears as successive horizontal sections on the display terminal. Figure 1-6 is an example of a chart at the upper end of this range, and Figure 1-8 represents the lower end.

Security/Auditability

GDDM and PGF do not provide any security features beyond those available in the subsystem being used.

Tables generated by the products are normally held in pageable main storage, obtained on behalf of the application by the subsystem. They will therefore be protected from accidental or deliberate interference by other users to the extent provided by the subsystem.

The application program can specify a user routine to receive control following each call to a GDDM function. The routine receives the function code of the GDDM function called. Return from the user routine transfers control to the program statement following the one which called the GDDM function. This could be used by the application to provide a trace of GDDM/PGF operations.

Reliability, Availability, Serviceability (RAS)

The RAS characteristics of GDDM/PGF are heavily dependent on those of the supporting system or subsystem. RAS features provided by the host environment can be used.

The application programmer will have access to system-provided application services such as:

- Journaling
- Tracing
- Abnormal condition interception

GDDM contains the following features to augment RAS characteristics:

1. The detection of error conditions in GDDM/PGF or user-written code, and the recognition of error conditions returned by the system or subsystem services. In either case, if corrective action is not possible, an appropriate error message common to all operating environments is sent to the terminal. The application program can specify an error exit to a user routine, which receives the error record.

2. A trace function to permit the application to initiate internal tracing. The function argument specifies the tracing level to be used.

The application programmer has responsibility for using these features as required by the application. For example, the error exit function can be used for application tracing, and the trace function for diagnosis of possible GDDM/PGF internal errors.

Chapter 3. Application Programming

For the application programmer, ease of use is an important feature of both GDDM and PGF. The functions needed to produce alphanumerics and/or graphics displays are available by simple call statements from application programs. The three interactive utilities can be used in stand-alone mode, and the Interactive Chart Utility in PGF can also be invoked by an application program.

Using GDDM

GDDM is the lowest level of interface to the display screen or printer which is available to the application by call statements. It operates the terminal (see Chapter 2) in full-screen mode, so the application has complete control of the display. Provision is made, however, for the interruption of this control by line-by-line or full-screen output from other sources. Following such interruption, the previous screen display may be restored. GDDM provides five broad classes of basic functions, plus the Interactive Symbol Editor Utility.

BASIC FUNCTIONS

Format and Control Operations

The screen or printer is considered as displaying a page of information containing one or more alphanumeric fields and/or one graphics field. The latter is a rectangular area on a page in which pictures are constructed. Alphanumeric fields may overlap the graphics field; in such cases the alphanumeric field prevails.

A page has height and width, defined in a row-column coordinate system starting at (1,1) in the top left corner. The maximum (and default) page size is equal to the size of the display screen. Page size is set when a page is created by a call from the application.

Only one page can be displayed at a time, but other previously-defined pages may be selected for display at any time.

Call statements are available to perform the following page operations:

- Create (assign identification number, define the size)
- Select
- Delete
- Clear
- Query page information
- Save on file

- Read from file and display

Other call statements provide:

- Control functions (initialization, termination, error exit and/or trace specification)
- Device operations (screen update or restore, reading, etc.)

Symbol Set Operations

In addition to the character definitions built into the display devices, GDDM can use two types of symbol sets:

1. Image Symbol Sets (ISS)

   ISS can be created by the Interactive Symbol Editor Utility, and they can take two forms:

   a. Low-quality symbols. These are designed for loading by GDDM directly into the display device internal storage (Programmed Symbols(PS)). They are used in graphics/alphanumerics in exactly the same way as the symbol definitions built into the device. Example: a user-designed set of special alphanumeric symbols.

   b. Medium-quality symbols. ISS of this form are not loadable as PS, but are read into GDDM storage for use in graphics. The symbols are of fixed size, defined when created, but their spacing can be controlled when used. Typical examples are large symbols, marker symbols, and shading patterns.

2. Vector Symbol Sets (VSS)

   These can be produced by the Interactive Vector Symbol Utility, and are used by GDDM to draw high-quality symbols. They can be positioned to the nearest display point, and their size and orientation can be controlled when used by GDDM (See Figure 3-1). VSS cannot be loaded as PS.

Both types of symbol sets can be stored in files, and GDDM contains functions for passing them to and from files and the application program or GDDM storage, and for loading PS sets into the device.

Alphanumeric Operations

Each alphanumeric field created on a page is defined by identification number, location, size, and type. The type defines how the field may be used; for example, a field may be for output only, or may be light-pen sensitive.

Other field characteristics, or attributes, may be specified by application call statements:

- Intensity (normal, invisible, bright)
- Highlighting (normal, blink, reverse video, underscore)
- Color

- Character set to be used

The contents of an alphanumeric field may be specified or changed. For each character position in the field, the application can specify the symbol set to be used, the color, and highlighting.

Additional call statements are available to:

- Reformat fields
- Clear fields
- Query field contents
- Query field attributes
- Query character attributes for a field
- Position the cursor
- Define the mode of operator interaction (whether the operator may or not change character attributes)
- Set default attributes for new field definitions
- Output all fields to the screen and accept new input

Graphics Operations

The construction of pictures involves a hierarchy of definitions which determine the location of the picture on the page, and its form and appearance. These definitions are specified by calls to GDDM from the application program; some of them will assume default values in the absence of specific calls. The definitions are:

- Graphics field
- Picture space
- Viewport
- Window
- Segment
- Primitives

The graphics field is an area on the page in which pictures may be constructed. Only one graphics field is allowed on a page; the field is defined by a call statement which specifies:

- The position on the page (row, column coordinates) of the top left corner of the field
- Height (rows) and width (columns) of the field

The aspect ratio of the field is the ratio of its width to height.

The picture space is the portion of the graphics field to be used for pictures. It is defined by specifying the width and height. One of these must be unity; the other can have any decimal value greater than zero and not exceeding one. This establishes the aspect ratio of the picture space and also defines the basic normalized coordinate system for positioning pictures on the graphics field.

The picture space is mapped onto the graphics field so that the centers coincide and the picture space covers as much of the field as possible while maintaining the picture space aspect ratio. If the aspect ratios of both the field and picture space are identical, the two areas coincide. Otherwise, either the top and bottom edges or the left and right edges coincide. Once defined, either by a specific call or by default, the picture space for a field cannot be changed.

A <u>viewport</u> is a sub-region of the picture space, and is defined in picture-space coordinates, with the coordinate origin at the lower left corner of the picture space. Viewports may be used for positioning the parts of a composite picture; for example, a picture made up of several smaller ones. A specific viewport can be defined by a call statement which includes the positions of the left and right, lower and upper boundaries, in picture-space coordinates. For example: (0,0.5,0,1.0).

A <u>window</u> is a defined area within the picture space, which is used for:

1. Specifying the <u>user-defined</u> coordinates (<u>world coordinates</u>) which are to correspond to the viewport boundaries for the picture.

2. Establishing limiting boundaries for the picture which will appear in the viewport. An option is available to delete those parts of the picture within the picture space but lying outside the viewport. This process is called <u>clipping</u>.

A window is defined by a GDDM call statement which contains the world coordinates of the left, right, lower, and upper boundaries. These are mapped to the corresponding viewport boundaries; hence it is important for the user to realize that if the window and viewport have different aspect ratios, some distortion of the picture will appear. For example, circles would appear as ellipses on the screen.

Once defined, a window remains effective for all graphics operations in the picture space until the window is redefined. Thus several viewports could share a common window.

A <u>segment</u> is a group of logically-related graphic <u>primitives</u> (lines, arcs, text, symbols, etc.) which have a common window and viewport. Segments are created by a call statement in the application program. The segment created becomes the current one, to which primitives are added. The window and viewport cannot be changed while the segment is being constructed.

Graphic <u>primitives</u> are the basic elements used to construct a segment. Operations with primitives may be thought of as moving the cursor on the screen from its current position to another location, which becomes the new current position. When a segment is created, the current position is set to the origin of the world coordinates at the lower left corner of the window.

Functions may be called to:

- Move (without drawing) to a specified position.
- Draw a straight line from current position to a specified position.
- Draw a sequence of straight lines, starting at current position and passing through a specified array of points.
- Draw a circular arc about a specified center point.

- Draw a character string starting at a specified point.
- Append a character string to a previous one.
- Draw a single marker at a specified point, or a series of markers at specified points.
- Shade an area enclosed by a boundary which is defined by lines or arcs.

Attributes are general characteristics of primitives. They may be grouped as follows:

1. Attributes common to all types of graphic primitives:

   a. Color

   b. Color mix (color of a primitive can be opaque, or additively mixed with any underlying color)

2. Line attributes applicable to lines and arcs:

   a. Line type (solid, dotted, dashed, or dash-dot)

   b. Line width (normal, wide)

3. Graphics text attributes:

   a. Character set to be used

   b. Character box size

      This determines the spacing of consecutive characters and rows, or the size of vector symbol characters. Not applicable to hardware or PS characters.

   c. Character angle

      This specifies the direction angle to be used in drawing a character string.

   d. Character mode

      Mode 0: The default; same as Mode 1.

      Mode 1: Applies to either the hardware character set or PS characters. Character box size and angle are ignored.

      Mode 2: Box size determines spacing, actual size depends on the image symbol set in use. Characters remain upright when the angle is not horizontal.

      Mode 3: Used for vector symbol characters. Box size and angle are followed exactly. Individual characters are scaled to fit the box, and are rotated so that they are normal to the angle vector.

4. Shading attributes applicable only to shaded areas:

   a. Standard shading patterns

      Two sets are supplied with GDDM: a set of 16 geometric patterns and a set of seven color shades. In addition, a set of 64 geometric patterns and a set of 64 color shades are supplied in user-loadable form.

b. User-designed shading patterns

These can be created by the user with the Interactive Symbol Editor Utility.

GDDM maintains attribute tables for each graphic segment. Upon creation of a segment, all of its attributes are set to the default values, but call statements are available to reset any of them to desired values during segment construction. Primitives already defined will not be affected by the new values.

Printing

GDDM supports host-directed printing of the page contents. Three types of printing are available:

1. Complete screen copy

   The entire contents of the page as displayed on the screen are copied to equivalent positions on the printer page.

2. Graphics-only copy

   Only the graphics field will be printed. It may be repositioned on the printer page and changed in size if desired.

3. Character-string copy

   A specified character string is printed.

The second form is considered to be more useful in printing pictures, for the following reasons:

- The complete page may contain alphanumeric input fields or a selection menu not wanted on the printed page.
- The printer character size is smaller than that of the terminal, so the vertical resolution might suffer in a complete screen copy. Repositioning and enlarging the picture would produce a more acceptable print.

Errors

GDDM design specifies that parameters passed to GDDM from the application are checked for validity. If an invalid parameter or error is detected, an internal error record is created. This record is passed to an error exit if the severity exceeds a specified threshold; it is also retained for subsequent query.

The error exit can be specified by the application with a call to GDDM; the call also specifies the error threshold. If a threshold of zero is specified, the exit is invoked after each call to any GDDM function, regardless of any errors. The default error exit displays an error message on the terminal.

A call statement is available to return the most recent error record to the application. The parameters included in the error record are:

- Severity of the error

- Error number
- The entry point function code
- Message length
- Error message

INTERACTIVE SYMBOL EDITOR UTILITY

The Interactive Symbol Editor Utility is designed to operate in any host environment which will support GDDM. It allows a user to create or modify an image, which may be used as a character, marker, or shading pattern. Symbol sets of this type are called Image Symbol Sets (ISS).

Image symbols are display-point matrices which can be displayed by GDDM, and the maximum dimensions of a symbol are 255 by 255 points. However, in symbol sets which will be loaded into display device internal storage (Programmed Symbols, or PS), the matrix dimensions must match the cell size of the device to be used at run time. For example, the cell size of an IBM 3279 Display is 9 by 12, and the IBM 3287 Printer cell is 10 by 8. Larger symbols cannot be loaded directly as PS; they can be loaded into GDDM storage. All symbols in a set must have the same matrix dimensions.

ISS created by this utility are stored on named disk files for later use. Each file may contain up to 190 symbols, accessed by hexadecimal index (X'41' to X'FE') or by character (symbol whose hexadecimal index is the same as the character specified). A symbol set can also be generated as a file in "object deck" format. In this form, the symbol set can be link-edited with an application program, which can then use either GDDM or basic subsystem services to transmit the set to a display device.

Upon invocation, the utility communicates with the user by displaying menus and the user enters or changes data in the menu fields. A brief description of these menus follows:

1. Functional Options
   Define symbol set characteristics
   Edit the symbol set
   Test
   Generate object deck

2. Symbol Set Characteristics
   Matrix size
   Color set (monochrome or colors)
   Storage format (display, printer, or graphics)
   Range of hexadecimal codes

3. Symbol Selection
   This menu displays all valid symbols in the set. The symbol to be edited is selected by the user with the cursor. If a new set is being created, this menu will be initially blank.

4. Symbol Definition

The symbol is displayed in matrix form, with one screen cell for each symbol display point, and numbered axes to aid the identification of point positions. The size of the matrix is determined by the symbol size. Scrolling commands are used to access all parts of very large symbols.

Within the matrix, display points which are "on" are indicated by a '#' character of appropriate color on color terminals; on monochrome terminals, this is replaced by the first letter of the color, if other than neutral. "Off" points are either blank, or optionally may be indicated by the period character (.).

Display points can be turned off or on by positioning the cursor and overtyping a blank (off) or any other character (on). The EOF key will also turn a point off. Point color may be set by using color select keys on the terminal, if it has them. Otherwise the current color (selectable by command) is used. This is the procedure for designing or editing multicolored symbols when using the Symbol Editor on a monochrome terminal.

Two symbol matrices can be displayed adjacently if desired. The second is called the reference matrix, and it is used to:

- Display for comparison another symbol from the set being edited
- Store temporarily an alternative version of a new symbol while the user decides which to save

Each menu except Symbol Selection has a command field where the user can enter various commands to control the editing process. Commands are available to terminate editing, save the symbol set on file, clear a symbol, shift a symbol, set color, scroll, etc. A HELP command specifying a command displays explanatory information for the command; if the command name is unspecified, an explanation of the entire Image Symbol Utility is displayed.

Errors detected during editing are immediately reported in the message area at the top of the menu in use. Also, the alarm is sounded.

Using PGF

Many requirements for graphics display can be satisfied with minimal application programming by using the functions offered by PGF. These are the Presentation Graphics Routines, the Interactive Chart Utility, and the Interactive Vector Symbol Utility.

PRESENTATION GRAPHICS ROUTINES

The Presentation Graphics Routines are invoked by call statements in an application program to generate the more common forms of charts. The chart selected is constructed and presented on the display screen or printer by GDDM, using data supplied by the application.

The routines use the appropriate GDDM functions to construct the chart; therefore the application program can be much simpler than one which uses only GDDM. Further simplification is provided by an extensive set of defaults for various parameters. These defaults can be changed by PGF calls, so the user can tailor the chart appearance.

The following types of charts can be constructed:

- Line graph
- Surface chart
- Histogram
- Bar chart
- Pie chart
- Venn diagram

For the first four types, data to be plotted is supplied as a set of paired (independent and dependent) values. Data input for the pie chart and Venn diagram will be explained later.

A chart may contain several such sets of data having common values of the independent variable and different values of the dependent variable. This is a multicomponent chart, and the components can be distinguished by various methods. See Figures 1-2 and 1-5 for examples.

The independent variable axis normally is horizontal with the dependent variable axis vertical, but this orientation may be reversed at the user's option. With the normal alignment, the bars of the histogram and bar charts will be vertical; the Venn diagram and multiple pie charts will be aligned horizontally.

Chart Descriptions

Line Graph (Figure 1-5)
This type consists of a set of data points normally drawn as lines connecting marker symbols which are drawn at each point. Options are available to omit the symbols, leaving just the lines, or to delete the lines, forming a "scatter" plot.

Surface Chart (Figure 1-6)
A surface chart is similar to a line graph without the marker symbols, and with shading and/or color covering the area between the line curve and the independent variable axis or a datum line parallel to it. On multicomponent charts, the areas shaded are bounded by successive component curves. The shading between the first component curve and the axis or other reference line can be optionally omitted.

Histogram (Figure 1-7)
In the histogram, each value of the dependent variable corresponds to a range of values of the independent variable. The data is supplied as a set of ranges and a set of corresponding data values. The histogram is plotted as a set of bars, one for each range of the independent variable. The bar width is proportional to the range, and the length to the corresponding dependent data value. Bars may be shaded and/or colored, and lines common to adjacent bars can be suppressed.

Bar Chart

Bar charts are used when the independent variable is not continuous or has no physical meaning. The bars are evenly spaced along the independent variable axis and the bar length is proportional to the value of the dependent variable.

Multicomponent bar charts can be constructed in three forms:

Composite (Figure 1-2)
: Each bar consists of end-to-end sections. The total bar length is proportional to the sum of the components, and the length of each section represents the contribution of one component.

Multiple (Figure 1-8)
: In this form, each component is represented by an individual bar and the component bars are grouped side by side.

Floating
: This is a special case of a composite bar chart. The difference results from blanking the first component, so the bars are not connected to the axis or reference line. In Figure 1-2, blanking the blue bars would represent a three-component chart of this type.

For all bar charts, the application may specify the ratio of the space between the bars (or bar groups, for multiple bar charts) to the bar width, or use defaults. Color and/or shading of bars can be controlled.

Pie Chart (Figure 1-9)

A pie chart can be used to show how a variable is distributed among several classes. This is represented graphically by dividing a circle into sectors, each sector representing the contribution of each class to the total. Data for a pie chart is provided as a set of values, one for each class or sector. These values may be expressed as absolute values (default) or percentages of the total. A corresponding set of labels identifying the sectors is also supplied by the application.

A multicomponent pie chart consists of a separate pie for each component. Their centers will lie along a horizontal line, or optionally they may be aligned vertically.

Options also permit the reduction of pie size, or the adjustment of the sizes of multiple pies to show their proportion of the total of all sectors. This latter option requires that the input data be absolute rather than percentage values.

Shading/color may be used to distinguish sectors.

Venn Diagram (Figure 1-10)

A Venn diagram represents logical relationships between or among sets of objects. For example, the set of all automobile owners may have some members in common with the set of all computer programmers. The Venn diagram to represent this situation would consist of two overlapping circles. One of the circles represents all car owners, the other all programmers, and the overlap region the subset of programmers who own cars.

PGF allows only two circles for each Venn diagram. The application supplies the data values and labels for the three areas. The circles are arranged horizontally (default) or vertically. The areas may have different shading/color.

Chart Layout

Before using any of the PGF functions, the graphics field must be defined by a call to GDDM. If a chart with a specific aspect ratio is required, the picture space can also be defined by a GDDM call. Alternatively, the default (equal to the graphics field) can be used. Then a PGF call (CHAREA) may be used to define the portion of the picture space to be used for the chart. This is equivalent to defining a viewport in GDDM; the default is the entire picture space.

Within the viewport, PGF sets up a chart layout consisting of a plotting area surrounded by margins (left and right, lower and upper). These margins are reserved for chart titles, labels, legend, etc., and can be changed from their default sizes by the application.

Chart Heading

This is an optional text string to be used as an overall chart heading, which may consist of several lines of text. By default it is centered in the upper chart margin; options exist to left- or right-justify the title, and to place it in the lower margin.

Axis Definitions

Excluding the pie chart and Venn diagram, PGF will normally draw two orthogonal axes at the edges of the plotting area, with the independent variable axis horizontal. Options exist to reverse the axis orientation, or to suppress either axis or both. The axes may be further defined as follows:

Axis label type (numeric (default), alphanumeric)

Axis scale type

An axis may have a linear (default) or logarithmic scale.

Axis intercept

By default, the axes intercept at the point whose coordinates are the minimum values of the axes. Optionally, either axis (or both) can be repositioned by specifying the intersection point.

Axis ranging

The default is <u>autoranging</u>, in which the axis ranges are determined by the extremes of the data passed by the application. Alternatively, the application can specify the range(s) to be used.

<u>Axis Titles</u>

These are optional text strings to be used as titles for the independent and dependent axes. They normally appear in the margins, centered along the axes. Optionally, the titles may be justified to the right for the horizontal axis, to the top for the vertical axis, or the vertical axis title may be placed above the vertical axis.

<u>Axis Scale Marks</u>

Excluding bar charts, pie charts and Venn diagrams, PGF places scale ("tick") marks along the axes, but these may be suppressed if desired. On linear axes, they are spaced at intervals of 1, 2 or 5 axis units multiplied by some power of ten, so they are separated by about nine character spaces. On logarithmic axes, the marks will be placed at each power of ten. The application can override the defaults and specify the intervals for both major and minor scale marks. No minor marks are drawn on logarithmic scales. If used, minor marks are drawn half the length of the majors. The position of the scale marks with respect to the axis can also be specified. By default, they are drawn from the axis outward (towards the margins).

<u>Axis Labeling</u>

If an axis has been specified to have numeric labeling, PGF generates labels from the values of the axis coordinate at the major scale mark positions. By default, the labels will appear adjacent to these points, but optionally they may be placed between them. An option is available to apply a scaling factor to the label values. Another option allows label suppression.

Alphanumeric labeling is required for pie charts and Venn diagrams; it may be used on other charts instead of numeric labeling. The labels are supplied in call statements by the application as lists of alphanumeric strings, in order of use. Labels for pie charts may be drawn adjacent to the sectors, or optionally they may appear in a chart legend. Two tables of predefined labels are provided: months of the year and days of the week. Options permit the use of full spelling or abbreviations.

Secondary Axes

It is possible to define secondary axes in addition to the primary axes usually employed. For example, one could have two vertical axes, at opposite ends of the horizontal axis. Primary and secondary axes are independent; they can have different ranges, scale marks, titles, etc., and data can be plotted with reference to either. Secondary axes do not apply to pie charts or Venn diagrams.

Datum Lines

A datum line is a reference line drawn parallel to an axis. One or more datum lines may be specified for each axis, but they cannot be drawn for pie charts or Venn diagrams.

Grid Lines

The application can specify grid lines parallel to either axis or both. When drawn, each grid line will pass through a major scale mark, which will be deleted. Grid lines are not applicable to pie charts or Venn diagrams.

Chart Legend

For multicomponent charts or pie charts, a chart legend can be added to identify the components or sectors. This contains a list of keys, each consisting of a key symbol followed by a short descriptive label, for each component or sector.

On a multicomponent line graph, the key symbols will be horizontal line segments of the color and line type used for each component. If marker symbols are used, the markers will be centered on the line segments; on a "scatter" plot, only the markers will appear. For all other charts, the key symbols will be small boxes of the color/and or shading pattern used for each component or sector.

The descriptive label for each key appears one character space to the right of the key symbol. These labels are supplied by the application. The following additional options are available:

- The key positions and alignment may be specified.
- The legend may be surrounded by a box.
- The area to be occupied by the legend may be blanked before the keys are drawn.

Chart Attributes

Instead of using PGF default values, the application may issue call statements to specify any of the following:

Text attributes for headings, titles and labels:

Color
Character mode (0,1,2,3; see description of GDDM functions)
Symbol set
Character spacing for mode 2, size for mode 3 (not applicable to mode 0 or 1)

Line attributes for axes, grid lines and datum lines:

Color
Line type
Line width

Component attributes (multicomponent charts):

Color
Marker symbols
Line types
Shading patterns

For multicomponent charts (and the sectors of pie charts), the attributes of each component (or sector) are taken from the current values in the attribute tables in sequential order. When the end of the table is reached, the index is reset to the first value. Default tables of colors, marker symbols, and shading patterns are set up when PGF is initialized. The application can override these and respecify the tables, if desired.

Drawing the Chart

When the chart format and appearance have been defined by specifying the variables described above or utilizing applicable defaults, it can be drawn by calling one of the following PGF routines:

- CHPLOT (line graph)
- CHSURF
- CHHIST
- CHBAR
- CHPIE
- CHVENN

The data values to be plotted are passed as parameters of the call statement.

When the chart has been displayed, a similar one with the same specifications but new data can be produced as follows:

1. If the new chart is to replace the the previous one, a call to GSCLR is used to clear the screen. Alternatively, a call to CHAREA can set up a new chart area in a different location on the same page, retaining the previous chart.

2. A call to CHSTRT resets the attribute tables, etc. to their initial values as used in the previous chart.

3. A call to CHPLOT, CHSURF, etc. with the new data values draws the new chart, with the same titles, labels, axes, legend, etc. as before.

If the next chart is to be completely different from the previous one, a call to CHRNIT would be used in step 2, instead of CHSTRT. This will reinitialize the PGF routines and set all internal variables to their defaults. The application would then issue the necessary calls to define the new chart and display it.

Errors

Parameters passed to the Presentation Graphics Routines are checked as far as possible for validity. When a parameter error is detected, or if an unexpected error occurs, the error module is entered. This constructs an error message and passes control to the GDDM error exit, which is either the default or an exit previously specified by a GDDM call to FSEXIT. The default exit displays a message at the terminal.

INTERACTIVE CHART UTILITY

As noted in Chapter 1, this utility provides an interactive method of using PGF functions to create specific types of charts and to modify their characteristics. Since it is easy to use, either independently or by a user program, the utility can be a valuable tool for both the application programmer and the non-technical terminal user.

From the programmer's viewpoint, the utility offers the following benefits:

- A simple introduction to graphics concepts. The interactive, menu-driven operation allows the programmer to quickly gain familiarity with many PGF functions, by employing the various utility options and observing the results on the terminal screen. This approach also could be used for instructing people who are not programmers on the use of the utility.

- The capability to predesign and save chart format definitions. Suppose that a company or department decides that all basic chart types used to present certain data should be standardized. The charts can be designed with the utility to suit the requirements and stored for subsequent recall. If requirements change, the necessary modifications can be easily made.

- Easy addition of graphics to existing programs. The utility can be invoked by a call statement which includes the data to be plotted. Using the utility functions, the terminal operator can retrieve a chart format, display the chart, modify it and redisplay, or print, then return control (and any data changes, if desired) to the calling program.

INTERACTIVE VECTOR SYMBOL UTILITY

The Interactive Vector Symbol Utility allows the terminal operator to create Vector Symbol Sets (VSS) or to edit existing ones. VSS can be assigned names by the user and saved on files for subsequent retrieval and use by GDDM or PGF for "high-quality" symbols (usually graphics text). They cannot be loaded directly into the display device as PS, but are loaded into GDDM storage by a GDDM call statement.

The user specifies the name of the VSS and identifies the symbol to be created or edited. The utility displays a square grid on the screen; if the symbol already exists, it is superimposed on the grid. The symbol is then defined or modified by specifying lines and/or moves in grid coordinates. Program Function (PF) keys are used by the terminal operator for these operations. The screen display is updated concurrently, so the user can observe the form of the symbol as it is defined or changed. Explanatory information can be displayed on the terminal, if needed.

Vector symbols created with the utility can be sized and/or rotated as desired when used by an application program with GDDM. Figure 3-1 shows an example of such usage.

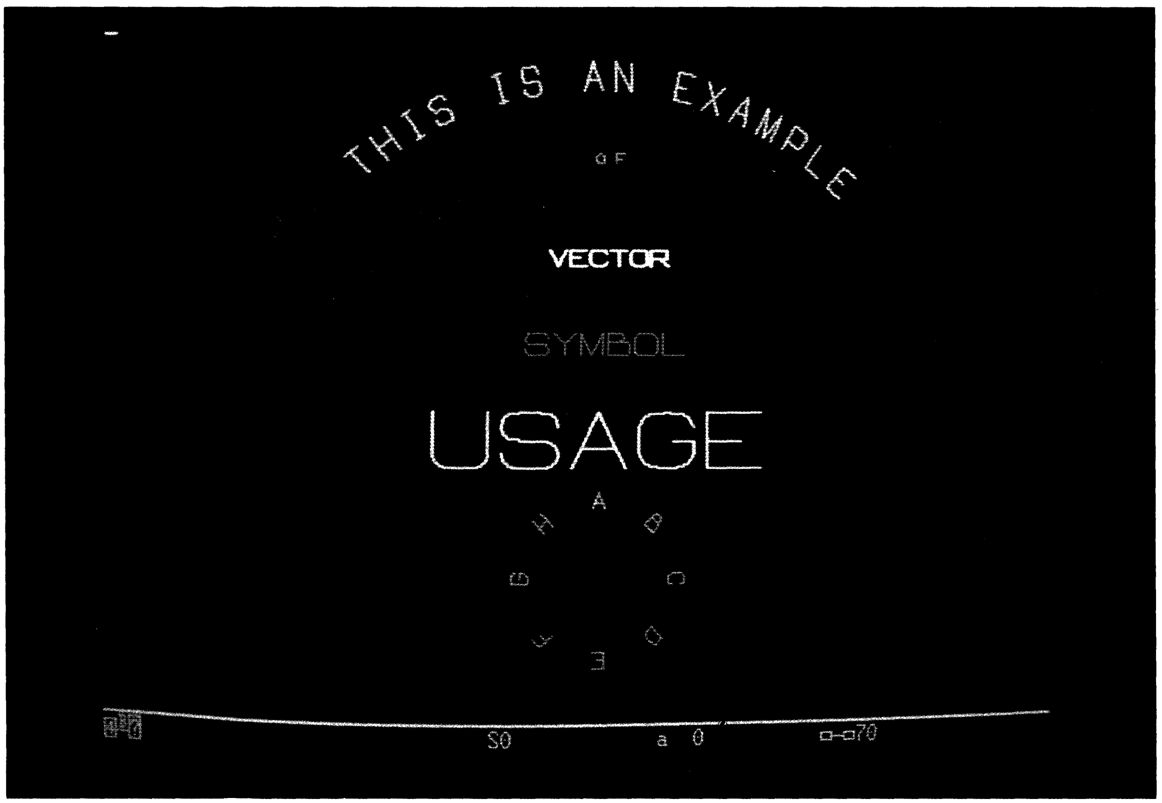

Figure 3-1. Sample Vector Symbols

SAMPLE APPLICATION PROGRAM

The following is a sample application program in PL/I language, which utilizes the PGF Presentation Graphics Routines to produce the histogram shown in Figure 1-7. The program is intended primarily to illustrate the simplicity of using the routines rather than demonstrate their flexibility. With a few exceptions, the default chart attributes are used.

```
HIST: PROC OPTIONS(MAIN);

/***********************************************************************/

/*   DECLARE ENTRY POINTS                                              */

/***********************************************************************/

|  DCL   FSINN ENTRY EXTERNAL OPTIONS(INTER ASM);
|  % INCLUDE ADMUPLNO;

/***********************************************************************/

/*   DECLARE DATA                                                      */

/***********************************************************************/

| DCL    YELLOW(1)FIXED BIN(31) INIT(6),
|        RED(1)FIXED BIN(31) INIT(2),
|        PAT(1)FIXED BIN(31) INIT(1),
|        (MOD,TYP,CNT)FIXED BIN(31);

  DCL    RANGE1(7)FLOAT INIT(0,10,20,30,40,50,60),
         RANGE2(7)FLOAT INIT(10,20,30,40,50,60,70),
         DATA(7)FLOAT INIT(12.3,15,21.3,20.2,15.3,11.4,4.5);

/***********************************************************************/

/*   INITIALIZE & DEFINE GRAPHICS FIELD                                */

/***********************************************************************/

|  CALL FSINN;
   CALL GSFLD(1,1,32,80);

/***********************************************************************/

/*   SPECIFY CHART HEADING                                             */

/***********************************************************************/

   CALL CHHEAD(48,'PERCENTAGE POPULATION DISTRIBUTION BY AGE GROUPS');

/***********************************************************************/

| /*   SET HEADING COLOR                                               */

/***********************************************************************/

|  CALL CHHATT(1,YELLOW);

/***********************************************************************/

/*   SPECIFY AXIS TITLES                                               */

/***********************************************************************/

   CALL CHXTTL(10,'AGE GROUPS');
   CALL CHYTTL(10,'PERCENTAGE');

/***********************************************************************/

/*   SPECIFY X AXIS LABELS & POSITION                                  */

/***********************************************************************/
```

```
|   CALL CHXLAB(8,5,'      <11 11-2021-3031-4041-5051-60 >60 ');
    CALL CHXSET('LABMIDDLE');

  /**********************************************************************/

  /*   SET AXIS RANGES & SCALE MARKS                                    */

  /**********************************************************************/

    CALL CHXRNG(0,70);
    CALL CHYRNG(0,22);
    CALL CHXTIC(10,0);
    CALL CHYTIC(2,1);

  /**********************************************************************/

| /*   SET COLOR & SHADING FOR HISTOGRAM BARS                           */

  /**********************************************************************/

|   CALL CHCOL(1,RED);
|   CALL CHPAT(1,PAT);

  /**********************************************************************/

  /*   DRAW THE CHART                                                   */

  /**********************************************************************/

    CALL CHHIST(1,7,RANGE1,RANGE2,DATA);

| /**********************************************************************/

  /*   DISPLAY THE CHART & SAVE THE DATA STREAM                         */

  /**********************************************************************/

    CALL ASREAD(TYP,MOD,CNT);
    CALL FSSAVE('HIST1');

  /**********************************************************************/

  /*   TERMINATE THE PROGRAM                                            */

  /**********************************************************************/

    CALL CHTERM;
    CALL FSTERM;

  END HIST;
```

Index

Each page number in this index refers to the start of the paragraph containing the indexed item. Where more than one page reference is given, the major reference is first.

Graphical Data Display Manager (GDDM) and
Presentation Graphics Feature (PGF)
General Information

READER'S COMMENT FORM

GC33-0100-0

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Number of your latest Technical Newsletter for this publication. . . . .

Note: *Staples can cause problems with automated mail sorting equipment. Please use pressure-sensitive or other gummed tape to seal this form.*

Cut Along Dotted Line

If you want an acknowledgement, give your name and address below.

Name . . . .

Job Title . . . . Company . . . .

Address . . . .

. . . . Zip . . . .

Thank you for your cooperation. No postage stamp is necessary if mailed in the U.S.A. (Elsewhere, your IBM representative or IBM branch office will be happy to forward your comments.)

Reader's Comment Form

Fold and tape · Please do not staple · Fold and tape

No postage necessary if mailed in the United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT 40 ARMONK, NEW YORK

Postage will be paid by addressee:

International Business Machines Corporation
Department 812HP
1133 Westchester Avenue
White Plains, New York 10604

Fold and tape · Please do not staple · Fold and tape

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

GDDM/PGF General Information Printed in U.S.A. GC33-0100-0

IBM / **Technical Newsletter**

This Newsletter No.	SN33-6264
Date	May 1980
Base Publication No.	GC33-0100-0
File No.	
Previous Newsletters	None

Graphical Data Display Manager and Presentation Graphics Feature: General Information

This technical newsletter, a part of release 1 of the program product Graphical Data Display Manager, provides replacement pages and/or additional pages for the subject manual. Pages to be inserted and/or removed are listed below.

Title page, edition notice
5, 6
15- 18
23-28
33, 34
37, 38

A change to the text is indicated by a vertical line to the left of the change.

Summary of Amendments

The most significant changes are in Chapter 2, where the storage requirements have been revised, and in Chapter 3, where the sample program has been updated. Other, minor changes occur throughout the publication.

Note: *Please file this cover letter at the back of the manual to provide a record of changes.*

IBM United Kingdom Laboratories Ltd., Technical Documentation Department, Hursley Park, Winchester, Hampshire SO21 2JN, England.

GC33-0100-0

GDDM/PGF General Information Printed in U.S.A. GC33-0100-0

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601