INTERNATIONAL STANDARD

ISO/IEC 8632-1

Second edition
1999-12-15

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 1: **Functional specification**

Technologies de l'information — Infographie — Métafichier de stockage et de transfert des informations de description d'images —

Partie 1: Description fonctionnelle

Reference number
ISO/IEC 8632-1:1999(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 734 10 79
E-mail copyright@iso.ch
Web www.iso.ch

Printed in Switzerland

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 8632 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 8632-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics and image processing*.

This second edition cancels and replaces the first edition (ISO/IEC 8632-1:1992), which has been technically revised. Note that the previous edition of ISO/IEC 8632-1, published in 1992, was a first edition but second edition was indicated by error on its cover page and in the foreword.

ISO/IEC 8632 consists of the following parts, under the general title *Information technology — Computer graphics — Metafile for the storage and transfer of picture description information*:

— *Part 1: Functional specification*

— *Part 3: Binary encoding*

— *Part 4: Clear text encoding*

Annexes A, B, C, H and I form a normative part of this part of ISO/IEC 8632. Annexes D to G are for information only.

NOTE In previous editions of ISO/IEC 8632, Part 2 defined a Character Encoding. Part 2 was withdrawn in 1998, due to its lack of implementation and use.

Introduction

0.1 Purpose

The Computer Graphics Metafile provides a file format suitable for the storage and retrieval of picture information. The file format consists of a set of elements that can be used to describe pictures in a way that is compatible between systems of different architectures and devices of differing capabilities and design. This picture description includes the capability for describing static pictures. Static pictures are those where elements which may lead to dynamic effects (for example those leading to regeneration) are prohibited within the picture body.

0.2 Reasons for this International Standard

The main reasons for producing a standard computer graphics metafile are

a) to allow picture information to be stored in an organized way on a graphical software system;

b) to facilitate transfer of picture information between different graphical software system;

c) to enable picture information to be transferred between graphical devices;.

d) to enable picture information to be transferred between different computer graphics installations.

0.3 Design requirements

To reach these objectives, a number of design principles were adopted:

a) The metafile should provide a suitable set of elements for the transfer of a wide range of pictorial information.

b) The metafile should address the more usual and essential features found on graphical devices directly and should provide access to less common facilities via an escape mechanism.

c) The design of the metafile should not preclude extension of ISO/IEC 8632 at a later stage to cover facilities beyond those included in this version of the Standard. It should also not preclude further extensions to support future standards.

d) The metafile should be usable from GKS (Graphical Kernel System — ISO 7942) with both metafile input and metafile output functions. It should include the capability to support ISO 7942 (GKS) static picture capture.

e) ISO/IEC 8632 should address the needs of different applications that have conflicting requirements for size of metafile, speed of generation and interpretation, readability, editability and ease of transfer through different transport mechanisms.

0.4 Design criteria

The requirements of 0.3 were used to formulate the following criteria which were used to decide between different design possibilities.

a) Completeness: In any area of ISO/IEC 8632, the functionality specified by ISO/IEC 8632 should be complete in itself.

b) Conciseness: Redundant elements or parameters should be avoided.

c) Consistency: Contradictory elements should be avoided.

d) Extensibility: The ability to add new elements and generality to ISO/IEC 8632 should not be precluded.

e) Fidelity: The minimal results and characteristics of elements should be well defined.

f) Implementability: An element should be able to be supported efficiently on most host systems and/or graphics hardware.

g) Orthogonality: The elements of the metafile should be independent of each other, or any dependencies should be structured and well defined.

h) Predictability: ISO/IEC 8632 should be such that the recommended or proper use of standard elements guarantees the results of using a particular element.

i) Standard practice: Only those elements that reflect existing practice, that are necessary to support existing practice, or that are necessary to support proposed standards should be standardized.

j) Usefulness: Functions should be powerful enough to perform useful tasks.

k) Well-structured: The assumptions that elements make about each other should be minimized. An element should have a well-defined interface and a simply stated unconditional purpose. Multipurpose elements and side effects should be avoided.

0.5 Access to a metafile

The metafile has been designed so that, although its main usage is anticipated as being with completely sequential access, non-sequential access is also possible. Once the basic environment of the metafile has been established, individual pictures may be accessible if the medium, the encoding and the implementation support this form of access.

0.6 Generation and interpretation of metafiles

The specific mechanisms of metafile generation and interpretation are not described by ISO/IEC 8632, although it does describe the intended result of such interpretation. The basic set of metafile elements includes a capability for the addition of application-dependent data, which do not have graphical meaning and for which no intended interpretation results are described.

0.7 Distinction between formal specification and encodings

The functionality provided by the metafile is separated from the specification of any particular encoding format. ISO/IEC 8632 provides for both standard and private encodings of the elements described in this part of ISO/IEC 8632. Guidelines for private encodings are specified in annex E; these guidelines do not form part of ISO/IEC 8632.

Standardized encodings are specified in parts 3 and 4 of ISO/IEC 8632. Each of the standardized encodings is capable of representing the full functionality described in this part of ISO/IEC 8632. Translation between the standardized encodings is possible without loss of picture information, although subsequent translation back into the original encoding may not result in precisely the same data stream, due to different quantizations of precisions in the different encodings.

The binary encoding specified in ISO/IEC 8632-3 provides an encoding that requires least effort to generate and interpret on many systems, and which is relatively compact as well.

The clear text encoding specified in ISO/IEC 8632-4 provides an encoding that can be created, viewed and edited with standard text editors. It is therefore also suitable for transfer through networks that support only transfer of text files.

0.8 Relationship to other International Standards

ISO/IEC 8632 draws extensively for its model of a graphics system on GKS (Graphical Kernel System — ISO 7942). In addition, ISO/IEC 8632 specifies a metafile that may be used as a static picture-capture metafile by GKS.

This part of ISO/IEC 8632 uses font concepts and the font architecture defined in ISO/IEC 9541-1 for defining CGM references to fonts and font resources. The font properties of ISO/IEC 9541-1 are adopted, where appropriate, to define CGM mechanisms to provide information useful for font substitution between parties interchanging Metafiles. This part of ISO/IEC 8632 includes the minimum font description subset defined in ISO/IEC 9541-2. Clause 3 contains a number of glossary definitions that are taken from, and are identical to, those found in ISO/IEC 9541-1. This part of ISO/IEC 8632 also defines access to extended families of glyph based on the principles and procedures of ISO/IEC 10036.

This part of ISO/IEC 8632 uses a colorimetrically precise reference colour space to allow for interchange of precise colour specifications. It uses concepts defined in ISO/IEC 8613/Amd.2 which are based on CIE publications. ISO/IEC 8613/Amd.2 provides tutorial material on relevant definitions and colour concepts, which is useful for understanding the material in this Standard but is not incorporated into this Standard in that amount of detail.

The binary encoding specified in ISO/IEC 8632-3 employs the mechanism for representing floating point numbers specified in ANSI/IEEE 754-1986.

For certain elements, the CGM defines value ranges of parameters as being reserved for registration. The meanings of these values will be defined using the established procedures (see subclause 6.12) of the ISO International Registration Authority for Graphical Items. These procedures do not apply to values and value ranges defined as being reserved for private use; these values and ranges are not standardized. There is a very close relationship between many of the elements in ISO 8632 and a subset of the functions in the CGI (Computer Graphics Interface — ISO/IEC 9636).

0.9 Versions

ISO/IEC 8632 defines several versions of the Computer Graphics Metafile. A version is defined by a formal grammar and additional specifications contained in clauses 6, 7, and 8.

The following versions are currently defined: Version 1 (one); Version 2 (two); Version 3 (three); and Version 4 (four).

NOTE 1 A valid Version 3 metafile is also a valid Version 4 metafile. A valid Version 2 metafile is also a valid Version 3 metafile. A valid Version 1 metafile is also a valid Version 2 metafile.

NOTE 2 Version 1 metafiles are as defined by the original CGM standard, which was designated ISO 8632:1987. Version 2 metafiles are as defined by the first amendment to the CGM standard, which was designated ISO 8632:1987/Amd.1:1990. Version 3 metafiles are as defined by an amendment to Version 2 of the CGM standard. This amendment was originally designated ISO/IEC 8632:1987/Amd.3:1991, but was never published as an amendment. Instead all documents were consolidated to produce the 1992 revision of ISO 8632; Versions 1, 2, and 3 were all defined by the 1992 revision. Version 4 metafiles were originally defined by ISO/IEC 8632:1992/Amd.2-1995, which has been integrated into this (1999) revision of ISO/IEC 8632; Versions 1, 2, 3, and 4 are all defined by this revision

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 1:
Functional specification

1 Scope

ISO/IEC 8632 provides a file format suitable for the storage and retrieval of picture description information. The file format consists of an ordered set of elements that may be used to describe pictures in a way that is compatible between systems of different architectures, compatible with devices of differing capabilities and design, and meaningful to application constituencies. This picture description includes the capability for describing static images.

The elements specified provide for the representation of a wide range of pictures on a wide range of graphical devices. The elements are organized into groups that delimit major structures (metafiles, pictures, and application structures), that specify the representations used within the metafile, that control the display of the picture, that perform basic drawing actions, that control the attributes of the basic drawing actions, that allow application-specific structuring to be overlaid on the graphical content, and that provide access to non-standard device capabilities.

The metafile is defined in such a way that, in addition to sequential access to the whole metafile, random access to individual pictures and individual context-independent application structures is well-defined. Applications which require random access to pictures and/or context-independent application structures within pictures may, within the metafile, define directories to these pictures and/or context-independent application structures. The metafile may then be opened and randomly accessed without interpreting the entire metafile.

In addition to a functional specification, two standard encodings of the metafile syntax are specified. These encodings address the needs of applications that require small metafile size plus minimum effort to generate and interpret, and maximum flexibility for a human reader or editor of the metafile.

This part of ISO/IEC 8632 describes the format using an abstract syntax. The remaining parts of ISO 8632 specify standardized encodings that conform to this syntax: ISO/IEC 8632-3 specifies a binary encoding; ISO/IEC 8632-4 specifies a clear text encoding.

2 Conformance

ISO/IEC 8632 defines specific criteria for conformance of metafiles, conformance of metafile generators, and conformance of metafile interpreters to this International Standard. Conformance is defined in terms of profiles. Specific conformance criteria for metafiles, generators, and interpreters are defined in clause 9. The criteria for valid, conforming profiles, plus a Model Profile, are specified in clause 9 and annex I. A methodology for defining valid profiles, and a pro forma (Profile Proforma, or PPF) for writing profiles, are defined in clause 9 and annex I.

3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 8632. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO/IEC 8632 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 646:1991, *Information technology — ISO 7-bit coded character set for information interchange.*

ISO 2022:1986, *Information processing — ISO 7-bit and 8-bit coded character sets — Code extension techniques.*

ISO 2375:1985, *Data processing — Procedure for registration of escape sequences.*

ISO 7942:1985, *Information processing systems — Computer graphics — Graphical Kernel System (GKS) functional description.*

ISO 8601:1988, *Data elements and interchange formats — Information interchange — Representation of dates and time.*

ISO 8859-1:1987, *Information processing — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1*.

ISO/IEC 9541-1:1991, *Information technology — Font information interchange — Part 1: Architecture.*

ISO/IEC 9541-2:1991, *Information technology — Font information interchange — Part 2: Interchange format.*

ISO/IEC 9636-1:1991, *Information technology — Computer graphics — Interfacing techniques for dialogues with graphical devices (CGI) — Functional specification — Part 1: Overview, profiles, and conformance.*

ISO/IEC 9636-2:1991, *Information technology — Computer graphics — Interfacing techniques for dialogues with graphical devices (CGI) — Functional specification — Part 2: Control.*

ISO/IEC 9636-3:1991, *Information technology — Computer graphics — Interfacing techniques for dialogues with graphical devices (CGI) — Functional specification — Part 3: Output.*

ISO/IEC 9636-4:1991, *Information technology — Computer graphics — Interfacing techniques for dialogues with graphical devices (CGI) — Functional specification — Part 4: Segments.*

ISO/IEC TR 9973:1988, *Information processing — Procedures for registration of graphical items.*

ISO/IEC 10036, *Information technology — Font information interchange — Procedure for registration of glyph and glyph collection identifiers.*

ISO/IEC TR 10000-1:1992, *Information technology — Framework and taxonomy of International Standardized Profiles — Part 1: Framework.*

ISO/IEC TR 10000-2:1992, *Information technology — Framework and taxonomy of International Standardized Profiles — Part 2: Taxonomy of OSI Profiles.*

ISO/IEC 10646-1:1993, *Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane.*

ISO/IEC 10646-1:1993, Amd.2:1996, *Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane, Amendment 2: UCS Transformation Format 8 (UTF-8).*

CCITT Recommendation T.4 (1988), *Standardization of group 3 facsimile apparatus for document transmission.*

CCITT Recommendation T.6 (1988), *Standardization of group 4 facsimile apparatus for document transmission.*

CIE Publication 15.2, *Colorimetry, 1986 (2nd edition).*

CIE Publication S002, *Colorimetric Observers, 1986 (1st edition).*

4 Terms and definitions

4.1 Definitions

For the purposes of this part of ISO/IEC 8632, the following definitions apply.

NOTE As far as possible, commonly accepted graphics terminology is used.

4.1.1
anisotropic mapping
A mapping in which the scale factors applied along each axis are not equal. This is often used in reference to the mapping from VDC to distance units on the physical display surface. With anisotropic mapping, the angle between any pair of non-parallel line segments can change; circles cease to be circles and become ellipses after such a transformation. See "isotropic mapping".

4.1.2
aspect ratio
The ratio of the width to the height of a rectangular area, such as a window or viewport. For example, an aspect ratio of 2.0 indicates an area twice as wide as it is high.

4.1.3
aspect source flag (ASF)
Indicator as to whether a particular attribute selection is to be individual or bundled.

4.1.4
aspects of primitives
Ways in which the appearance of a primitive can vary. Some aspects are controlled directly by primitive attributes; some are controlled indirectly through a bundle table.

4.1.5
attribute elements
Metafile elements that describe the appearance of graphical element.

4.1.6
boundary
The mathematical locus that defines, in abstract VDC space, the limits of a region to be filled (for fill primitives and closed figures). The visual appearance of interior style ‘hollow' consists of a depiction of the boundary obtained after clipping has been taken into account.

4.1.7
brightness
Attribute of a visual sensation according to which an area appears to emit more or less light.

4.1.8
bundle
Set of attributes associated with one of the following graphical element types: line, marker, text, and filled area.

4.1.9
bundle index
An index into a bundle table for a particular output primitive.

4.1.10
bundle table
An indexed table containing a set of attributes for each index.

4.1.11
character set
The set of displayable symbols mapped to individual characters in a TEXT, APPEND TEXT, or RESTRICTED TEXT string. This corresponds to the "G-set" defined in ISO 2022. A character set is independent of the font or typeface; examples of character sets are: ASCII (X3.4), German and Katakana.

4.1.12
chroma
Colourfulness (chromaticness) of an area judged in proportion to the brightness of a similarly illuminated area that appears white or highly transmitting.

4.1.13
chromaticity
Ratio of each of a set of three tristimulus values to their sum. (As the sum of the three chromaticity coordinates equals 1, two of them are sufficient to define a chromaticity.)

4.1.14
CIELAB colour space
A CIE recommended, approximately uniform, colour space with rectangular coordinates , L^*, a^*, and b^*. L^* is the approximate correlate of lightness, a^* and b^* are used to calculate approximate correlates of hue and chroma. CIELAB uses the tristimulus ratios X/X_n, Y/Y_n, Z/Z_n instead of chromaticity coordinates as in CIELUV. X_n, Y_n, Z_n are the values of X, Y, Z for the appropriate chosen reference white. The reduced perceptual significance of a given difference in chromaticity as the colour becomes darker is incorporated by using the tristimulus ratios.

4.1.15
CIELUV colour space
A CIE recommended, approximately uniform, colour space with rectangular coordinates L^*, u^*, and v^*. L^* is the approximate correlate of lightness, u^* and v^* are used to calculate approximate correlates of hue and chroma. The colour stimulus is described by Y, u', v' and the reference white by Y_n, u_n', v_n'. A given difference in chromaticity is reduced in magnitude by the factor L^* as the colour becomes darker.

4.1.16
CIE tristimulus values
Amounts of the three reference colour stimuli, in a given trichromatic system, required to match the colour of the stimulus considered.

4.1.17
CIE uniform colour spaces
Two CIE recommended, approximately uniform colour spaces, CIELAB and CIELUV, are allowed in the CGM. These colour spaces are non-linear transformations of the CIE 1931 XYZ tristimulus space, into the approximate perceptual correlates of lightness, hue, and chroma. CIELAB and CIELUV closely approximate uniform colour spaces over small distances, and they each provide an approximately uniform measure of perceived colour differences. Both colour spaces allow for the perceptual effect that a given difference in chromaticity represents a smaller and smaller colour difference as the lightness is reduced.

4.1.18
clip indicator
Indicator as to whether metafile graphical elements are to be clipped at the limits of CLIP RECTANGLE.

4.1.19
clipping mode
A generic term referring to one of Line Clipping, Marker Clipping or Edge Clipping Modes. An object clipping may be either ‘locus', ‘shape' or ‘locus then shape'.

4.1.20
clip rectangle
A rectangle defined in VDC space which is used as a clipping boundary when the metafile graphical elements are to be clipped.

4.1.21
clipping
Removing parts of display elements that lie outside a given boundary.

4.1.22
closed figure
A compound primitive that behaves as a fill primitive of more general shape. It is formed by delimiting a sequence of line or fill primitives, edge attributes, and certain control elements, with the elements BEGIN FIGURE and END FIGURE.

4.1.23
CMYK colour space
A colour space based on the subtractive colour mixture of Cyan (C), Magenta (M) and Yellow (Y) primaries with the inclusion of black (K).

4.1.24
colour component
One of the dimensions of a colour space.

4.1.25
colour model
A specification of a 3D colour coordinate system and a 3D subspace in the coordinate system within which each displayable colour is represented by a point. Some colour models include a fourth, redundant, dimension to allow the independent representation of black. For the purpose of ISO/IEC 8632 colour model refers to one of RGB, CIELAB, CIELUV, CMYK, or RGB-related, which may be designated as the colour space for metafile content, or CIEXYZ, which is used in this International Standard as a reference space for specifying colour transformations.

4.1.26
colour selection mode
Indicator as to whether colour selection is to be direct (by specifying a colour value) or indexed (by specifying an index into a table of colour values). See COLOUR VALUE.

4.1.27
colour space
See COLOUR MODEL.

4.1.28
colour stimulus
Visible radiation entering the eye and producing a sensation of colour.

4.1.29
colour table
A table for use in mapping from a colour index to the corresponding colour. See DIRECT COLOUR, INDEXED COLOUR.

4.1.30
colour value
Value of the n-tuple of components describing a colour in a given colour model.

4.1.31
compound primitive
A compound primitive is specified by a sequence of CGM elements, as opposed to primitives represented by a single element. Compound text and closed figures are examples of compound primitives in the CGM.

4.1.32
Computer Graphics Interface (CGI)
The specification for interface techniques for dialogues with graphical devices.

4.1.33
Computer Graphics Metafile (CGM)
The specification for a mechanism for storing and transferring picture description information.

4.1.34
conjugate diameter pair (CDP)
A pair D,d of diameters of an ellipse such that a tangent to the ellipse at each endpoint of one diameter is parallel to the other diameter.

4.1.35
control elements
Metafile elements that specify address space, clipping boundaries, picture delimiters, and format descriptions of the Metafile elements.

4.1.36
data interface
An interface between software modules or devices comprising one or more packets containing opcodes and data (as contrasted with a subroutine or function call interface).

4.1.37
descriptor elements
Metafile elements that describe the functional content, format, default conditions, identification, and characteristics of the Metafile.

4.1.38
device coordinates
The coordinates native to a device; device-dependent coordinates; physical device coordinates.

4.1.39
device driver
The device-dependent part of a graphics implementation which supports a physical device. The device driver generates device-dependent output.

4.1.40
device viewport
A rectangular subset of the physical display surface into which VDC EXTENT is mapped. See "effective viewport".

4.1.41
direct colour
A colour selection scheme in which the colour values are specified directly, without requiring an intermediate mapping via a colour table. See COLOUR TABLE, INDEXED COLOUR.

4.1.42
display surface
That part of a graphics device upon which a visible image appears (for example, the screen of a display, the paper in a plotter).

4.1.43
edge
The rendering of the perimeter of a filled region, controlled by edge attributes. Edges are clipped after being applied to the boundary, as distinct from the rendition of the boundary obtained from interior style ‘hollow'. See "boundary".

4.1.44
effective viewport
The actual viewport resulting from forced isotropic mapping from the VDC extent to the viewport.

4.1.45
escape elements
Metafile elements that describe device- or system-dependent elements used to construct a picture, but that are not otherwise standardized.

4.1.46
external elements
Metafile elements that communicate information not directly related to the generation of a graphical image.

4.1.47
escapement
During the rendering of text strings onto a display, the movement of the current position on the display surface after a glyph representation is imaged.

4.1.48
escapement point
A glyph metric; a point in the glyph coordinate system, to which the current position on the display surface is usually translated, after the glyph representation is imaged.

4.1.49
font
A collection of glyph images having the same basic design, e.g., *Courier Bold Oblique*.

4.1.50
font family
A collection of fonts of common design, e.g., *Courier, Courier Bold, Courier Bold Oblique*.

4.1.51
font metrics
The set of dimensions and positioning information in a font resource common to all glyph representations contained in that font resource.

4.1.52
font resource
A collection of glyph representations together with descriptive and font metric information which are relevant to the collection of glyph representations as a whole.

4.1.53
foreground colour
The colour used in the rendering process in which primitives are rendered on the display surface, as opposed to the BACKGROUND COLOUR or AUXILIARY COLOUR. The foreground colour is set separately for each class of primitive.

4.1.54
glyph
A recognizable abstract graphical symbol, which is independent of any specific design.

4.1.55
global segment
A segment that is defined in the Metafile Descriptor (see "segment"), in which case it may be referenced from within any picture.

4.1.56
graphical primitive elements
Metafile elements that describe images in the Metafile.

4.1.57
Graphical Kernel System (GKS)
An ISO standard application programmer's interface to graphics (ISO 7942).

4.1.58
graphics device
A device (for example, refresh display, storage tube display, or plotter) on which display images can be represented.

4.1.59
graphic object
A graphical primitive or a compound primitive, together with the associated attributes.

4.1.60
hatch style
A format for filling closed figures. A hatch style consists of one or more sets of lines whose presence represents the interior of the figure in question.

4.1.61
hue
Attribute of a visual sensation according to which an area appears to be similar to one of the perceived colours, red, yellow, green, and blue, or to a combination of two of them.

4.1.62
indexed colour
A colour selection scheme in which the colour index is used to retrieve colour values from a colour table. See COLOUR TABLE, DIRECT COLOUR.

4.1.63
isotropic mapping
A mapping which is invariant with respect to direction; equal scaling in all orthogonal representational dimensions. It is often used to describe the mapping from VDC to distance units on the physical display surface. With isotropic mapping, the angle between any pair of non-parallel line segments remains unchanged; for example, circles remain circles. See "anisotropic mapping".

4.1.64
local segment
A segment which is defined in the picture descriptor or picture body, and whose definition is local to the picture in which it appears.

4.1.65
message
A string of characters used to communicate information to operators at Metafile interpretation time.

4.1.66
metafile
A mechanism for retaining and transporting graphical data and control information. This information contains a device-independent description of one or more pictures.

4.1.67
Metafile Descriptor (MD)
A metafile element that describes the format of the metafile (but not its encoding method) and the functionality expected of a metafile interpreter.

4.1.68
metafile element
A functional item that can be used to construct a picture or convey information.

4.1.69
metafile generator
The process or equipment that produces the Computer Graphics Metafile.

4.1.70
metafile interpreter
The process or equipment that reads the Computer Graphics Metafile and interprets the contents. An interpreter may be needed in order to drive a Computer Graphics Interface or other device interface to obtain a picture that resembles the intended picture as closely as possible.

4.1.71
normalized device coordinates (NDC)
Coordinates specified in a device-independent coordinate system, normalized to some range (typically 0 to 1). See VDC EXTENT, VDC RANGE, VDC SPACE, VIRTUAL DEVICE COORDINATES.

4.1.72
object clipping
Object clipping is applied to a graphic object. For example, clipping is applied to a line after it has had the width attribute associated with it.

4.1.73
pattern style
A format for filling closed figures with patterns. A pattern style consists of an array of variously coloured or shaded cells

4.1.74
Picture Descriptor (PD)
A set of metafile elements used to set the interpretation modes of attribute elements for the entire picture.

4.1.75
pixel
The smallest element of a display surface that can be independently assigned colour.

4.1.76
posture
The extent to which the shape of a glyph or set of glyphs appear to incline, including any consequent design or form change.

4.1.77
primary colour stimuli
Three selected coloured lights used to specify the colour of any light represented by the amounts of the three lights that must be mixed additively to produce light matching the light presented. (Any three coloured lights may serve as primaries provided no one of them can be matched by a mixture of the other two. To achieve the maximum gamut of colours by additive mixture, saturated red, green, and blue primaries are commonly used.)

4.1.78
primary colorants
A small number of colorants (dyes or pigments) that may be mixed subtractively to produce a large gamut of colours. The most common primary colorants are cyan (greenish blue), magenta (purplish red), and yellow.

4.1.79
realized edge
The zero-width ideal boundary line of the filled-area if the edge is invisible, and the finite-width displayed line if the edge is visible.

4.1.80
realized interior
In a filled area element, that portion of the ideal interior as extending to and terminating at the realized edge.

4.1.81
region
In the context of closed figures or the POLYGON SET element, an area that is explicitly or implicitly closed, that is a subset of the full area being filled. Regions can be nested, disjoint or overlapping. The boundaries of all regions are considered together when applying the interior test for filling a closed figure or POLYGON SET.

4.1.82
RGB colour space
A colour space with colorimetric coordinates based on red, green and blue reference stimuli or primaries. Colour values may be negative in certain areas outside the gamut defined by the RGB primaries.

4.1.83
RGB-related colour space
A colour space related to the RGB colour space through a linear transformation (3x3 matrix). (Examples are YUV (PAL, SECAM), YIQ (NTSC) or $\mathbf{YC_RC_B}$ (CCITT video codecs and ITU 601 studio standard)).

4.1.84
reference colour model
Basic colour model within CGM relative to which relationships to specifiable colour models (RGB, CMYK, CIELUV, CIELAB, and RBG-related) are calibrated. The reference colour model is defined by the CIE 1931 standard colorimetric system (XYZ).

4.1.85
saturation
Colourfulness of an area judged in proportion to its brightness.

4.1.86
segment
A collection of primitives, primitive attributes and some additional attributes associated with the segment as a whole. See "segment attribute".

4.1.87
segment attribute
An attribute associated with a segment as a whole rather than attributes of individual primitives.

4.1.88
size specification mode
A generic term for Line Width Specification Mode, Edge Width Specification Mode, or Marker Size Specification Mode.

4.1.89
skewed
Used to describe stroke precision text when the CHARACTER ORIENTATION vectors are non-perpendicular; CELL ARRAYs when the three defining points form a parallelogram which is not a rectangle; or a segment transformation that causes rectangles to become non-rectangular parallelograms.

4.1.90
symbol
A graphical object which is included by reference at some point in the metafile.

4.1.91
trichromatic system
System for specifying colour stimuli in terms of tristimulus values, based on matching colours by additive mixture of three suitable chosen reference colour stimuli.

4.1.92
view surface
See DISPLAY SURFACE.

4.1.93
virtual device
An idealized graphics device that presents a set of graphics capabilities to graphics software or systems via the Computer Graphics Interface.

4.1.94
virtual device coordinates (VDC)
The coordinates used to specify position in the VDC space. These are absolute two-dimensional coordinates. See VDC SPACE.

4.1.95
VDC extent
A rectangular region of interest contained within the VDC range. See VDC RANGE, VDC SPACE.

4.1.96
VDC range
A rectangular region within VDC space consisting of the set of all coordinates representable in the declared coordinate type, precision, and encoding format of the metafile. See VDC EXTENT, VDC SPACE.

4.1.97
VDC space
A two-dimensional Cartesian coordinate space of infinite precision and extent. Only a subset of VDC space, the VDC range, is realizable in a metafile. See VDC EXTENT, VDC RANGE, VIRTUAL DEVICE COORDINATES.

4.1.98
weight
The ratio of a glyph's or set of glyphs' stem width to font height.

4.1.99
colour device
A device which offers more than two colours.

4.1.100
grey-scale device
A special case of colour device where hue and saturation are the same for all colours (generally saturation = 0).

4.1.101
geometric degeneracy
The degeneracy is intrinsic to the parameterization of the element. A degeneracy which results when parameterization for the geometry does not provide sufficient information to draw the intended primitive.

4.1.102
interoperability
The generator and the interpreter have the same understanding of the encodings, the syntax, and the semantics of the metafile.

4.1.103
monochrome device
A device which has only two colours, a foreground and a background colour. The background colour is the colour of the display surface after it has been cleared.

4.1.104
Profile Proforma (PPF)
A template consisting of profile specifications, which is used by writers of profiles for generating instances of a profile. A completed PPF specifies the rules and options of a profile of ISO/IEC 8632.

4.1.105
application structure
A sequence of metafile elements delimited by the BEGIN APPLICATION STRUCTURE and END APPLICATION STRUCTURE elements, containing one BEGIN APPLICATION STRUCTURE BODY element, and optionally containing one or more APPLICATION STRUCTURE ATTRIBUTE elements.

5 Symbols and abbreviated terms

The following abbreviations are used in all parts of ISO/IEC 8632.

APS Application Structure

ASF Aspect Source Flag

CDP	Conjugate Diameter Pair
CGI	Computer Graphics Interface
CGM	Computer Graphics Metafile
GDP	Generalized Drawing Primitive
GKS	Graphical Kernel System
ISP	International Standardized Profile
MD	Metafile Descriptor
NDC	Normalized Device Coordinate(s)
PD	Picture Descriptor
PPF	Profile Proforma
VDC	Virtual Device Coordinate(s)

6 Concepts

6.1 Introduction

The objective of the Computer Graphics Metafile (CGM) is to provide for the description, storage, and communication of graphical information together with related, application-specific information in a device-independent manner. To accomplish this, ISO/IEC 8632 defines the form (syntax) and functional behaviour (semantics) of a set of elements that may occur in the CGM. The following classes of elements are defined:

— Delimiter Elements, which delimit significant structures within the Metafile.

— Metafile Descriptor Elements, which describe the functional content, default conditions, identification, and characteristics of the CGM; and optionally, define a directory.

— Picture Descriptor Elements, which set the interpretation modes of attribute elements for each picture and optionally, define a directory to the application structures contained in each picture.

— Control Elements, which allow picture boundaries and coordinate representation to be modified.

— Graphical Primitive Elements, which describe the visual components of a picture in the CGM.

— Attribute Elements, which describe the appearance of graphical primitive elements.

— Escape Elements, which describes device- or system-dependent elements used to construct a picture; however, the elements are not otherwise standardized.

— External Elements, which communicate information not directly related to the generation of a graphical image.

— Segment Elements, which enable the grouping of graphic objects for graphical operations such as copying.

— Application structure elements, which enable the grouping of elements for retrieval, electronic linking, and other specific application-dependent operations.

A Computer Graphics Metafile is a collection of elements from this standardized set. The BEGIN METAFILE and END METAFILE elements each occur exactly once in a complete metafile; as many or as few of the elements in the other classes may occur as are needed. A metafile needs to be interpreted in order to display its pictorial content on a graphics device. The descriptor elements give the interpreter sufficient data to interpret metafile elements and to make informed decisions concerning the resources needed for display.

Any CGM contains certain delimiter elements. In addition it may include control elements for metafile interpretation, Picture Descriptor elements for declaring parameter modes of attribute elements and defining the representations of certain attribut index values, graphical primitive elements for defining graphical entities, attribute elements for defining the appearance of the graphical primitive elements, escape elements for accessing non-standardized features of particular devices, and external elements for communication of information external to the definition of the pictures in the CGM. In Version 2, Version 3, and Version 4 metafiles, graphical output primitives, attributes, and control elements may be grouped in segments.

A minimal correct metafile consists of BEGIN METAFILE, a Metafile Descriptor consisting of METAFILE VERSION and METAFILE ELEMENT LIST, and END METAFILE. In Version 4 metafiles, graphical output primitives, attributes, control elements, and application structure attributes may be grouped in application structures.

6.2 Delimiter elements

Every metafile starts with a BEGIN METAFILE element and ends with an END METAFILE element. This allows multiple metafiles to be stored or transferred together.

Each picture starts with a BEGIN PICTURE element and ends with an END PICTURE element. Between these delimiters the Picture Descriptor is separated from the picture body by a BEGIN PICTURE BODY element.

Once the Metafile Descriptor has been read, access to individual pictures, on a random as opposed to sequential basis, may be safely accomplished if the encoding, access mechanism and implementation permit.

BEGIN METAFILE and BEGIN PICTURE both have parameters for a name by which the metafile and picture (respectively) can be identified.

In Version 2, Version 3 and Version 4 metafiles, primitives may be grouped together to form a composite primitive known as a closed figure. The primitives to be included in the closed figure being defined are delimited by the elements BEGIN FIGURE and END FIGURE.

In Version 2, Version 3 and Version 4 metafiles, groups of elements, called segments, are delimited by BEGIN SEGMENT and END SEGMENT. Each segment is uniquely identified by a segment identifier. Segments may be defined in the Metafile Descriptor, in the Picture Descriptor, or within picture bodies.

In Version 3 and Version 4 metafiles, a compound clipping or shielding region may be defined by line and filled-area elements occurring between BEGIN PROTECTION REGION and END PROTECTION REGION.

In Version 3 and Version 4 metafiles a compound path may be defined for drawing a compound line primitive. A compound line is defined by line primitive elements occurring between BEGIN COMPOUND LINE and END COMPOUND LINE elements. A compound path may also be defined for displaying text strings along an arbitrary text path. A compound text path is defined by line primitive elements occurring between BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements. The allowed elements and rules for definition are identical for the two types of paths.

In Version 3 and Version 4 metafiles, a tile array may be defined by tile array elements occurring between BEGIN TILE ARRAY and END TILE ARRAY.

The exact list of elements which may occur in any of these definition states will be found in the state table, Table 8a.

In Version 4 metafiles, application structures consisting of groups of elements within picture bodies, are delimited by BEGIN APPLICATION STRUCTURE and END APPLICATION STRUCTURE. Between these delimiters the BEGIN APPLICATION STRUCTURE BODY element separates the APPLICATION STRUCTURE ATTRIBUTE elements in the application structure descriptor from the other elements contained in the application structure body.

6.3 Metafile descriptor elements

The Metafile Descriptor (MD) is a group of elements that describes the functional capabilities required to interpret the CGM. These elements are

METAFILE VERSION
METAFILE DESCRIPTION
VDC TYPE
INTEGER PRECISION
REAL PRECISION
INDEX PRECISION
COLOUR PRECISION
COLOUR INDEX PRECISION
MAXIMUM COLOUR INDEX
COLOUR VALUE EXTENT
METAFILE ELEMENT LIST
METAFILE DEFAULTS REPLACEMENT
FONT LIST
CHARACTER SET LIST
CHARACTER CODING ANNOUNCER
NAME PRECISION
MAXIMUM VDC EXTENT
SEGMENT PRIORITY EXTENT
COLOUR MODEL
COLOUR CALIBRATION
FONT PROPERTIES
GLYPH MAPPING
SYMBOL LIBRARY LIST
PICTURE DIRECTORY

NOTE 1 Other elements, as defined in this part of ISO/IEC 8632, may appear within the Metafile Descriptor within the definition of a global segment.

In a particular metafile, the METAFILE ELEMENT LIST lists at least those standardized elements that occur in the metafile. The CGM interpreter is thus informed of the capabilities required to successfully interpret the Computer Graphics Metafile. The CGM contains a single Metafile Descriptor. The Metafile Descriptor immediately follows the BEGIN METAFILE element in a metafile (with the possible exception of intervening external and escape elements).

METAFILE VERSION and METAFILE ELEMENT LIST shall occur only once in the Metafile Descriptor for metafiles of all three versions.

NOTE 2 It is recommended that the following elements: METAFILE VERSION, METAFILE ELEMENT LIST and (possibly multiple occurrences of) METAFILE DESCRIPTION appear first in the Metafile Descriptor and in the order listed.

6.3.1 Identification

The identifying information includes declaration of the version of ISO/IEC 8632, by the METAFILE VERSION element, and descriptive information about the metafile (origin, owner, generation date, etc.), by the METAFILE DESCRIPTION element.

6.3.2 Functional capability

The contents of the Computer Graphics Metafile are defined by the METAFILE ELEMENT LIST element. This shall contain a list of the non-mandatory elements that are utilized in the metafile (see 6.1 for mandatory elements). Several shorthand names for CGM elements are also provided for use with the METAFILE ELEMENT LIST. The shorthand names shall not be considered macro names, nor shall they be construed to be levels of conformance

6.3.2.1 Drawing set

The drawing set includes the mandatory CGM elements (i.e., those that shall appear in every conforming CGM) and most of the graphical primitive elements and attribute elements. The drawing set is specified by the shorthand name DRAWING SET.

The elements included in the drawing set are:

BEGIN METAFILE
END METAFILE
BEGIN PICTURE
BEGIN PICTURE BODY
END PICTURE
METAFILE VERSION
METAFILE DESCRIPTION
VDC TYPE
METAFILE ELEMENT LIST
AUXILIARY COLOUR
TRANSPARENCY
CLIP RECTANGLE
CLIP INDICATOR
LINE TYPE
LINE WIDTH
LINE COLOUR
MARKER BUNDLE INDEX
MARKER TYPE
MARKER SIZE
MARKER COLOUR
TEXT BUNDLE INDEX
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOUR

VDC EXTENT
BACKGROUND COLOUR
COLOUR SELECTION MODE
POLYLINE
DISJOINT POLYLINE
POLYMARKER
TEXT
RESTRICTED TEXT
APPEND TEXT
POLYGON
POLYGON SET
CELL ARRAY
GENERALIZED DRAWING PRIMITIVE
RECTANGLE
CIRCLE
CIRCULAR ARC 3 POINT
CIRCULAR ARC 3 POINT CLOSE
CIRCULAR ARC CENTRE
CIRCULAR ARC CENTRE CLOSE
ELLIPSE
ELLIPTICAL ARC
ELLIPTICAL ARC CLOSE
LINE BUNDLE INDEX
CHARACTER HEIGHT
CHARACTER ORIENTATION
TEXT PATH
TEXT ALIGNMENT
FILL BUNDLE INDEX
INTERIOR STYLE
FILL COLOUR
HATCH INDEX
PATTERN INDEX
EDGE BUNDLE INDEX
EDGE TYPE
EDGE WIDTH
EDGE COLOUR
EDGE VISIBILITY
FILL REFERENCE POINT
PATTERN TABLE
PATTERN SIZE
COLOUR TABLE
ASPECT SOURCE FLAGS
ESCAPE
MESSAGE
APPLICATION DATA

6.3.2.2 Drawing plus control set

The drawing-plus-control set may be used to indicate all of the elements in the drawing set plus additional control, Metafile Descriptor, Picture Descriptor, and attribute elements. It is specified by the shorthand name DRAWING PLUS CONTROL SET.

The elements included in the drawing-plus-control set are all of the elements in the drawing set and the following elements:

INTEGER PRECISION
REAL PRECISION
INDEX PRECISION
COLOUR PRECISION
COLOUR INDEX PRECISION
MAXIMUM COLOUR INDEX
COLOUR VALUE EXTENT
METAFILE DEFAULTS REPLACEMENT
FONT LIST
CHARACTER SET LIST
CHARACTER CODING ANNOUNCER
VDC INTEGER PRECISION
VDC REAL PRECISION
SCALING MODE
LINE WIDTH SPECIFICATION MODE
MARKER SIZE SPECIFICATION MODE
EDGE WIDTH SPECIFICATION MODE
CHARACTER SET INDEX
ALTERNATE CHARACTER SET INDEX

NOTE The drawing-plus-control set essentially constitutes a "Version-1 set", however that designation was not defined in the original ISO 8632:1987 and therefore cannot be used in a Version 1 metafile for reasons of upward compatibility of versions.

6.3.2.3 Version 2 set

The Version-2 set includes all elements from the drawing-plus-control set and the following elements defined in Version 2 metafiles:

BEGIN SEGMENT
END SEGMENT
BEGIN FIGURE
END FIGURE
NAME PRECISION
MAXIMUM VDC EXTENT
SEGMENT PRIORITY EXTENT
DEVICE VIEWPORT
DEVICE VIEWPORT MAPPING
DEVICE VIEWPORT SPECIFICATION MODE
MARKER CLIPPING MODE
EDGE CLIPPING MODE
NEW REGION
SAVE PRIMITIVE CONTEXT
RESTORE PRIMITIVE CONTEXT
CIRCULAR ARC CENTRE REVERSED
CONNECTING EDGE
PICK IDENTIFIER
COPY SEGMENT
INHERITANCE FILTER

LINE REPRESENTATION
MARKER REPRESENTATION
TEXT REPRESENTATION
FILL REPRESENTATION
EDGE REPRESENTATION
LINE CLIPPING MODE
CLIP INHERITANCE
SEGMENT TRANSFORMATION
SEGMENT HIGHLIGHTING
SEGMENT DISPLAY PRIORITY
SEGMENT PICK PRIORITY

6.3.2.4 Extended primitives set

The extended-primitives set may be used to indicate those primitives which are available in Version 1 metafiles but are not defined in ISO 7942 (GKS). These elements are:

DISJOINT POLYLINE
RESTRICTED TEXT
APPEND TEXT
POLYGON SET
RECTANGLE
CIRCLE
CIRCULAR ARC 3 POINT
CIRCULAR ARC CENTRE
CIRCULAR ARC CENTRE CLOSE
CIRCULAR ARC CENTRE REVERSED
ELLIPSE
ELLIPTICAL ARC
ELLIPTICAL ARC CLOSE
CIRCULAR ARC 3 POINT CLOSE

6.3.2.5 Version 2 GKSM set

The Version-2-GKSM set is a set of Version 2 metafile elements for ISO 7942 (GKS) picture capture. The elements included in the Version-2-GKSM set are:

BEGIN METAFILE
BEGIN PICTURE
BEGIN PICTURE BODY
END PICTURE
BEGIN SEGMENT
END SEGMENT
END METAFILE
METAFILE VERSION
METAFILE DESCRIPTION
VDC TYPE
INTEGER PRECISION
REAL PRECISION
INDEX PRECISION
COLOUR PRECISION
COLOUR INDEX PRECISION
NAME PRECISION
MAXIMUM COLOUR INDEX
COLOUR VALUE EXTENT
METAFILE ELEMENT LIST
METAFILE DEFAULTS REPLACEMENT
FONT LIST
CHARACTER SET LIST
CHARACTER CODING ANNOUNCER
MAXIMUM VDC EXTENT
SEGMENT PRIORITY EXTENT
VDC EXTENT
DEVICE VIEWPORT
DEVICE VIEWPORT MAPPING
DEVICE VIEWPORT SPECIFICATION MODE
LINE REPRESENTATION
MARKER REPRESENTATION
TEXT REPRESENTATION
FILL REPRESENTATION
VDC INTEGER PRECISION
VDC REAL PRECISION
CLIP RECTANGLE
CELL ARRAY
GDP
LINE BUNDLE INDEX
LINE TYPE
LINE WIDTH
LINE COLOUR
MARKER BUNDLE INDEX
MARKER TYPE
MARKER SIZE
MARKER COLOUR
TEXT BUNDLE INDEX
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOUR
CHARACTER HEIGHT
CHARACTER ORIENTATION
TEXT PATH
TEXT ALIGNMENT
CHARACTER SET INDEX
ALTERNATE CHARACTER SET INDEX
FILL BUNDLE INDEX
INTERIOR STYLE
FILL COLOUR
HATCH INDEX
PATTERN INDEX
FILL REFERENCE POINT
PATTERN TABLE
PATTERN SIZE
COLOUR TABLE
ASPECT SOURCE FLAGS
PICK IDENTIFIER
ESCAPE
MESSAGE
APPLICATION DATA

POLYLINE
POLYMARKER
TEXT
POLYGON
SEGMENT TRANSFORMATION
SEGMENT HIGHLIGHTING
SEGMENT DISPLAY PRIORITY
SEGMENT PICK PRIORITY

6.3.2.6 Version 3 set

The Version-3 set may be used to indicate all elements in the Version-2 set and the elements:

BEGIN PROTECTION REGION
END PROTECTION REGION
BEGIN COMPOUND LINE
END COMPOUND LINE
BEGIN COMPOUND TEXT PATH
END COMPOUND TEXT PATH
BEGIN TILE ARRAY
END TILE ARRAY
COLOUR MODEL
COLOUR CALIBRATION
FONT PROPERTIES
GLYPH MAPPING
SYMBOL LIBRARY LIST
INTERIOR STYLE SPECIFICATION MODE
LINE AND EDGE TYPE DEFINITION
HATCH STYLE DEFINITION
GEOMETRIC PATTERN DEFINITION
PROTECTION REGION INDICATOR
GENERALIZED TEXT PATH MODE
MITRE LIMIT
TRANSPARENT CELL COLOUR
HYPERBOLIC ARC
PARABOLIC ARC
NON-UNIFORM B-SPLINE
NON-UNIFORM RATIONAL B-SPLINE
POLYBEZIER
POLYSYMBOL
BITONAL TILE
TILE
LINE CAP
LINE JOIN
LINE TYPE CONTINUATION
LINE TYPE INITIAL OFFSET
TEXT SCORE TYPE
RESTRICTED TEXT TYPE
INTERPOLATED INTERIOR
EDGE CAP
EDGE JOIN
EDGE TYPE CONTINUATION
EDGE TYPE INITIAL OFFSET
SYMBOL LIBRARY INDEX
SYMBOL COLOUR
SYMBOL SIZE
SYMBOL ORIENTATION

6.3.2.7 Version 4 set

The Version 4 set may be used to indicate all the elements in Version 3 and the elements:

PICTURE DIRECTORY
APPLICATION STRUCTURE DIRECTORY
BEGIN APPLICATION STRUCTURE
BEGIN APPLICATION STRUCTURE BODY
END APPLICATION STRUCTURE
APPLICATION STRUCTURE ATTRIBUTE

6.3.3 Default metafile state

The default state is the state to which the interpreter is returned at the start of each picture. The default states of all metafile elements are defined in clause 8. These default values may be selectively replaced by using the METAFILE DEFAULTS REPLACEMENT element. The correspondence between character set indexes and registered or private character sets, and the meaning assigned to text font indexes, are also established in the Metafile Descriptor.

6.3.4 Fonts and character sets

6.3.4.1 Font list and font resources

ISO/IEC 9541-1 defines an architecture for font resources, but does not define or standardize applications' use of the information in a font resource — ranging from gross or aggregate properties such as font posture to very specific and detailed properties such as individual glyph metrics. A metafile generator (with its associated application) will be a user of such font resource information. The application, in defining a picture which contains text strings, has knowledge of the properties of the font resource. It makes use of these properties to format or layout strings of text so that the complete strings have the desired characteristics.

CGM is used to transmit such pictures from a generating application to an interpreting application, possibly remote in time and space and possibly of very different architecture and resource availability. The font facilities of CGM are designed to provide a font referencing mechanism. Font referencing is the process of identifying or characterizing a font resource. Referencing may include identification of a specific font by name, or provide sufficient descriptive information to permit identification of a suitable font or substitute. This concept is described in ISO/IEC 9541-1, annex B.

The FONT LIST element of CGM allows the exact naming of a font resource. Such font resources may in the future be registered and given structured names under the mechanisms of ISO/IEC 9541. In the ideal case the metafile interpreter recognizes and has available the font resource named in the FONT LIST. For cases where the named font is not available to the interpreter, the CGM has elements (FONT PROPERTIES and GLYPH MAPPING) which allow generators to pass to interpreters additional descriptive information about desired fonts and font resources. An alternative font can be selected by an interpreter through this descriptive information if the specified one is not available.

6.3.4.2 Font properties

The FONT PROPERTIES element can be used to guide selection of a best fit font if an exact match is not available on a specific device. The font properties which may appear are those in the Minimum Font Description Subset of ISO/IEC 9541-2. Applications may use registered extensions to access additional properties from amongst the ISO/IEC 9541 font properties.

NOTE Registration of additional font properties is done using the registration procedures of the ISO International Register of Graphical Items, as described in ISO/IEC TR9973.

The element allows the importance of each property to be assigned a priority relative to the other properties. In the case that a font named in the FONT LIST is not available, the priorities of the properties instruct the interpreter of the relative importance of the various characteristics of the requested font. In some cases it may not even be necessary to get a particular font, but rather any font with certain characteristics — boldness, presence of serif, etc. The FONT PROPERTIES element enables generators to specify such concepts. The use of this information by interpreters is not standardized.

6.3.4.3 Character set repertoire in graphical text strings

The CHARACTER SET LIST element allows designation of registered character sets for use in the metafile. It also allows designation of private character sets, and in combination with the GLYPH MAPPING element allows designation of collections of glyphs which are registered (according to the procedures of ISO/IEC 10036).

Format effector control characters (NUL, CR, LF, BS, HT, VT, and FF) are permitted in parameters of type String but their interpretation is implementation dependent. Control characters used for character set invocation and designation are permitted according to the setting of CHARACTER CODING ANNOUNCER. The other control codes from the C0 and C1 sets are reserved for future standardization.

NOTE SI and SO are only defined and usable with 7-bit coding.

The mechanisms described in this section for character set definition, and the mechanisms for accessing the character sets (CHARACTER CODING ANNOUNCER, CHARACTER SET INDEX, and ALTERNATE CHARACTER SET INDEX) apply only to string parameters (S) of graphical text elements (TEXT, RESTRICTED TEXT, APPEND TEXT, and appropriate GDPs). They do not apply to non-graphical text strings (SF) such as the *metafile identifier* parameter of the BEGIN METAFILE element.

For designation of ISO registered character sets, two pieces of information are specified by the parameters: a character set type, and the tail of its designating sequence by which it is known in the ISO register.

There are five types of character sets: 94-character G-sets, 96-character G-sets, 94-character multibyte G-sets, 96-character multibyte G-sets, and character sets intended to be designated as "complete codes".

94-CHARACTER G-SETS. These character sets are designated by IS0 2022 escape sequences of the form <ESC> <I1> <I>(o) <F>. Here, <I1> is either 2/8, 2/9, 2/10, or 2/11; <I>(o) represents zero or more intermediate characters from column 2 of the code chart; and <F> is a final character from columns 3 through 7 of the code chart. If <F> is from column 3 of the code chart, the character set is a "private" character set. If <F> is from columns 4 through 7 of the code chart, the character set is a "standard" character set in the sense that it and its designating escape sequences are registered in the International Register Of Coded Character Sets To Be Used With Escape Sequences.

For 94-character G-sets, the character set declaration consists of ‘94-character G-set', followed by a string consisting of all characters in the IS0 2022 designating escape sequence except the first two characters, <ESC> <I1>.

For example, the G-set from the U.K.'s national 7-bit character set is registered in the International Register Of Coded Character Sets To Be Used With Escape Sequences. Its designating escape sequences are as follows:

<ESC>	2/8	4/1	{to designate it as G0}
<ESC>	2/9	4/1	{to designate it as G1}
<ESC>	2/10	4/1	{to designate it as G2}
<ESC>	2/11	4/1	{to designate it as G3}

Again, the French character set (1982 version, from the 1982 version of AFNOR NF Z 62-010) is registered in the International Register Of Coded Character Sets To Be Used With Escape Sequences. Its designating escape sequences are as follows:

<ESC>	2/8	6/6	{to designate it as G0}
<ESC>	2/9	6/6	{to designate it as G1}
<ESC>	2/10	6/6	{to designate it as G2}
<ESC>	2/11	6/6	{to designate it as G3}

Therefore, a CHARACTER SET LIST element could specify that the U.K. character set is to be referred to by character set index 1, and the French character set by character set index 2, as follows:

<CHARACTER-SET-LIST: U.K., French>

‘94-character G-set' 4/1
‘94-character G-set' 6/6

96-CHARACTER G-SETS. These character sets are similar to 94-character G-sets, but include the code positions 2/0 and 7/15, which are excluded from 94-character G-sets. Their IS0 2022 designating escape sequences take the form <ESC> <I1> <I>(o) <F>, where the first intermediate character <I1> is either 2/13, 2/14, or 2/15. The remainder of the escape sequence is similar to the escape sequences for 94-character G-sets: zero or more intermediate characters from column 2 of the code chart and a final character from columns 3 through 7 of the code chart.

For 96-character G-sets, the character set declaration consists of ‘96-character G-set', followed by a string consisting of all characters in the IS0 2022 designating escape sequence except the first two characters, <ESC> <I1>.

It is possible for interchanging parties to agree on a private 96-character G-set whose designating escape sequences would end with a character from column 3 of the code chart. For example, the following might be private escape sequences to designate such a G-set:

<ESC>	2/13	3/0	{to designate it as G1}
<ESC>	2/14	3/0	{to designate it as G2}
<ESC>	2/15	3/0	{to designate it as G3}

(96-character G-sets may not be designated as G0 sets.)

For example, the following CHARACTER SET LIST element establishes the U.K. 94-character G-set, the French 94-character G-set, and a private 96-character G-set as the character sets named by character set indexes 1, 2, and 3, respectively:

<CHARACTER-SET-LIST: U.K., French, private 96-character G-set>

‘94-character G-set' 4/1
‘94-character G-set' 6/6
‘96-character G-set' 3/0

94-CHARACTER MULTIBYTE G-SETS. A 94-character multibyte G-set can contain 94 to the Nth power characters, each coded as a sequence of N bytes from columns 2 through 7 of the code chart, not including the bytes 2/0 and 7/15, which are excluded from 94-character G-sets. For example, a 94-character 2-byte G-set can contain 8,836 characters.

The IS0 2022 designating escape sequences for 94-character multibyte G-sets have the following forms:

<ESC>	2/4	2/8 <F>	{to designate it as G0}
<ESC>	2/4	2/9 <F>	{to designate it as G1}
<ESC>	2/4	2/10<F>	{to designate it as G2}
<ESC>	2/4	2/11<F>	{to designate it as G3}

For compatibility with the first version of ISO 2022, when <f> is 4/0, 4/1, or 4/2 the byte 2/8 may be left out of the designating escape sequence as a GO set.

For 94-character multibyte G-sets, the character set declaration consists of ‘94-character multibyte G-set', followed by a string consisting only of the final character in the IS0 2022 designating escape sequence.

For example, a Japanese 2-byte character set of 6802 graphic characters has been registered in the International Register Of Coded Character Sets To Be Used With Escape Sequences, and its designating escape sequences have the form shown above, with the final character <F> being 4/0. Thus, the following CHARACTER SET LIST element could be used to specify that this 2-byte Japanese character set is to be referred to by character set index 1:

<CHARACTER-SET-LIST: Japanese 2-byte character set>

‘94-character multibyte G-set' 4/0

96-CHARACTER MULTIBYTE G-SETS. A 96-character multibyte G-set is similar to a 94-character multibyte G-set except that it can include the bytes 2/0 and 7/15. Thus, a 96-character 2-byte G-set could have 96 times 96 (or 9216) 2-byte character codes.

The ISO 2022 designating escape sequences for 96-character multibyte G-sets have the following forms:

<ESC>	2/4	2/13<F>	{to designate it as G1}
<ESC>	2/4	2/14<F>	{to designate it as G2}
<ESC>	2/4	2/15<F>	{to designate it as G3}

It is not possible to designate a 96-character multibyte G-set as a G0 set.

The character set declaration for a 96-character multibyte G-set consists of ‘96-character multibyte G-set', followed by a string consisting only of the final character <F> in the character set's ISO 2022 designating escape sequence.

At the time of this publication, no 96-character multibyte G-sets have been registered in the International Register of Coded Character Sets To Be Used with Escape Sequences.

CHARACTER SETS INTENDED TO BE DESIGNATED AS COMPLETE CODES. Other character sets may not fit the ISO 2022 "G-set" structure. ISO 2022 provides an escape sequence format for invoking coding systems different from ISO 2022. The complete code escape sequences have the following form

<ESC> 2/5 <I>o <F>

where <I>o means "zero or more characters from column 2 of the code chart", and <F> is a final character from columns 3 through 7 of the code chart. If <F> is from column 3, the coding system is a private code. If <F> is from columns 4 through 7, it is a code for which a designating and invoking escape sequence has been registered in the International Register Of Coded Character Sets To Be Used With Escape Sequences.

The character set declaration for a character set that would be invoked as a coding system different from ISO 2022 consists of ‘complete code' followed by a string consisting only of those characters in the code's ISO 2022 escape sequence which come after the first two characters, <ESC> 2/5.

As well as using a registered complete code, a private code could be used. For example, suppose the interchanging parties have agreed on a private 8-bit code to be invoked by the following escape sequence:

<ESC> 2/5 3/0

The following CHARACTER SET LIST element would declare the French character set to have character set index 1 and that 8-bit private code to have character set index 2:

<CHARACTER-SET-LIST: French, private coding system>

‘94-character G-set' 6/6
‘complete code' 3/0

Information regarding the designation sequence tail parameter is found in the International Register of Coded Character Sets to be Used with Escape Sequences. This register is maintained by the Registration Authority for ISO 2375, which is the European Computer Manufacturers Association (ECMA), Rue du Rhone 114, CH-1204, Geneva, Switzerland.

6.3.4.4 Glyph repertoire

ISO/IEC 10036 specifies a procedure and a registrar (registering authority) for registering typographic glyph collections. There currently is no standard that associates codes (i.e., character codes) with these glyphs. However the registrar — the Association for Font Information Interchange, or AFII — assigns a unique 4-byte integer indentifier with each glyph.

This part of ISO/IEC 8632 defines a means to access these registered glyph collections for use in graphical text strings. The GLYPH MAPPING element associates the AFII 4-byte identifiers with single-byte or multi-byte codes. A set of such codes is defined as a collection, forming a locally defined character set for use within the metafile. The local character set is associated with an index, and within the body of the CGM the normal character set access and switching mechanisms (based upon, and adapted from, ISO 2022) may be used to access the AFII registered glyphs within CGM text strings.

NOTE1 The glyph complement is a property of a font resource in the ISO/IEC 9541 font architecture. When the separate mechanisms of ISO/IEC 8632 for font reference and glyph access are used there is potential for incompatibility between the specifications — the requested glyph complement may not be representable in the requested font.

6.3.4.5 Character set selection in non-graphical text strings

The mechanisms discussed in the preceding sections apply to graphical text strings — text strings which result in display as graphical entities. These are the parameters of TEXT, RESTRICTED TEXT, APPEND TEXT, and possibly GDP elements.

They do not apply to other, non-graphical text strings which occur as metafile parameters. See 7.1**,** type String Fixed (SF). For these strings, character set selection shall be accomplished only through use of embedded ISO 2022 controls, to switch sets within the string.

For the purposes of coding type SF parameters in ISO/IEC 8632 metafiles, 8-bit coding environment shall be used, by default. It is possible using ISO 2022 controls to select other coding environments, such as multi-byte complete codes.

In SF parameters, therefore, by default the two character sets which are currently designated as GO and G1 may be accessed by using respectively the code values 32..127 and 160..255.

BEGIN METAFILE causes ISO 8859-1 Left Hand Side to be designated as the G0 set and ISO 8859-1, Right Hand Side of Latin Alphabet Nr. 1 to be designated as the G1 set.

EXAMPLE 1 — If the following 4-octet ISO 2022 sequence were to occur as the first 4 octets of the ‘metafile id' parameter of the BEGIN METAFILE element, ESC 02/05 02/15 04/09,

then the character set for subsequent type SF character data would be the variable length multi-byte UTF-8 transformation of Unicode (ISO 10646).

EXAMPLE 2 — The ISO 2022 designating sequence

ESC 2/8 4/2, for example, would designate

ISO 646, U.S. National Character Set (ASCII) as the G0 set (the default set).

EXAMPLE 3 — Similarly, the ISO 2022 designating sequence

ESC 2/13 4/2

would designate ASCII as the G1 set, superceding the default G1 set (ISO 8859-1) — character codes following an SI would then refer to ASCII, not ISO 8859-1.

Format effector control characters (NUL, CR, LF, BS, HT, VT, and FF) are permitted in a parameters of type String Fixed but their interpretation is implementation dependent. Control characters used for character set invocation and designation (SI, SO, ESC, SS2, and SS3) are permitted according to the setting of CHARACTER CODING ANNOUNCER. The other control codes from the C0 and C1 sets are reserved for future standardization.

6.3.4.6 Revision of character sets

ISO 2022:1986 allows for the revision of character sets. It states: ESC 2/6 F, when used, shall immediately precede a designating escape sequence and indicate a revision of a registered set. F will specify the revision number 1 to 63 by taking values 4/0 to 7/14 respectively. Revisions are only permitted to add a character or characters to a set and shall be submitted to the Registration Authority as required by ISO 2375, pointing out that the submission is a revision of a registered set.

6.3.5 Picture directory

The PICTURE DIRECTORY element contains locations of the pictures in a metafile. The picture directory contains a list of picture identifiers and their locations relative to the BEGIN METAFILE element. The picture directory may optionally contain the location of the APPLICATION STRUCTURE DIRECTORY element in each picture relative to the BEGIN METAFILE element. An interpreter wishing to randomly address a picture in a metafile with a picture directory need only interpret the Metafile Descriptor and skip to the locations of the picture specified in the PICTURE DIRECTORY. The next element read will be the desired BEGIN PICTURE.

6.4 Picture descriptor elements

Picture Descriptor elements include elements to declare the parameter modes of other elements for an entire picture, to configure that portion of coordinate space that is of interest in the picture, and to set the colour to which the view surface is cleared at the start of the picture.These elements are:

SCALING MODE	LINE REPRESENTATION
COLOUR SELECTION MODE	MARKER REPRESENTATION
LINE WIDTH SPECIFICATION MODE	TEXT REPRESENTATION
MARKER SIZE SPECIFICATION MODE	FILL REPRESENTATION
EDGE WIDTH SPECIFICATION MODE	EDGE REPRESENTATION
VDC EXTENT	INTERIOR STYLE SPECIFICATION MODE
BACKGROUND COLOUR	LINE AND EDGE TYPE DEFINITION
DEVICE VIEWPORT	HATCH STYLE DEFINITION
DEVICE VIEWPORT SPECIFICATION MODE	GEOMETRIC PATTERN DEFINITION
DEVICE VIEWPORT MAPPING	APPLICATION STRUCTURE DIRECTORY

In Version 1 metafiles, any Picture Descriptor elements within a picture shall appear after the BEGIN PICTURE element and before the BEGIN PICTURE BODY element. Escape and external elements are permitted in the Picture Descriptor.

In Version 2, Version 3 and Version 4 metafiles, some of the Picture Descriptor elements may appear within the picture body. Table 8a indicates which Picture Descriptor elements may appear and where they may appear for each metafile version.

6.4.1 Scaling mode

VDC space may be either an abstract space, which may be mapped to an arbitrary size on a physical device, or a metric space, which is intended to be mapped to a particular size. Selection of the mode to be used can be made on a picture-by-picture basis by means of the SCALING MODE element. The scaling mode element provides a flag to select abstract space or metric space, and a scale factor which specifies the number of millimeters per VDC unit when metric space is selected.

6.4.2 Colour selection mode

COLOUR SELECTION MODE selects either indexed or direct colour specification and is described further under colour attributes, 6.7.6. For Version 1 metafiles, the selection is for the whole picture.

6.4.3 Specification modes

Line width, marker size, interiors of filled-area elements, and edge width may be specified in more than one way. The width of lines, for example, may be specified as either a measure in VDC units, a scale factor to be applied to a device-dependent nominal line width at interpretation time, a fraction of the device view surface, or a measure in millimetres. For each attribute element having such multiple modes, there is an associated control element that defines the mode of the parameter of the attribute element.

For each of the four values -- scaled, absolute, mm, fractional -- of the line width specification mode, there is an associated current value of the line width attribute in that mode. This current value is defined either by the last occurrence of a LINE WIDTH attribute element in that mode, or by the default line width for that mode. The same is true for markers, edges, and filled-area interiors -- there are conceptually four current values of each of these attributes, one associated with the each of the four values of the appropriate specification mode element.

6.4.4 VDC extent

There is a metafile element to define the VDC extent. The extent is set with the VDC EXTENT element by specifying the addresses (in VDC) of the lower-left corner and the upper-right corner of this extent as seen by the viewer of the picture. Specification of values outside the VDC extent is permitted in CGM elements.

It is intended that the visible portion of an image be contained within the VDC extent. It thus provides a frame for the region of interest in a picture. The values of the coordinates for either dimension may be either increasing or decreasing from the lower-left to the upper-right corner. For example, for devices with an upper-left origin, a picture may be described in coordinates that map directly to the device but still may be displayed correctly on a device with a lower-left origin. Figure 1 illustrates these concepts.

The VDC extent thus establishes the sense and orientation of VDC space (that is, the directions of the positive x (+x) and positive y (+y) axes, and whether the +y axis is 90° clockwise or 90° counterclockwise from the +x axis). In particular, VDC EXTENT establishes the direction of positive and negative angles as follows: positive 90° is defined to be the right angle from the positive x-axis to the positive y-axis (see figure 1).

Note that some attributes such as text attributes (for example, the directions of the ‘up' and ‘base' component vectors of CHARACTER ORIENTATION, and therefore the meaning of the enumerative values ‘right', ‘left', ‘up', ‘down') are intimately bound to these definitions.

The default state of the extent is specified in clause 8 and can be changed in the METAFILE DEFAULTS REPLACEMENT element in the Metafile Descriptor. VDC EXTENT returns to this default state at the beginning of each picture. MAXIMUM VDC EXTENT defines an extent which bounds the VDC extent values which may be found in the metafile. It may be, but need not be, a closest bound in the sense that it exactly equals the union of the extent rectangles in the metafile. This element may be used, for example, to map integer virtual device coordinates of the metafile to a unit square in a normalized device space.

6.4.5 CGM tailoring

The ability to specify the VDC range and the VDC extent provides the flexibility to configure the metafile addressability in any way desired. It can be configured as an abstract, normalized address range for maximum device independence. It can also be configured to mimic the addressability of a particular target device in order to take advantage of particular device characteristics. The address range of such a device-specific metafile is just another normalized address range with the normalization limits inherent in the VDC-customizing element; therefore, device independence is maintained.

Such tailoring of the coordinates in a metafile can eliminate the need for transformation of coordinates at metafile interpretation time for the target device. The ability to specify the VDC extent thus allows for the exact registration of coordinates in a metafile with addressable points on the target graphics device.

The use of VDC EXTENT to directly encode world coordinates of large dynamic range and very small granularity will likely result in performance penalties at metafile interpretation time, and may result in decreased portability if such VDC extents exceed those compatible with less capable metafile interpreters.

In addition to VDC tailoring, a metafile generator can limit or tailor the functional content of a metafile to accomodate particular devices or applications, and announce such functional tailoring through the use of METAFILE ELEMENT LIST.

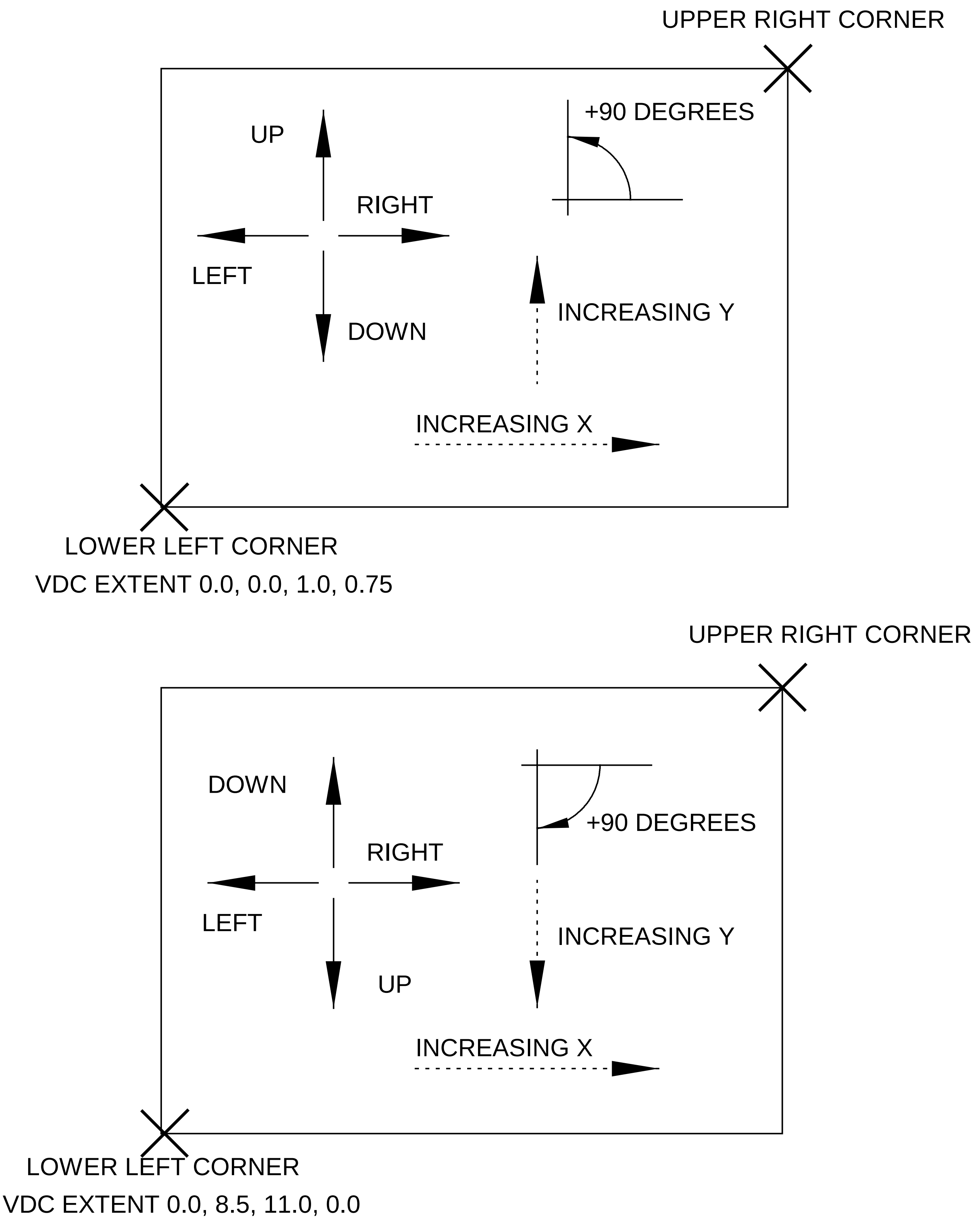


Figure 1 — VDC EXTENT establishes the direction of positive and negative angles

6.4.6 Background colour

Each picture defines a graphical image that is independent of the other images in a metafile.

The background colour of the image may be specified by the BACKGROUND COLOUR Picture Descriptor element. It may also be specified by the COLOUR TABLE definition of colour index 0 in the METAFILE DEFAULTS REPLACEMENT (metafiles of all versions), or by the COLOUR TABLE definition of colour index 0 in the Picture Descriptor (metafiles of Versions 2 and higher). Definition of colour index 0 within the picture body has no effect on the background colour, only on subsequent primitives displayed with colour index 0 — the background colour is determined and bound at the occurrence of BEGIN PICTURE BODY.

6.4.7 Device viewport control

The device viewport specifies the region of the device display surface into which the VDC extent is to be mapped on interpretation. VDC-to-Device mapping is determinied by the VDC extent, device viewport, and device viewport mapping.

The position of the device viewport is specified in one of three coordinate systems selected by the DEVICE VIEWPORT SPECIFICATION MODE element:

— by fraction [0.0 to 1.0] of the available display surface, which allows reasonable placement and relative sizing of the viewport;

— in millimetres times a scale factor, which allows absolute sizing of images;

— in physical device coordinates.

The device viewport is specified in terms of two points on the device display surface at diagonally opposite corners of the rectangle. Mirroring or 180° rotation of the image may be achieved by specifying the corners in some way other than the first as below and to the left of the second.

The DEVICE VIEWPORT MAPPING element may be used to force isotropic mapping even if the specified VDC extent and device viewport would not otherwise have led to one. In such a case, the VDC extent is mapped on to a subset of the specified device viewport on interpretation. This subset is defined by shrinking either the vertical or horizontal dimension of the specified viewport as needed to reach the required aspect ratio. This smaller "effective viewport" is then used to define the coordinate mapping from VDC to the device's coordinates. The placement of the effective viewport rectangle within the original one can be specified. This placement can be one of ‘left', ‘right' or ‘centred' when the shrinking is horizontal, and ‘top', ‘bottom' or ‘centred' when it is vertical. These meanings are relative to the display surface of the device.

The VDC-to-Device mapping maps the first point specifying the VDC extent on to the corner of the effective viewport corresponding to the first point specifying the device viewport, and similarly for the second point. The mapping is linear in each dimension, but is not necessarily isotropic (for example, a circle in VDC may not appear as a circle to the viewer).

Both the way VDC space is oriented relative to the display surface and the way the effective viewport is placed on the physical device may lead to mirroring and 180° rotation.

The behaviour of primitives and attributes with significance in VDC space under transformations is further described in 6.6.

If both device viewport and scaling mode appear in the same metafile then the last specified is used. If neither appears then the default values for device viewport take precedence.

6.4.8 Representations

The elements LINE REPRESENTATION, MARKER REPRESENTATION, TEXT REPRESENTATION, FILL REPRESENTATION and EDGE REPRESENTATION are used to set all of the attribute values in a bundle table entry at the same time. The attributes that may be bundled are described in 6.7.

6.4.9 Definable attributes

A precise definition of line and edge types as well as hatch styles may be accomplished as described in 6.7. Moreover, geometric patterns may be defined (in addition to patterns which are arrays of colours confined to parallellogram-like cells) using the segment definition mechanism. See 6.7.

6.4.10 Application structure directory

The APPLICATION STRUCTURE DIRECTORY element contains the locations of the various application structures in a picture. The application structure directory contains a list of application structure identifiers and their locations relative to the BEGIN PICTURE element. APSs within a metafile containing multiple pictures can be accessed directly in two steps. First, the picture directory is used to locate a picture and possibly the application structure directory for the picture. Second, the APSs in the picture are located using the application structure directory.

6.5 Control elements

Control elements specify address space, clipping boundaries, and format descriptions of the CGM elements. Control of some of these format descriptions may be accomplished by Metafile Descriptor elements, while control of others is accomplished by control elements, which may appear in the picture bodies in the metafile. Those items in the former category are fixed for a given metafile, while those in the latter category are changeable; that is, they may change within a picture. Some of the control elements may appear in the Picture Descriptor if this is permitted by the formal grammar for the metafile version. See Table 8a.

6.5.1 VDC space and range

The graphical primitive elements of a metafile define virtual images. The coordinates of these elements (that is, the addresses of points in the virtual image) are absolute two-dimensional Virtual Device Coordinates (VDC). VDC space is a two-dimensional coordinate space of infinite precision and infinite extent. Only a subset of VDC space, the VDC range, is realizable. The VDC range comprises all coordinates representable in the format specified by the declared VDC TYPE and (depending on the type) the VDC INTEGER PRECISION or VDC REAL PRECISION.

The VDC range cannot be set directly; it is completely determined by VDC TYPE and either VDC INTEGER PRECISION or VDC REAL PRECISION elements in the metafile. These elements are controllable, some by dynamic elements in the metafile body and some by static elements in the Metafile Descriptor. Note that the VDC range thus defined (a rectangular subregion of the VDC space) does not enclose a continuum of values, but has a distinct granularity. Regardless of the aspect ratio of the VDC range and the granularity within the range, it is implicit that one VDC unit in the x-direction represents the same distance as one VDC unit in the y-direction in VDC space.

6.5.2 Clipping

In order to defer clipping of graphical primitive elements (particularly, expandable elements such as CIRCLE, CIRCULAR ARC 3 POINT, TEXT, etc.) until metafile interpretation time, a clipping control feature is provided in the CGM. Clipping control is achieved by defining CLIP RECTANGLE in VDC space. Whether clipping to the limits of CLIP RECTANGLE actually occurs at metafile interpretation time is controlled by the CLIP INDICATOR element that sets the mode of the metafile to ‘on' or ‘off'.

In Version 3 and Version 4 metafiles, primitives may also be clipped against more general regions as defined by BEGIN PROTECTION REGION and END PROTECTION REGION, and as controlled by PROTECTION REGION INDICATOR (see 6.5.4). Clipping effects in the remainder of this section are described in terms of the basic CLIP RECTANGLE capability, but they apply equally to general clip regions.

There are three different clipping modes for lines, markers and edges. The required clipping mode is recorded in the metafile with the elements: LINE CLIPPING MODE, MARKER CLIPPING MODE, and EDGE CLIPPING MODE. When the CLIP INDICATOR associated with a graphical primitive is ‘on', only those parts of a graphical primitive that are considered inside the effective clipping region are rendered on interpretation. The object clipping modes allow precise specification as to how clipping is applied to primitives on interpretation.

Clipping may be either ‘locus', ‘shape' or ‘locus then shape'. Conceptually, a locus is a mathematical object like a point or line segment, while a shape is an area in 2-dimensional space. Loci are 0-, 1- or 2-dimensional subsets of real-valued 2-space. For markers and text, they are points. For lines, they are the individual line segments or portions of arcs. The locus of an area is the shape and the boundary. Shapes reflect the realization of geometric attributes and are generally 2-dimensional subsets of real-valued 2-space.

'Locus' clipping is applied for each portion of a graphical object based on its mathematical location and is independent of the area it will occupy after rendering. For example, no portion of a line segment is rendered if the ideal mathematical line lies outside the effective clipping region (even if its line width would carry some portion of the rendering of it into the clipping rectangle); no portion of a marker is rendered if its location lies outside the clipping rectangle.

If 'locus' clipping is used, the rendering is applied to the locus of the graphical object after clipping. The resulting rendered shape areas may therefore extend outside the effective clipping region.

'Shape' clipping is applied after the abstract rendering of shape in device coordinate space. The 2-dimensional point set associated with the graphical object is intersected with the effective clipping region, which has been transformed to device coordinate space.

'Locus then shape' clipping allows the specification that both 'locus' and 'shape' clipping be applied to graphical objects as described above. In this case however, the rendered shape will not extend outside the effective clipping region. A thick line whose locus is outside the clip rectangle will not have any portion visible even if its line width would carry some portion of the rendering inside the clip rectangle.

Figure 2 shows some examples of the effect of the clipping modes.

When a width or size specification mode is 'scaled', 'fractional', or 'mm', the rendering of shape proceeds in device coordinate space after application of the VDC-to-Device mapping.

When a width or size specification mode is 'absolute', the rendering of shape proceeds, in VDC space before application of the copy transformation and before application of the segment transformation (if the primitive is in a copied segment), and before the VDC-to-Device mapping.

Fill and text primitives do not have associated object clipping modes (though the edge of a fill primitive and the boundary edges of a closed figure do). Clipping for fill primitives is always consistent with 'shape' clipping (see 6.6.4). For text primitives, the type of clipping is determined by the associated text precision:

— For 'string' precision text, clipping proceeds, on a per string basis, in a manner consistent with 'locus' clipping.

— For 'character' precision text, clipping proceeds, on a per character basis, in a manner consistent with 'locus' clipping.

— For 'stroke' precision text, the clipping always proceeds in a manner consistent with 'shape' clipping.

NOTE It is valid for an interpreter to perform 'shape' clipping of text in all cases, i.e., 'stroke' text precision is a valid realization of 'character' and 'string' text precision (and 'character' text precision is a valid realization of 'string' text precision).

Clip rectangles applied to graphical primitive elements within segments may be subject to transformations in VDC space. Intersection of clip rectangles (untransformed or transformed) may result in polygonal clipping boundaries (see 6.10.5)

6.5.3 Save and restore primitive context

Two elements, SAVE PRIMITIVE CONTEXT and RESTORE PRIMITIVE CONTEXT, are provided to save and restore a context; that is, attributes and control elements as collections. This capability allows a list of attributes and control elements (see 7.5.11) to be stored in the metafile which can be referenced by name at a later point in the metafile. This capability can be used to save and restore attributes and control elements in conjunction with opening and closing segments.

The values for attributes controlled by specification or selection modes are saved in the mode in which they were last specified along with the value of the corresponding mode. In restoring a context, the current specification and selection modes are not changed.

6.5.4 Compound clipping and shielding

The clipping and shielding elements consist of BEGIN PROTECTION REGION, END PROTECTION REGION, and PROTECTION REGION INDICATOR. The BEGIN PROTECTION REGION and END PROTECTION REGION elements are delimiter elements, and the PROTECTION REGION INDICATOR element is a control element.

Protection regions are identified by an index. Protection regions are constructed by the same primitive elements as closed figures. The interior of a given protection region is defined in the same way as the interior of a closed figure. Regions which are constructed by line elements are closed by NEW REGION, END PROTECTION REGION, or any filled area element. If the endpoints and beginning points of subsequent line elements are not identical they are implicitly connected by a straight line.

If a protection region is used within a segment (i.e., the PROTECTION REGION INDICATOR has the value ‘clip' or ‘shield' for that region inside the segment), it behaves as do clip rectangles with respect to transformations — it transforms by the copy transformation which is associated with the COPY SEGMENT element. Protection regions used in segments are also affected by the segment transformations of the segment.

The effective protection region for clipping and/or shielding, when protection regions are defined, or referenced, within segments, is determined by CLIP INHERITANCE in the same way as for the simpler rectangle clipping (see 6.10.5).

NOTE Inheritance of regions is complicated by the fact that, conceptually, the entire list of regions together with their indicators must be subject to the inheritance mechanism, down to the point in the segment hierarchy where reference to a region is made or the region must be applied for clipping or shielding (in the case of the ‘intersection' value of CLIP INHERITANCE).

If a protection region is defined within a segment, then its definition persists after the END SEGMENT terminating that segment definition. This is exactly as for clip rectangles, and primitive attributes. Thus the definition and activation of a region within a local segment in a picture body potentially has effect in the picture body following the segment. If a segment is copied into the picture body with a COPY SEGMENT function, then region definitions within the segment do not affect primitives following, again exactly as clip rectangle.

A protection region is associated with an index at definition time. If a region is defined with an index that is already in use, then the old definition is deleted. The associated protection region indicator for the newly defined region takes the initial value ‘off'.

When a protection region is set for clipping, only the portions of the graphical elements inside or on the boundary of the protection region are drawn. When a protection region is set for shielding, only the portions of the graphical elements outside the protection region and its boundary are drawn.

Several clip and shield regions may be in effect simultaneously. In this case, only portions of the graphical elements inside or on the boundary of the intersection of all individual clip regions and outside the union of all individual shield regions are drawn. See figure 3.

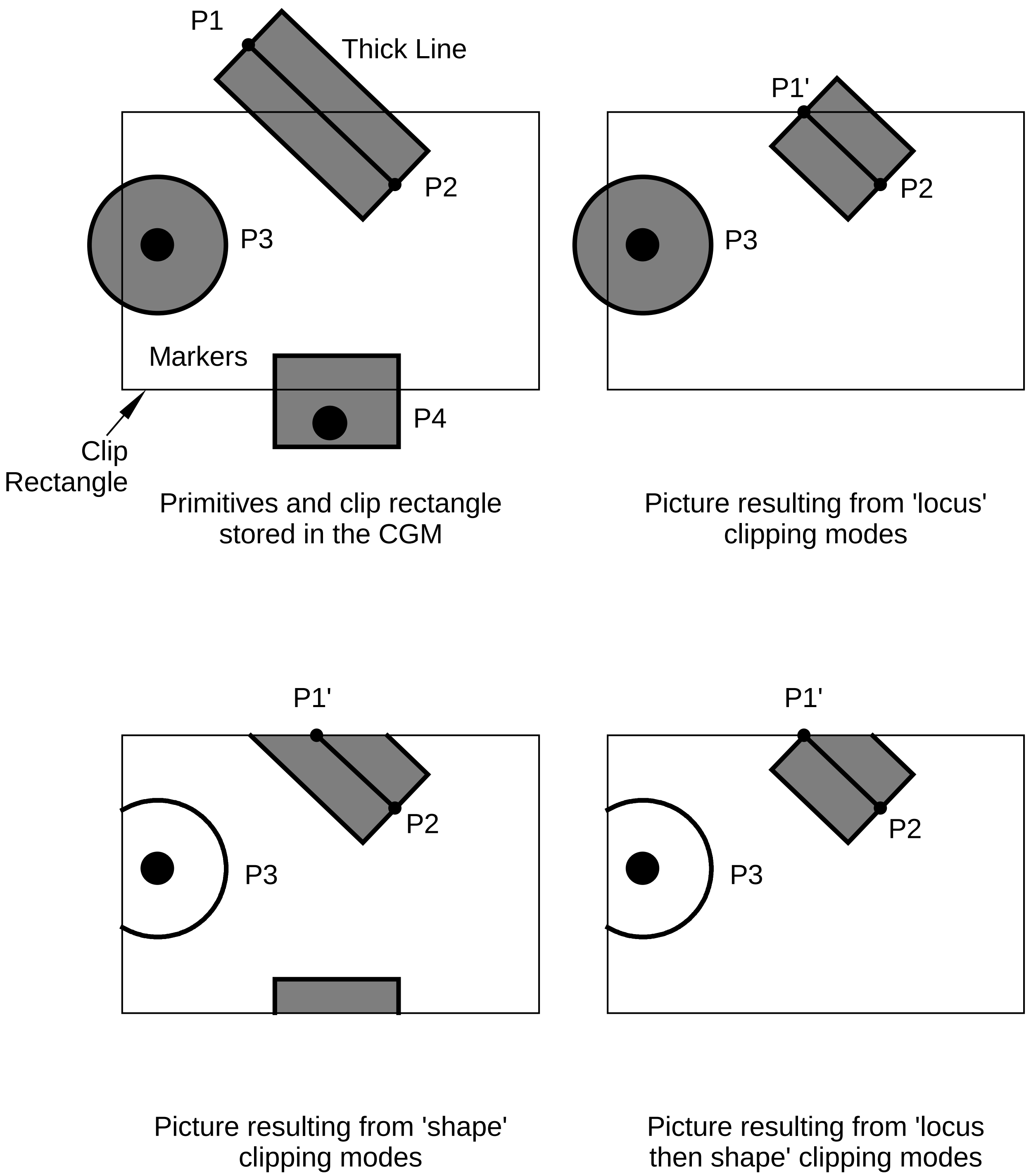


Figure 2 — Clipping modes

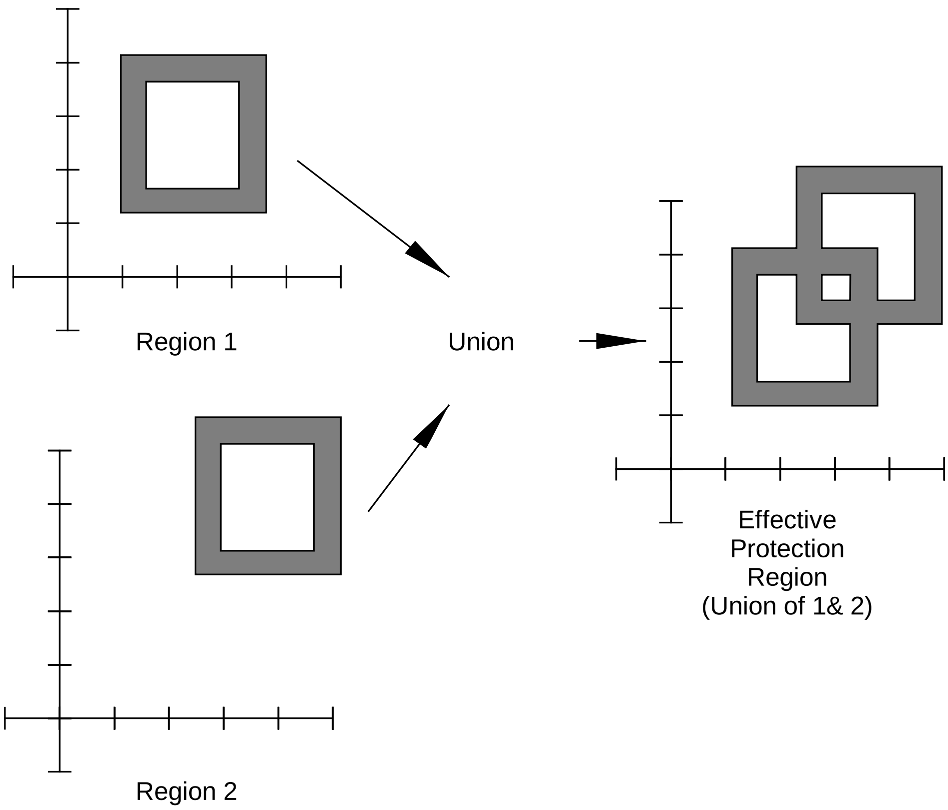


Figure 3 — Combining protection regions (hatching denotes the region)

6.5.5 Generalized text path

The GENERALIZED TEXT PATH MODE element selects the method for placing the text along the text path. When the mode is ‘off', the text is displayed along the text path defined by the CHARACTER ORIENTATION and TEXT PATH elements.

When GENERALIZED TEXT PATH MODE is ‘non-tangential', the characters are drawn along the last compound text path (defined by the BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements), but the character orientation vectors are not rotated relative to the text path. When GENERALIZED TEXT PATH MODE is ‘axis-tangential', the characters are positioned along the last compound text path (defined by the BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements) and the character orientation vectors for each character are rotated by an amount equal to the angle of the tangent to the text path at the character position.

Illustrations of GENERALIZED TEXT PATH MODE are shown in figure 21 through figure 26.

6.5.6 Mitre limit

The control elements include an element for controlling how the joins of lines and filled-area edges are drawn. Line or edge joins may be rendered with mitre join style, as determined by the appropriate line and edge attribute elements. The mitre join is formed by projecting the outer edges of the lines or edges at the corner until the projections meet at a point. When mitre joins are being rendered, there is the possibility of the mitre projecting very far if the line segments meet at a very sharp angle at the vertex. The MITRE LIMIT provides a means of specifying that long mitres are to be truncated at some point to form a flat bevel. *Mitre length* is defined to be the distance from the point at which the inside edges of adjoining line segments meet to the point at which the outside edges meet. The parameter of MITRE LIMIT is a single real number. If the mitre length divided by the line width or edge width exceeds the value of the parameter of the MITRE LIMIT element, then the mitre join is truncated at that limit. However, if the truncation would result in a flat bevel within the triangular notch of the corresponding bevel join style, then the bevel join style is used.

6.5.7 Transparent cell colour

The control elements include an element, TRANSPARENT CELL COLOUR, which allows certain parts of CELL ARRAY elements, TILE and BITONAL TILE elements, and the filling patterns defined by PATTERN TABLE elements to be designated as transparent.

The usual way of rendering these elements is to draw each cell with the colour designated by its associated cell colour specifier. TRANSPARENT CELL COLOUR allows a given value of a cell colour specifier (indexed or direct) to be defined to be transparent. Any cell whose cell colour specifier matches this value is not drawn when the primitive is rendered.

In order to invoke the transparency effect in an element, the cell colour specifier shall have the transparent designation at the time of occurrence of the affected element in the metafile. If a cell in a pattern definition is to be designated as transparent, then the colour value of the associated cell colour specifier shall be defined as transparent prior to the occurrence in the metafile of the PATTERN TABLE element which defines the particular pattern. That is, the TRANSPARENT CELL COLOUR element only affects elements which occur after it in the metafile and has no effect on previously occurring elements or previously drawn primitives.

The rendering of any element affected by TRANSPARENT CELL COLOUR shall not involve more than one distinct value of cell colour specifier simultaneously being designated as transparent.

NOTE It is recommended that metafile generators limit themselves to using a single value of the cell colour specifier as their transparent cell value.

6.6 Graphical primitive elements

Graphical primitive elements are those elements that describe the visual components of a picture. Their coordinate arguments, if any, are specified in VDC units. The CGM provides the graphical primitive elements

POLYLINE
DISJOINT POLYLINE
POLYMARKER
TEXT
CIRCULAR ARC CENTRE CLOSE
ELLIPSE
ELLIPTICAL ARC
ELLIPTICAL ARC CLOSE

RESTRICTED TEXT	CIRCULAR ARC CENTRE REVERSED
APPEND TEXT	CONNECTING EDGE
POLYGON	HYPERBOLIC ARC
POLYGON SET	PARABOLIC ARC
CELL ARRAY	NON-UNIFORM B-SPLINE
GENERALIZED DRAWING PRIMITIVE (GDP)	NON-UNIFORM RATIONAL B-SPLINE
RECTANGLE	POLYBEZIER
CIRCLE	POLYSYMBOL
CIRCULAR ARC 3 POINT	BITONAL TILE
CIRCULAR ARC 3 POINT CLOSE	TILE
CIRCULAR ARC CENTRE	

The metafile supports access to special geometric output capabilities of devices and workstations through the GDP. The GDP has a list of points in VDC as a parameter. It is thus well suited for non-standardized output primitives, which have position, shape, extent, etc., whereas ESCAPE is better suited for non-standardized device control functions.

The formal definition of the CGM describes graphical primitive elements which are positionally independent by virtue of containing complete explicit positional information within each element definition.

The TEXT, RESTRICTED TEXT, and APPEND TEXT elements and related text attribute elements are defined in the current VDC space. Thus, they are affected by changes to the Virtual Device Coordinate format.

The following types or categories of graphical primitive elements are defined for the CGM: line elements, marker element, text elements, filled-area elements, cell elements, and symbol elements.

The line elements are

POLYLINE	CONNECTING EDGE
DISJOINT POLYLINE	HYPERBOLIC ARC
CIRCULAR ARC 3 POINT	PARABOLIC ARC
CIRCULAR ARC CENTRE	NON-UNIFORM B-SPLINE
ELLIPTICAL ARC	NON-UNIFORM RATIONAL B-SPLINE
CIRCULAR ARC CENTRE REVERSED	POLYBEZIER

The marker element is

POLYMARKER

The text elements are

TEXT
RESTRICTED TEXT
APPEND TEXT

The filled-area elements are

POLYGON	CIRCULAR ARC 3 POINT CLOSE
POLYGON SET	CIRCULAR ARC CENTRE CLOSE
RECTANGLE	ELLIPSE
CIRCLE	ELLIPTICAL ARC CLOSE

The cell elements consist of one cell array element and two tile array elements:

CELL ARRAY
BITONAL TILE
TILE

The symbol element is

POLYSYMBOL

In addition to these classes of elements, the GENERALIZED DRAWING PRIMITIVE (GDP) is a graphical primitive element that may be be used to access device (or implementation) specific graphical primitives that are not accessed by the standardized elements. In addition to the graphical primitive elements listed above, this part of ISO/IEC 8632 defines elements permitting the definition of 'compound primitives' from instances of one or several of the other graphical primitives. The following classes of compound primitives are defined: 'compound text', 'closed figure' and 'compound line'. The elements that may be used to specify compound primitives are listed in table 1.

Table 1 — Contributing primitives to compound primitives

Compound Primitive	**First Element**	**Primitives Included**	**Other Elements**	**Final Element**
Compound Text	TEXT$_{(1)}$, RESTRICTED TEXT$_{(1)}$	APPEND TEXT$_{(2)}$		APPEND TEXT$_{(3)}$
Closed Figure	BEGIN FIGURE	Line Primitives$_{(4)}$, Fill Primitives$_{(4)}$, GDP$_{(5)}$	NEW REGION	END FIGURE
Compound Line	BEGIN COMPOUND LINE	Line primitives, GDP$_{(5)}$		END COMPOUND LINE
Tile Array	BEGIN TILE ARRAY	BITONAL TILE, TILE		END TILE ARRAY

NOTE 1 The final/not final flag is 'not final'; the primitive defines the reference point of the entire compound text primitive; the text of the primitive is accumulated.

NOTE 2 The final/not final flag is 'not final'.

NOTE 3 The final/not final flag is 'final'; the text of the primitive is accumulated before the compound primitive is closed.

NOTE 4 All primitives of the identified classes may be included.

NOTE 5 Whether and how a GDP may contribute to closed figure or compound line, and whether or how it specifies that the figure open state or compound path state be opened, maintained or closed, is specified with the definition of the GDP in the International Register of Graphical Items.

With the exception of tile array, graphical primitives and compound primitive elements may be subject to transformation in VDC space (segment and copy transformation, see 6.10.4.2 and 6.10.5). Such a transformation may change the shape of some primitives. If there is a skew, a primitive initially specified as a rectangle may become a parallelogram. If there is an anisotropic scaling, a primitive initially specified as a circle may become an ellipse. Note that the shape of markers is not affected by such transformations. Anisotropic transformation will change the angle at which non-parallel lines intersect; isotropic transformation will preserve the angle at which non-parallel lines intersect.

6.6.1 Line elements

6.6.1.1 Description

There are two general line elements — POLYLINE and DISJOINT POLYLINE — as well as curve elements that define conic arcs (circular, elliptical, parabolic, and hyperbolic arcs) and elements that define spline curves (B-splines and Beziers). There is also a compound line element, whose definition is composed of these individual line elements.

POLYLINE	generates a set of connected lines as defined by a list of points, starting with the first, drawing a line through each successive point, ending at the last point.
DISJOINT POLYLINE	generates a set of unconnected lines as defined by a list of point pairs, drawing from the first to the second, the third to the fourth, etc..

CIRCULAR ARC xxx	generates a single circular arc; two parameterizations of the arc are possible; these are described in 7.6.13 and 7.6.15. A reverse direction arc can also be specified; see 7.6.20.
ELLIPTICAL ARC	generates a single elliptical arc; the parameterization of the arc is described in 7.6.18.
CONNECTING EDGE	a line segment connecting the last point of the preceding line element to the next point is generated during the construction of a closed figure. The next point is either the first point of the next line element or the current closure point.
HYPERBOLIC ARC	generates a hyperbolic arc; the parameterization is described in 7.6.22, and the principles underlying the transformable parameterization are described in 6.6.8.
PARABOLIC ARC	generates a parabolic arc; the parameterization is described in 7.6.23, and the principles underlying the transformable parameterization are described in 6.6.9.
NON-UNIFORM B-SPLINE	generates a Non-Uniform B-Spline curve; the parameterization is described in 7.6.24, and the principles underlying the definition of the element are described in 6.6.10.1.
NON-UNIFORM RATIONAL B-SPLINE	generates a Non-Uniform Rational B-Spline (NURBS) curve; the parameterization is described in 7.6.25, and the principle underlying the definition of the element are described in 6.6.10.1.
POLYBEZIER	generates a sequence of one or more cubic Bezier curves; the parameterization is described in 7.6.26 and the principles underlying the definition of the element are described in 6.6.10.2.

6.6.1.2 Compound line

The BEGIN COMPOUND LINE and END COMPOUND LINE elements delimit a compound line. These elements permit the definition of a line that consists of a number of distinct elements, such as straight lines and arcs, which is treated as if it were a single line element. Thus, for example, line style would apply without change or interruption through the end of a straight line segment and into a following arc segment. Likewise, the ends of the various component elements of the compound line are not considered as line ends but rather as line joints. Line attributes shall not change within a compound line. If two line segments which are adjacent in the definition are not contiguous, i.e., the end point of the first one does not coincide with the first point of the next one, then the path includes the straight line segment joining these two points.

6.6.1.3 Attributes

The appearance of all line elements is controlled by the line attributes, the LINE BUNDLE INDEX, and those ASPECT SOURCE FLAGs which are associated with the line attributes that may be bundled. These are described in 6.7.1.

6.6.1.4 Usage of line elements

POLYLINE is the most general of the primitives. DISJOINT POLYLINE is intended for situations where the alternative would be a large number of 2-point POLYLINE elements. The conic arc primitives (circular, elliptical, hyperbolic, and parabolic) and spline primitives (Polybezier, Non-uniform B-splines, and NURBS) provide data compression by comparison with POLYLINE and allow the arcs to be described without knowledge of the resolution of the final viewing surface.

6.6.1.5 Clipping of line elements

In Version 2, Version 3 and Version 4 metafiles, line clipping is controlled by the LINE CLIPPING MODE element, which can have one of the following values: 'locus', 'shape', or 'locus then shape'. However, clipping applies only if the CLIP INDICATOR is 'on'.

For 'locus' clipping, the mathematical locus of the line is clipped at the intersection with the clip rectangle before shape rendering is applied. Hence, part of the shape of a clipped line may appear outside the clip rectangle.

For 'shape' clipping, the shape of the rendered line is clipped to the intersection with the clip rectangle; that is, nothing is drawn outside the clip rectangle. A portion of a widened line may appear inside the clip rectangle even though the mathematical locus of the line itself may be entirely outside the clip rectangle.

For 'locus then shape' clipping, the mathematical locus of the line is clipped, as with locus clipping, and then subsequently the rendered shape of the clipped locus is again clipped. Note that, since the mathematical locus of the line may have been clipped as a result of locus clipping, subsequent shape rendering and clipping may produce a different appearance of a line from either of the other two clipping modes.

6.6.1.6 Transformation of line elements

If the LINE WIDTH SPECIFICATION MODE has the value 'absolute', then all line aspects — line width, line cap, line join, line dash and gap lengths — are subject to the VDC-to-Device mapping (see 6.4.7) as well as to both segment and copy transformation (see 6.10.4.2 and 6.10.5). Note that the entire locus of an arc is subject to these transformations. In the case of an anisotropic mapping or transformation, the rendered width of the line will change with the direction of the line segment.

If the line width is specified in mode 'scaled', 'fractional', or 'mm', it is not affected by any transformations.

6.6.2 Marker elements

6.6.2.1 Description

There is a single marker element.

POLYMARKER	generates markers of a specific type at each of a list of points.

6.6.2.2 Attributes

The appearance of the markers is controlled by the marker attributes, the MARKER BUNDLE INDEX and the ASPECT SOURCE FLAGs associated with the marker attributes that may be bundled. These are described in 6.7.2.

6.6.2.3 Clipping of the marker element

The following discussion applies to Version 1 metafiles. Markers conceptually indicate the location of their specifying points. Therefore, if the value of CLIP INDICATOR is 'on', the marker is visible if, and only if, its specifying point is within the rectangle specified by CLIP RECTANGLE. If its specifying point is inside the clip rectangle but part of the marker lies outside, the manner in which the marker is clipped (or not clipped) is not standardized for Version 1 metafiles.

NOTE When it is required, in Version 1 metafiles, to make visible those parts of a marker that are inside the clip rectangle, when the position of the marker is outside the rectangle, then the most appropriate primitive element is the TEXT element, used with a single-character string argument. Three PRECISIONs are available to control the precision with which TEXT elements are clipped. STROKE precision requires clipping even within the body of a text character. Centring of the text character at the specifying position may be achieved with the TEXT ALIGNMENT element.

In Version 2, Version 3 and Version 4 metafiles, marker clipping is controlled by the MARKER CLIPPING MODE element, which can have one of the following values: 'locus', 'shape' or 'locus then shape'. However, clipping applies only if the value of CLIP INDICATOR is 'on'.

For 'locus' clipping, the specifying points of each marker are clipped at the intersection with the clip rectangle before shape rendering is applied. The marker is only visible if its specifying point is within the clip rectangle.

Hence, part of the shape of a marker may appear outside the clip rectangle providing its specifying point is within the clip rectangle.

For ‘shape' clipping, the shape of the rendered marker symbols are clipped to the intersection with the clip rectangle; that is, nothing is drawn outside the clip rectangle. Portions of the marker symbol may appear inside the clip rectangle even if the marker's position is outside.

For ‘locus then shape' clipping, the clipping is first applied to the specifying points of each marker, as with ‘locus' clipping, and then subsequently the rendered shape of the markers are again clipped.

6.6.2.4 Transformation of the marker element

If the MARKER SIZE SPECIFICATION MODE has the value ‘absolute', then marker size is subject to the VDC-to-device mapping (see 6.4.7) as well as to both segment and copy transformation (see 6.10.4.2 and 6.10.5). The shape of markers is never affected by transformations; for example, a circle used as a marker type shall always appear as a circle. Only the marker size may be transformed. Conceptually, vectors with length equal to the marker size but at all possible orientations are transformed; the marker size is the maximum length of the vectors under the transformation.

If the marker size is specified in mode ‘scaled', ‘fractional', or ‘mm' it is not affected by any transformations.

6.6.3 Text elements

6.6.3.1 Description

Three text elements are provided.

TEXT	generates a text string (or part of a text string) aligned to a particular point.
RESTRICTED TEXT	generates a text string (or part of a text string) that is constrained within a given area.
APPEND TEXT	generates a part of a text string which was started with a TEXT or RESTRICTED TEXT element.

6.6.3.2 Attributes

The appearance of text elements is controlled by the text attributes, the TEXT BUNDLE INDEX, and those ASPECT SOURCE FLAGs which are associated with the text attributes that may be bundled. These are described in 6.7.3.

Changes to certain of the text attributes, as listed in the description of APPEND TEXT (see 7.6.6), are permitted between a non-final text element and its succeeding APPEND TEXT element.

6.6.3.3 Usage of text elements

Each text element has a ‘final/not-final' flag. This permits a text string to be started with a TEXT or RESTRICTED TEXT element and continued with one or more APPEND TEXT elements. The last element will have its flag set to ‘final'. The initial element is always TEXT or RESTRICTED TEXT; subsequent elements may only be APPEND TEXT.

The current setting of TEXT ALIGNMENT is used to align the complete text string assembled from the separate text elements.

6.6.3.4 Clipping of text elements

Clipping of text strings is described in 6.7.3.2.

6.6.3.5 Transformation of text elements

The vectors specified by the CHARACTER ORIENTATION element (see 6.7.3.2) are subject to the VDC-to-Device mapping (see 6.4.7) as well as to both segment and copy transformation (see 6.10.4.2 and 6.10.5).

6.6.4 Filled-area elements

6.6.4.1 Description

There are two general fill elements: POLYGON and POLYGON SET. In addition, there are several filled-area elements that correspond to the basic geometric shapes — circles, rectangles, pie sectors, etc. These metafile representations of the common geometric entities are compact, scalable, and independent of the resolution of the final viewing surface. Finally, there is also a compound filled-area element — Closed Figure whose definition is composed of a sequence of individual line and filled-area primitives (see 6.6.11).

POLYGON	generates an area and its edge, defined by a list of points; the style of the area is one of ‘hollow', ‘solid', ‘pattern', ‘hatch', ‘empty', ‘geometric pattern', or ‘interpolated'; the visibility and style of the edge of the area depend on the edge attributes alone.
POLYGON SET	generates a number of areas and their edges, defined by a list of vertex points and vertex flags; the set of styles is the same as for POLYGON; the vertex flags indicate the different polygons in the set; the vertex flags and the edge attributes together control the visibility and style of individual edge segments of each polygon.
RECTANGLE	generates an upright rectangular area; the set of styles is the same as for POLYGON.
CIRCLE	generates a circle; the set of styles is the same as for POLYGON.
CIRCULAR ARC xxx CLOSE	generates a partial circular area; ‘pie' and ‘chord' style arcs are possible; two parameterization of the arcs are provided; these are described in 7.6.14 and 7.6.16; the set of styles is the same as for POLYGON.
ELLIPSE	generates an ellipse; the parameterization of the ellipse is described in 7.6.17; the set of styles is the same as for POLYGON.
ELLIPTICAL ARC CLOSE	generates a partial elliptical area; ‘pie' and ‘chord' style arcs are possible; the parameterization is described in 7.6.19; the set of styles is the same as for POLYGON.

6.6.4.2 Attributes

The appearance of all filled-area elements is controlled by the fill attributes, the FILL BUNDLE INDEX, the EDGE BUNDLE INDEX, and the ASPECT SOURCE FLAGs associated with the fill attributes that may be bundled. These are described in 6.7.4.

6.6.4.3 Usage of fill elements

POLYGON provides for the representation of standard irregular areas. RECTANGLE, because it is upright, is a more efficient parameterization of a rectangle than a POLYGON and may be implemented directly in some systems.

The circular and elliptical fill primitives, as well as closed figure fill primitives incorporating such line primitives as the conic arc elements and spline curve elements (see 6.6.11), provide an efficient parameterization and allow the areas to be produced accurately without knowledge of the resolution of the final viewing surface.

POLYGON SET allows a related set of polygons to be represented. All attributes of each of the polygons are the same. The specification of the vertex flags allows disjoint polygons (such as both the body and the dot of the letter 'i'), holes (as in a broad ring) and overlapping areas. Accurate rendering of abutting areas of uniform or graded colour, pattern or hatch, and control over individual edge segment visibility, are possible.

6.6.4.4 Interior

The interior of a filled-area element is defined as follows. For a given point, create a straight line starting at that point and going to infinity. If the number of intersections between the straight line and the filled area boundary is odd, the point is within the filled area; otherwise it is outside. If the straight line passes a filled-area vertex tangentially, the intersection count is not affected. If a point is within the filled area, it is included in the area to be filled subject to the rules for boundaries and edges (see 6.7.4.3).

6.6.4.5 Edges

The edge of a filled-area element can be either visible or invisible. If visible, the individual edge attributes or the EDGE BUNDLE INDEX (according to the edge ASF values) govern the appearance.

If the edge is visible, it is drawn on top of the interior — the edge has precedence over the interior when drawn and will always be fully visible. The boundary drawn for style 'hollow' is considered as the representation of the interior. While the edge has precedence, the boundary may be partly visible as well. Parts of edges which are clipped become invisible — clipping of edges is identical to clipping of line elements. Parts of interiors which are clipped will, in the case of interior style 'hollow', have a boundary drawn at the clipping boundary.

The "realized edge" is defined to be the zero-width ideal boundary line of the filled-area if the edge is invisible, and the finite-width displayed line if the edge is visible. ISO/IEC 8632 does not mandate the alignment of the finite-width realized edge with respect to the zero-width ideal edge (i.e., whether the former is centred on the latter or aligned some other way such as inside).

The "realized interior" is defined as extending to and terminating at the realized edge. The discussion of interior in the remainder ISO/IEC 8632 should be considered to pertain to realized interior.

6.6.4.6 Clipping

If parts of a filled-area element are clipped, then the intersection of the interior and the clip boundary becomes part of the boundary of the resulting clipped area for the purposes of display of the boundary for interior style 'hollow'. If the edge is visible, it is not drawn along the new boundary segments created by the clipping of the area. Edge clipping is controlled by the EDGE CLIPPING MODE element, which has the same enumerations as LINE CLIPPING MODE. Edges are clipped in the same way that lines are clipped; see 6.6.1.6.

6.6.4.7 Transformation

The entire mathematical locus of rectangles, circular and elliptical filled-area elements is subject to the VDC-to-Device mapping (see 6.4.7), segment transformations (see 6.10.4.2) and copy transformations (see 6.10.5). Because anisotropic transformation does not preserve angles between non-parallel lines, rectangles may become parallelograms and circles may become ellipses.

If the INTERIOR STYLE SPECIFICATION MODE is 'absolute', the geometric aspects of fill interiors are subject to all transformations. These aspects include PATTERN SIZE; direction, spacing, and width of hatch lines; and reference geometry of interpolated interior. If the mode is 'scaled', 'fractional', or 'mm', then none of these aspects is subject to transformation.

Geometric aspects of edges are treated in exactly the same way as the corresponding aspects of lines.

6.6.5 Cell elements

The cell elements comprise a single Cell Array element and two Tile Array elements.

6.6.5.1 Cell array element

The single cell array element is:

CELL ARRAY	represents a 2-dimensional array of colour values, which cover a rectangle or parallelogram.

The colour values are either direct colour values or indexes into the COLOUR TABLE, according to the current COLOUR SELECTION MODE. The colour values are in the precision declared by a local colour precision parameter of the CELL ARRAY element.

CELL ARRAY has no associated attributes.

6.6.5.2 Tile Array Elements

A Tile Array is a compound raster image primitive, whose definition is delimited by the BEGIN TILE ARRAY and END TILE ARRAY delimiter elements. Between the delimiter elements is a series of equally sized individual "tiles" which all together form a contiguous rectangular block of tiles. Each tile is defined by a non-overlapping TILE or BITONAL TILE element. The first tile is placed at the position parameter of the BEGIN TILE ARRAY element. Any subsequent tiles are placed in order first in the cell path direction and then in the line progression direction. The tile positions are numbered as shown in figure 4.

The elements defining the tiles which comprise a Tile Array are

BITONAL TILE	defines a rectangular raster image, either uncompressed or compressed according to a selected compression method. Only two colours are used to define the image. Each cell is associated with one of the colour indexes 0 or 1, and the colour values associated with 0 and 1 are defined locally by each BITONAL TILE element.
TILE	defines a rectangular raster image, either uncompressed or compressed according to a selected compression methods. The colours associated with the cells may either be bitonal or full colour, may be specified by either indexed or direct mode, and are specified according to the applicable colour precisions and modes.

The TILE element contains a cell colour precision parameter, which behaves as does the local colour precision of CELL ARRAY (see previous section). BITONAL TILE does not contain this parameter, as its colours are always represented at 1-bit precision prior to compression.

The TILE and BITONAL TILE elements are not independent graphical primitives as are POLYLINE and CELL ARRAY. They contain no positioning or dimensioning information. Each element contains only the raster content of a single raster tile and any control parameters which apply to that tile. The complete Tile Array primitive is formed by one or more tiles between BEGIN TILE ARRAY and END TILE ARRAY. BEGIN TILE ARRAY contains all parameters which apply to the collection of tiles (if there is more than one tile). The parameters apply uniformly to each tile in the collection.

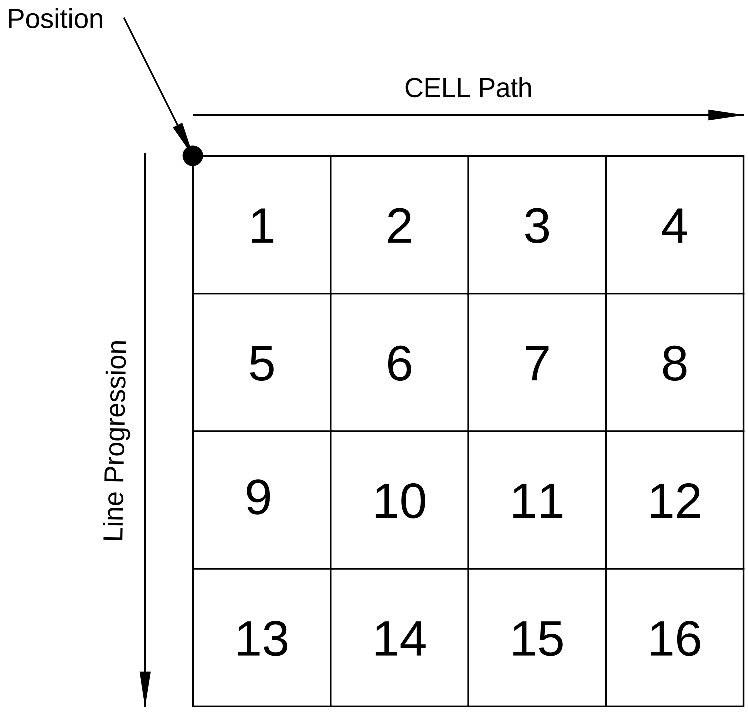


Figure 4 — Ordering and layout of tiles

6.6.5.2.1 Relationship to CELL ARRAY

Both tile arrays and cell arrays are composed of cells. While cells in cell arrays are subject to all transformations, cells in tiles are always axis aligned and rectangular. They scale to the display surface but do not otherwise transform. As with CELL ARRAY, the position point of a tile array corresponds with the corner of the first cell (rather than the centre of the cell).

6.6.5.2.2 Allowable states for tiles and tile array elements

The tile elements, TILE and BITONAL TILE, may appear only in Tile Array State (TAS).

Tile Array compound elements, delimited by BEGIN TILE ARRAY and END TILE ARRAY, may appear only in Picture Open State. They may not appear in segments or other compound primitive definitions.

6.6.5.2.3 Compressed cell data

The cell colour data of the tile elements is a compressed stream of cell colour specifiers. The datatype is Bitstream. For the BITONAL TILE element the Bitstream parameter consists of a sequence of 1-bit binary colour indexes which are compressed by the selected technique (the list of techniques includes ‘bitmap' which is uncompressed). The resulting compressed binary data object is the parameter of the element. Each of the CGM encodings (Binary, Character, and Clear Text) define a technique for representing and encoding the compressed binary data object.

6.6.5.2.4 Tiling

The tiling mechanism specified is based on the Tiled Raster Interchange Format that has been developed for ISO/IEC 8613-7. Definition of a tile array is initiated by the BEGIN TILE ARRAY delimiter element and terminated by the END TILE ARRAY element. During tile array definition subsequent tile elements define individual tiles within the tiled image. The number of tiles is determined by the parameters of the BEGIN TILE ARRAY element. A tile array contains one or more tiles.

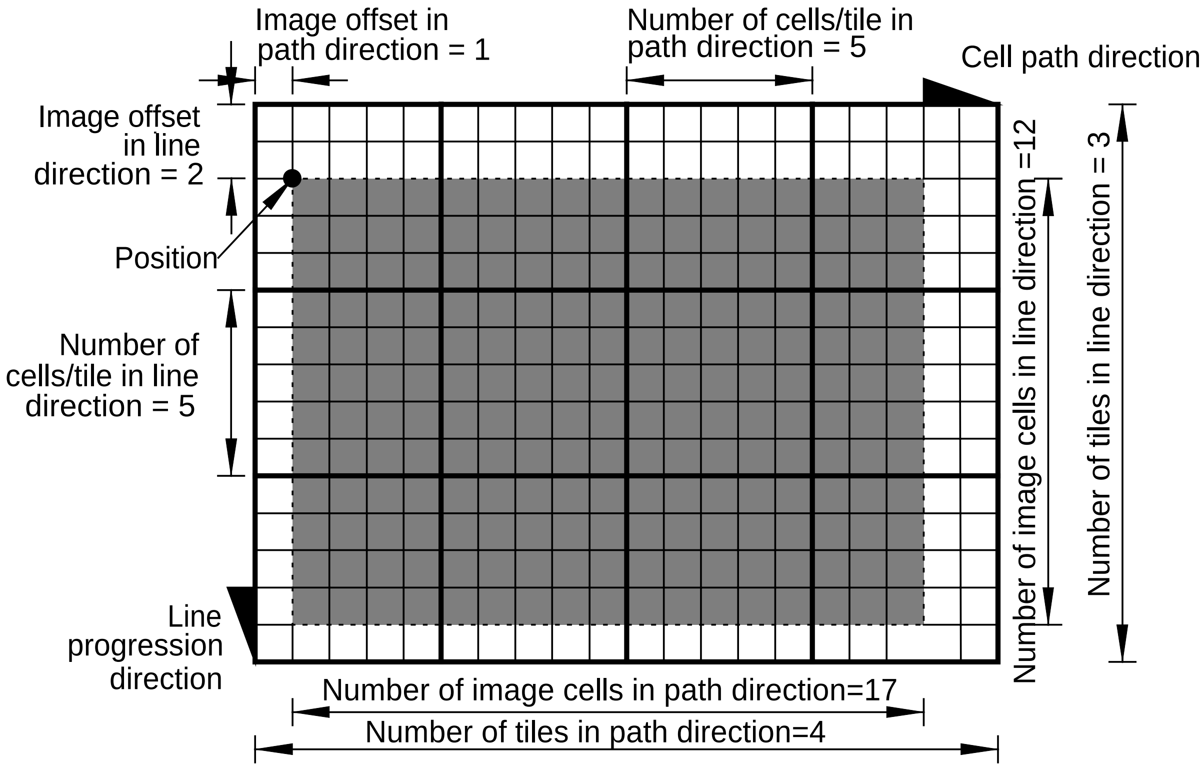


Figure 5 — Relationship of the image to the tile space

The number of tiles defined during tile array definition shall match the number indicated by the BEGIN TILE ARRAY element. Annex D contains recommendations for interpreter fallback in the case that the tiles are missing.

The tiling offset and size parameters define the position of the actual image within the tile space, relative to the position parameter of the BEGIN TILE ARRAY element. This is the portion of the whole tile array which contains information. There may be a border of unused cells surrounding the actual image. These cells contain no information and should not be drawn by interpreters. These unused cells are however included in the encoding of those tiles which overlap the border and the defined image. See figure 5 for the relationship of the actual image and the total tile space.

NOTE The unused cells must have some value assigned, and it is recommended that they all be set identically to the background colour.

6.6.6 Circular arc elements

The CGM provides for two forms of specification of circular arc elements: a centre-radius specification and a 3-point specification. Each has its advantages and disadvantages with respect to numerical accuracy, relationship of defining data to the VDC range, etc.

NOTE When choosing which parameterization to use, one should decide where possible numerical inaccuracy would be least disturbing. The 3-point form specifies exact arc endpoints, but might result in inaccurate centre-point calculations, whereas the centre form specifies exact centre point but might result in roundoff errors on the ends of the arc. The 3-point form would thus be more appropriate for smoothly joining an arc to a polyline in a line drawing, whereas the centre form would be more appropriate for pie charts.

6.6.7 Elliptical elements

6.6.7.1 Geometric concepts

Ellipses are specified by Conjugate Diameter Pairs. A Conjugate Diameter Pair (CDP) of an ellipse is a pair D,d of diameters of the ellipse such that a tangent to the ellipse at each endpoint is parallel to the other diameter. The four tangents to the ellipse at the endpoint of the CDP thus form a parallelogram whose sides are bisected by the endpoints of the diameters.

Any CDP of the ellipse remains a CDP across any graphical transformation which transforms an ellipse into an ellipse. This is illustrated in figure 6 in which the ellipse has been scaled by a factor of two in the y-direction only.

Thus any CDP of a desired ellipse can be used to specify the ellipse. Note that the (mutually perpendicular) major and minor axes of an ellipse and any pair of perpendicular diameters of a circle are CDP's, although they do not necessarily remain perpendicular across a transformation.

Thus to specify an ellipse, all that is needed is three points:

— the centrepoint of the ellipse;

— two CDP endpoints (one endpoint from each diameter).

6.6.7.2 Parameterization of elliptical elements in CGM

The ellipse itself in each of the three elliptical elements is parameterized as in the preceding section, the centrepoint and two CDP endpoints. For the two elliptical arc elements, the start and end of the defined arc section is parameterized by two semi-infinite rays originating at the centrepoint. The intersection of these rays with the ellipse defines two points on the ellipse, and these two points define the arc. The conjugate diameters parameterization of ellipses and elliptical arcs has the property of being transformable — the ellipse defined by the transformed parameter data is the transformed ellipse. The conjugate diameter parameterization has other useful properties as well.

For simplicity, consider the ellipse that is centred at the origin, and let $\mathbf{P_1}$ and $\mathbf{P_2}$ designate the endpoints of the conjugate diameters. Let **M** be the 2x2 matrix whose first column is $\mathbf{P_1}$ and whose second column is $\mathbf{P_2}$**.** The transformation **M** maps points on the unit circle centred at the origin ($x^2 + y^2 = 1$) onto the ellipse. The unit circle is referred to as the "canonical ellipse". If the ellipse is non-degenerate, then **M** is non-singular, hence invertable, and $\mathbf{M}^{-1}$ maps points on the ellipse onto points on the unit circle centred at the origin. **M** maps the unit vectors $\mathbf{u_1}$ and $\mathbf{u_2}$ respectively onto $\mathbf{P_1}$ and $\mathbf{P_2}$, where $\mathbf{u_1}$ and $\mathbf{u_2}$ are vectors from the origin to the points (1,0) and (0,1) respectively. These principles generalize easily to ellipses which are not centred at the origin there is a translation term in the mapping so that the transformation is not linear but is affine.

6.6.8 Hyperbolic arc element

The CGM parameterization of the hyperbolic arc closely parallels that of the ellipse (see figure 7). The "canonical hyperbola" is defined by $x^2 - y^2 = 1$. It passes through the point (1,0). At (1,0) the tangent to the hyperbola is parallel to the vector $\mathbf{u_2}$, which is the vector from the origin to the point (0,1). The canonical hyperbola has "centre" (the point where the asymptotes cross) at the origin. For any non-degenerate hyperbola centred at the origin, there is a linear transformation which maps the canonical hyperbola onto the given hyperbola. This transformation maps the points (1,0) and (0,1) respectively onto a pair of points $\mathbf{P_1}$ and $\mathbf{P_2}$. In this case, $\mathbf{P_1}$ is on the hyperbola but $\mathbf{P_2}$ is not. At $\mathbf{P_1}$ the tangent to the hyperbola is parallel to the line from the origin to $\mathbf{P_2}$. The asymptotes of the hyperbola are parallel to the vectors $\mathbf{P_1} + \mathbf{P_2}$ and $\mathbf{P_1} - \mathbf{P_2}$. Points with such properties are referred to as a the *conjugate radius endpoint* and *transverse radius endpoint* respectively. The *transverse radius* endpoint is the one which lies on the hyperbola; the *conjugate radius* endpoint does not. These points (plus the centre point) parameterize the hyperbola in CGM.

As with the ellipse, if the matrix **M** is formed whose columns are the points $\mathbf{P_1}$ and $\mathbf{P_2}$, then this is the invertable transformation which maps points on the canonical hyperbola onto points on the given hyperbola (and whose inverse maps the given hyperbola onto the canonical hyperbola). The generalization to hyperbolas whose centre is not the origin is straight forward.

As with elliptical arcs, the start and end of the hyperbolic arc are parameterized by vectors from the centre.

In both the case of the ellipse and the case of the hyperbola, the conjugate parameterizations can be derived from x-y implicit equations and vice-versa.

6.6.9 Parabolic arc element

The principles used to parameterize elliptical arcs are also used to parameterize parabolic arcs, but the analogy is not quite as strong between parabolic arc and elliptical arc as it is between hyperbolic arc and elliptical arc. The parameterization is again in terms of a transformation of a "canonical parabola". In this case, the canonical parabola is $2(x + y) = (x - y)^2 + 1$ for $x \leq 1$ and $y \leq 1$. This parabolic arc is symmetric about the line $y = x$, has endpoints (1,0) and (0,1), is tangent to the x-axis and y-axis respectively at these points, and remains entirely in the first quadrant. See figure 8.

The general parabolic arc is parameterized by the endpoints of the arc, $\mathbf{P_1}$ and $\mathbf{P_2}$ and the intersection of the tangents to the arc at the endpoints. This intersection point is called the "centre" of the parabolic arc, **C**. Define $\mathbf{V_1 = P_1 - C}$ and $\mathbf{V_2 = P_2 - C}$, and form the 2x3 matrix **M** whose first column consists of the components of $\mathbf{V_1}$, second column consists of the components of $\mathbf{V_2}$, and third column consists of the components of **C**. For non-degenerate parabolic arcs **M** is an affine transformation that maps points on the canonical parabolic arc onto points on the given parameterized parabolic arc.

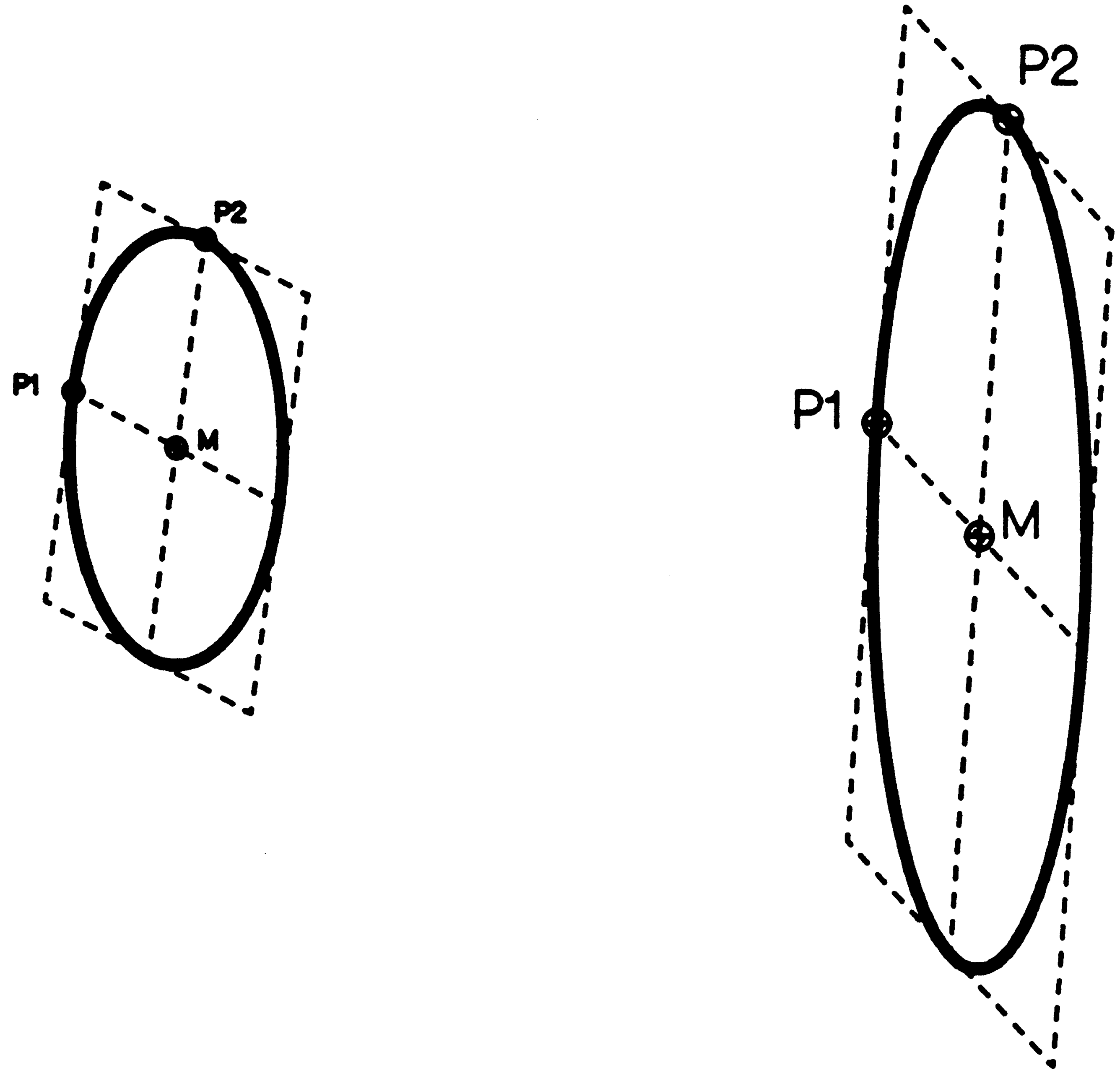


Figure 6 — Anisotropic scaling of an ellipse

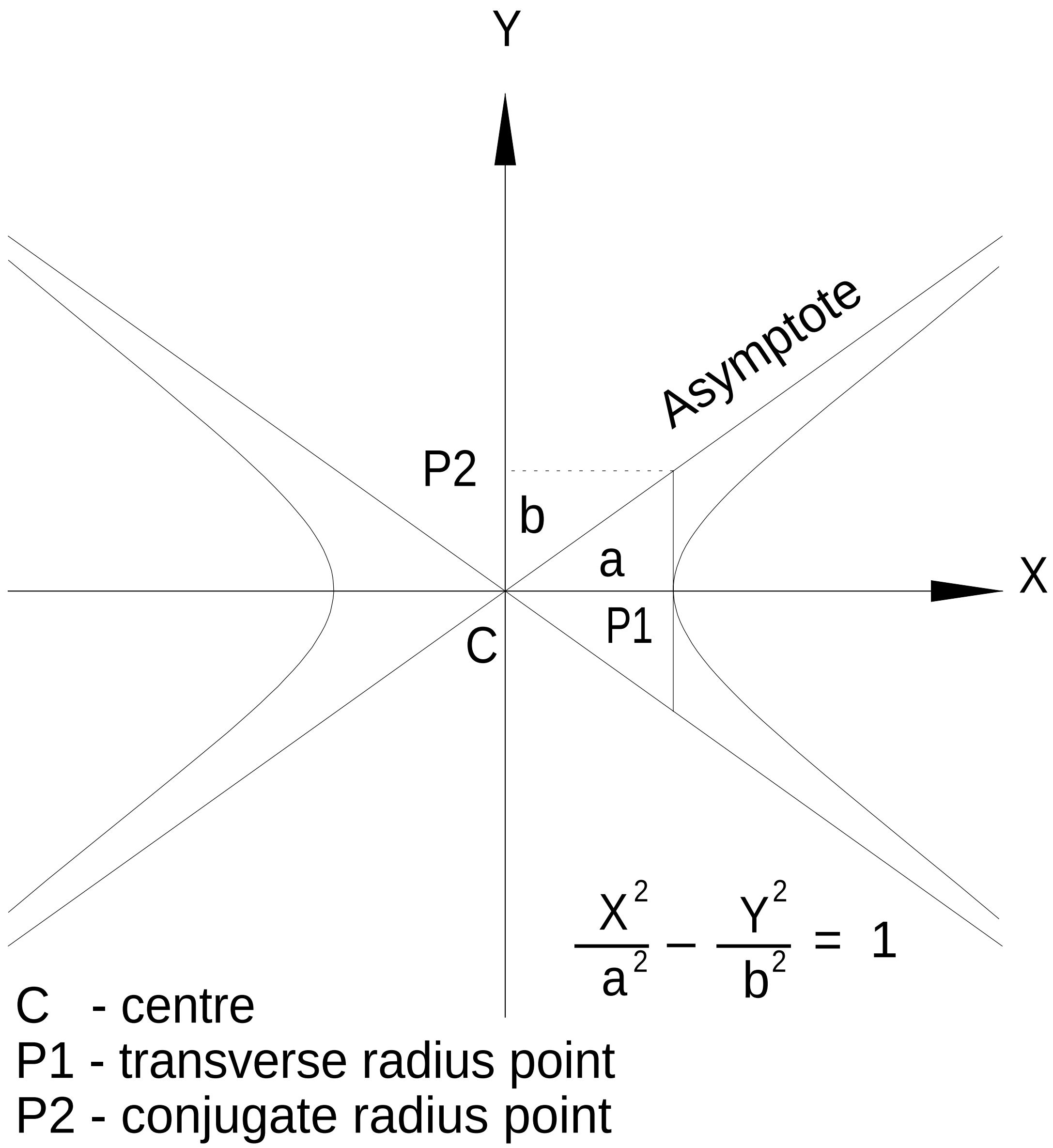


Figure 7 — Example of hyperbolic arc

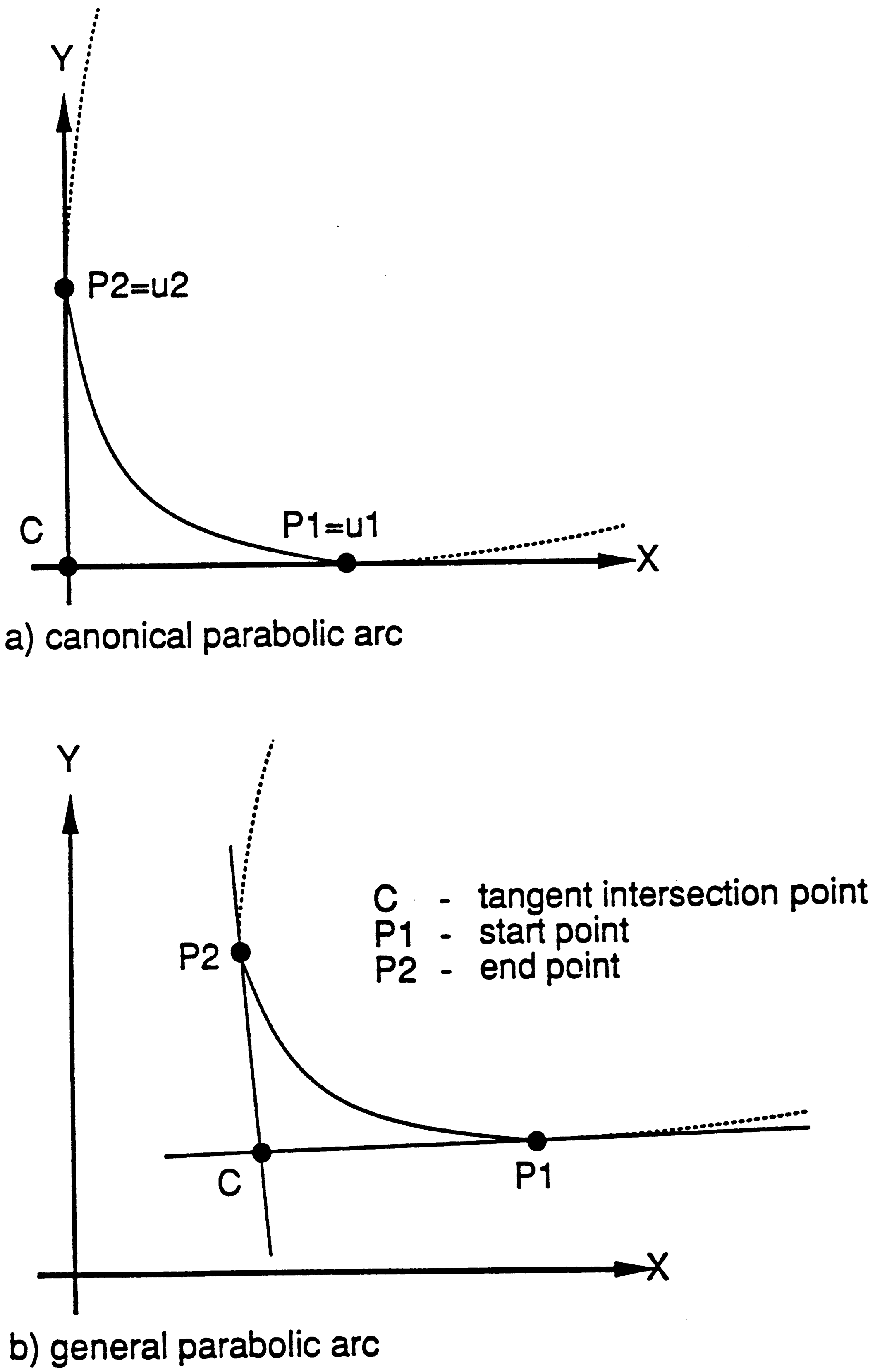


Figure 8 — Example of parabolic arc

6.6.10 Spline curve elements

The CGM provides three spline curve elements: non-uniform B-splines; non-uniform rational B-splines; and Bezier curves.

6.6.10.1 Non-uniform B-splines

The CGM provides both rational and non-rational B-splines of varying orders.

6.6.10.1.1 Parameterization

The non-uniform B-spline is parameterized by a spline order, a list of knots, a list of control points, and parameter range limits defining the curve section to be drawn.

The non-uniform rational B-spline is parameterized by a spline order, a list of knots, a list of control points, a list of weights associated with the control points, and parameter range limits defining the curve section to be drawn.

6.6.10.1.2 Mathematical definition

The non-uniform B-spline is expressed parametrically in the form

Equation 1 Non-uniform B-spline

$$G(t) = \sum_{i=1}^{n} P_i B_{i,k}(t)$$

where

n — number of control points;
P$_i$ — control points, (P_{x_i}, P_{y_i});
$B_{i,k}$ — B-spline basis functions defined by order k and knot vector T.

The knot vector consists of a non-decreasing sequence of real numbers $(T_1, ..., T_{n+k})$.

NOTE If the first k knots are identical and the last k knots are identical, then the spline is referred to as *clamped*. Such splines have useful properties, including that they start at the first control point and end at the last control point. The clamped form is recommended, but not required, by this International Standard. Profiles may restrict splines to clamped form, as does the Model Profile of Annex I.

The curve itself is defined for the range $T_k \le t \le T_{n+1}$ and can be confined to the range $[T_{\min}, ..., T_{\max}]$,where $T_k \le T_{\min} \le T_{\max} \le T_{n+1}$. $T_{\min}$ and $T_{\max}$ are specified as part of the non-uniform B-spline primitive.

The B-spline basis functions are defined by the recursive relation:

$$B_{i,1}(t) = 1, \quad \text{if } T_i \le t < T_{i+1}$$

$$B_{i,1}(t) = 0, \quad \text{otherwise}$$

and for $k > 1$

$$B_{i,k}(t) = \frac{(t - T_i) B_{i,k-1}(t)}{T_{i+k-1} - T_i} + \frac{(T_{i+k} - t) B_{i+1,k-1}(t)}{T_{i+k} - T_{i+1}}, \quad \text{if } T_i \le t < T_{i+k}$$

$$B_{i,k}(t) = 0, \quad \text{otherwise}$$

For multiple identical knot values, the fractional terms above may evaluate to 0/0. In such cases the terms are defined to be 0.

A non-rational non-uniform B-spline curve is generalized by using equation 1 with a set of control points

$$\mathrm{P}_i = (P_{x_i}, P_{y_i}, P_{z_i}) \quad \text{for } i = 1,...,n.$$

A non-uniform rational B-spline is expressed parametrically in the following form:

Equation 2 Non-uniform rational B-spline

$$G(t) = \frac{\sum_{i=1}^{n} B_{i,k}(t) P_i w_i}{\sum_{i=1}^{n} B_{i,k}(t) w_i}$$

Given a set of control points $\{\mathrm{P}_i\}$ and weights $\{w_i\}$, this is equivalent to creating a set of homogeneous control points $\{w_i\mathrm{P}_\mathrm{i}, w_i\}$ in 3D, using equation 1 in 3D and projecting the resulting points back into 2D, projecting the homogeneous point onto the {0,0,1} (w=1) plane in 3D. This is achieved by dividing the first two (homogeneous) coordinates by the third.

6.6.10.2 Polybezier

This element defines one or more cubic Bezier curves.

6.6.10.2.1 Parameterization

The polybezier is parameterized by a list of points and an indicator specifying the degree of continuity between the individual Bezier curves. If there are *n* $(n \geq 1)$ Bezier curves, then the data list will contain:

— 4n points if the continuity parameter is discontinuous;

— 3n + 1 points if the continuity parameter is continuous;

In the case of ‘discontinuous' the point list is divided into consecutive sets of 4 points. Each set defines a single Bezier curve as defined below.

In the case of ‘continuous' then after the first set of 4 points, defining the first Bezier curve, the subsequent curves are defined by three points each. The first point of each curve definition is omitted because it is identical to the last point of the preceding definition.

6.6.10.2.2 Geometric concepts

The following discussion assumes that there are 4 points defined for each Bezier curve. In one of the continuity conditions described above, one or two of the points may not actually be in the parameter data of the element. If the points in a given 4-point set are designated $P_0..P_3$, then the defined Bezier curve goes from P_0 to P_3 using P_1 and P_2 as control points. The defined curve starts at P_0 and at P_0 is tangent to the line segment from P_0 to P_1. The curve ends at P_3 and at P_3 is tangent to the line segment from P_2 to P_3. The curve lies entirely within the convex hull defined by the points.

The curve is defined by the cubic parametric equations:

$$X(t) = A_x t^3 + B_x t^2 + C_z t + X_0$$

$$Y(t) = A_y t^3 + B_y t^2 + C_y t + Y_0$$

as *t* ranges from 0 to 1. The six coefficients A_x, B_x, C_x, A_y, B_y, C_y are defined by

$$X_1 = X_0 + \frac{C_x}{3}$$

$$Y_1 = Y_0 + \frac{C_y}{3}$$

$$X_2 = X_1 + \frac{(C_x + B_x)}{3}$$

$$Y_2 = Y_1 + \frac{(C_y + B_y)}{3}$$

$$X_3 = X_0 + C_x + B_x + A_x$$

$$Y_3 = Y_0 + C_y + B_y + A_y$$

6.6.11 Closed figures

6.6.11.1 Construction of closed figures

A closed figure is a fill type compound object which commences with a BEGIN FIGURE element, followed by an ordered sequence of line and fill primitives (and optionally attributes and NEW REGION elements), and followed by END FIGURE. Edge attribute values are associated with the edge portions of the closed figure and fill attribute values are associated with the complete graphic object. BEGIN FIGURE and END FIGURE elements are delimiter elements; NEW REGION is a control element. The entire fill object is considered as a single unit on interpretation.

6.6.11.1.1 Closure point

The first point of the first line primitive in a new region is the closure point for that region. On interpretation this closure point is retained for use in closing the region. When the region is closed (with a NEW REGION or END FIGURE element, or by a fill primitive which begins a new region) an implicit boundary portion from the last point of the last line primitive in the region to this closure point is added to the closed figure on interpretation, unless these points are already coincident.

6.6.11.1.2 Regions

A closed figure consists of one or more regions. A region has a closed boundary which may be concave, convex, or self intersecting. A region is formed either by invoking a fill primitive in between BEGIN FIGURE and END FIGURE elements which closes the last region and contributes one or more complete regions; by invoking NEW REGION to start new regions to be formed from line primitives; or by a final invocation of END FIGURE. A closed figure constructed from only line primitives without use of NEW REGION consists of a single region.

The NEW REGION element may occur at any time during the closed figure construction. If the current region is closed, the element is ignored on interpretation. If the current region is open, an implicit boundary portion is added from the last point of the last primitive to the current closure point unless CONNECTING EDGE has been invoked after the last line primitive, in which case, an explicit boundary portion and edge portion is added by the CONNECTING EDGE line primitive.

6.6.11.2 Boundaries and edges

The boundary of each region consists of a combination of implicit boundary portions and edge portions.

6.6.11.2.1 Explicit boundary portions.

Explicit boundary portions and edge portions are those added by the inclusion of primitives during closed figure construction. These are generated in the following situations:

— For fill primitives other than POLYGON SET, the complete edge becomes an explicit boundary portion and edge portion in the closed figure.

— For line primitives, only those portions become explicit boundary portions and edge portions which would normally be drawn if an interpreter were rendering the line primitive independently in a metafile. In particular for DISJOINT POLYLINE, only the segments from the first point to the second point, from the third point to the fourth point, and so on, become explicit boundary portions and edge portions when incorporated into closed figures.

— When a CONNECTING EDGE primitive precedes an element in the closed figure definition sequence which would otherwise have resulted in the addition of an implicit boundary portion to the closed figure, either to close a region (including closing the closed figure itself) or to connect two line primitives, then that CONNECTING EDGE primitive results in the portion added being an explicit boundary portion and edge portion. CONNECTING EDGE preceding or following DISJOINT POLYLINE or POLYGON SET does not affect the interpretation of those elements with respect to boundaries and edges.

Edge portions have associated edge attribute values taken from the current attribute values on interpretation. These values can be changed between the line and fill primitives that result in edge portions in a closed figure, and hence each edge portion has a distinct set of attribute values associated with it.

6.6.11.2.2 Implicit boundary portions.

Edge attributes are never associated with implicit boundary portions. Implicit boundary portions are only rendered on interpretation for interior style HOLLOW and are a special representation of the interior, not a representation of any portion of the edge.

Implicit boundary portions are added on interpretation to the closed figure definition under the following circumstances:

— When NEW REGION, END FIGURE, or a fill primitive is interpreted and the current region has not been explicitly closed and CONNECTING EDGE has not occurred since the last line primitive, an implicit boundary portion is added from the last point of the last primitive to the current closure point to close the region.

— When the last point of the preceding line primitive is not coincident with the first point of the current line primitive, an implicit boundary portion is created to connect the last point of the preceding line primitive to the first point of the current line primitive.

— When portions of a DISJOINT POLYLINE primitive would not normally be rendered (i.e. from the second point to the third point, from the fourth point to the fifth point, and so on), implicit boundary portions are added between these points. (These are additional to the ones which may be added to connect to a preceding or following line primitive or to effect region closure after the disjoint polyline.)

— The portions of a POLYGON SET primitive as described below.

6.6.11.2.3 Conditions under which no boundary or edge is added

No boundary or edge portion is ever created connecting two regions, regardless of how those regions were created or closed.

6.6.11.3 Contribution of graphical primitive elements to the closed figure

6.6.11.3.1 Contribution of line elements to the closed figure

For line primitives, the ‘first point' of a line primitive is connected to the ‘last point' of the preceding line primitive, and the connecting implicit boundary portion becomes part of the boundary of the closed figure on interpretation. For each of the line primitives, the first and last points are defined to be as follows:

POLYLINE p1, p2, ..., pn:	p1 is the first point; pn is the last point.
DISJOINT POLYLINE p1, p2, ..., pn:	p1 is the first point; pn is the last point.
CIRCULAR ARC 3 POINT p1, p2, p3:	p1 is the first point; p3 is the last point.
CIRCULAR ARC CENTRE, CIRCULAR ARC CENTRE REVERSED:	The first point is the intersection of the circle with the ray (dx start, dy start) from the centre point (i.e. the clockwise end of the arc for CIRCULAR ARC CENTRE, the anti-clockwise end of the arc for CIRCULAR ARC CENTRE REVERSED); the last point is the intersection of the circle with the ray (dx end, dy end) from the centre point (i.e. the anti-clockwise end of the arc for CIRCULAR ARC

CENTRE, the clockwise end of the arc for CIRCULAR ARC CENTRE REVERSED).

ELLIPTICAL ARC: The first point is the intersection of the ellipse with the ray (dx start, dy start) from the centre point; the last point is the intersection of the ellipse with the ray (dx end, dy end) from the centre point.

GENERALIZED DRAWING PRIMITIVE: For GDPs which generate line primitives, the first point and last point are as defined in the GDP registration and associated documentation.

HYPERBOLIC ARC The first point is the intersection of the hyperbola with the ray (dx_start, dy_start) from the centre point, and the last point is the intersection of the hyperbola with the ray (dx_end, dy_end) from the centre point.

PARABOLIC ARC The first point is the start point, the last point is the end point.

NON-UNIFORM B-SPLINE
NON-UNIFORM RATIONAL B-SPLINE The first point is the point of the curve corresponding to the start value of the parameter, and the last point is the point of the curve corresponding to the end value of the parameter.

POLYBEZIER For the case ‘continuous': the first point corresponds to the point defined by *t = 0* for the first 4 control points, and the last point corresponds to the point defined by *t = 1* for the last 3 control points. For the case of ‘discontinuous': each 4 points define an individual Bezier curve, and an implicit boundary is drawn joining the last point of each Bezier curve to the first point of the following curve (similar to DISJOINT POLYLINE).

CONNECTING EDGE If the region is open, the start point of the connecting edge is the last point of the last line primitive, and the end point of the connecting edge is either the first point of the following primitive or the current closure point as described above. If the connecting edge would be of zero length (i.e. if the two points it connects are coincident), the element is ignored on interpretation. The current modal values of the edge attributes are associated with any edge portion generated by this element.

If the current region is not open, invocations of the CONNECTING EDGE elements encountered are ignored on interpretation (i.e. CONNECTING EDGE shall not be used to connect regions).

Invoking CONNECTING EDGE multiple times after a line primitive results in the first instance (with its associated attributes) being used on interpretation.

On interpretation the theoretical definitions of the line primitives, not their renditions on the display surface, are used to define the explicit boundary portions of the closed figure. In particular, clipping does not apply to the construction of the closed figure, and the gaps or spaces of the edge type or the rendered width of the edge width do not affect the definition of the boundary of the closed figure.

6.6.11.3.2 Contribution of fill elements to the closed figure.

Each fill primitive contributes a complete region to the figure (POLYGON SET may contribute more than one), after first closing the current region if one is open. On interpretation, an implicit NEW REGION is performed before and after a fill primitive (i.e. the new region resulting from a fill primitive is closed, and the next primitive begins a new region.)

The unclipped boundary of each fill primitive contributes to the unclipped boundary of the closed figure.

POLYGON SET primitives contribute to closed figure construction as follows:

— A POLYGON SET is considered to contribute one or more complete regions. If the current region has not been closed, an implicit NEW REGION is performed before the POLYGON SET is added to the figure definition. If the POLYGON SET does not end with a point whose edge-out flag is ‘close visible' or ‘close invisible', an implicit NEW REGION is performed after the POLYGON SET.

— Sequences of points with edge-out flag ‘visible' are treated as if they were polylines, terminating with the first point with a different edge-out flag. Each such polyline becomes an edge portion of the boundary of the figure. The edge attribute values (including EDGE VISIBILITY) in effect when POLYGON SET occurs are associated on interpretation with any edge portion added in this way.

— Sequences of points with edge-out flag ‘invisible' contribute implicit boundary portions which are polylines joining the points in the sequence, but not edges. Edge attribute values are not associated with these.

— Points with edge-out flag ‘close invisible' generate the equivalent of a NEW REGION, generating an implicit boundary portion from this point to the current closure point if these are not coincident, and closing the current region.

— Points with edge-out flag ‘close visible' generate the equivalent of a CONNECTING EDGE followed by a NEW REGION, resulting in an edge portion from this point to the current closure point if these are not coincident. The edge attribute values (including EDGE VISIBILITY) in effect when POLYGON SET is invoked are associated with any edge portion added in this way.

6.6.11.3.3 Contribution of GDPs to the closed figure.

A GDP which is defined as a line primitive shall specify which is the first point and the last point in its point list, with respect to closed figure construction. Such GDPs are assumed to contribute to a closed figure a boundary corresponding to the unclipped locus which would be rendered on interpretation if the element occurred outside closed figure construction. Any other behaviour shall be as documented explicitly in the GDP description. A GDP which is defined as being a fill primitive is treated as described in the previous section. Any variation or special handling for closed figure construction shall be documented explicitly in the GDP description.

6.6.11.4 Examples of closed figures

Examples of closed figures are shown in figure 9.

The POLYGON SET example shown in figure 30 may also be obtained using the closed figure:

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
        POLYLINE (P3, P1, P2)
        NEW REGION {see note 1}
        POLYLINE (P4, P5, P6, P4)
END FIGURE
```

NOTE 1 Invisible implicit boundary portion P2-P3 generated.

Figure 9a shows the closed figure resulting from interpretation of the elements listed below.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
POLYLINE (P1, P2)
        CIRCULAR ARC 3 POINT (P2, P3, P4)
        POLYLINE (P4, P5)
        CIRCULAR ARC 3 POINT (P5, P6, P1)
END FIGURE
```

Figure 9a could also be the result of interpreting the following sequence of elements which include CONNECTING EDGE.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
        CIRCULAR ARC 3 POINT (P2, P3, P4)
        CONNECTING EDGE
```

```
        CIRCULAR ARC 3 POINT (P5, P6, P1) {see note 2}
        CONNECTING EDGE
    END FIGURE {see note 3}
```

NOTE 2 Visible edge portion P4..P5 generated.

NOTE 3 Visible edge portion P1..P2 generated.

Figure 9b shows the closed figure resulting from interpretation of the elements listed below.

```
    EDGE VISIBILITY (ON)
    BEGIN FIGURE
        POLYLINE (P1, P2, P3, P4)
        CIRCULAR ARC 3 POINT (P4, P5, P1)
        EDGE VISIBILITY (OFF)
        NEW REGION
        CIRCULAR ARC CENTRE (P7, 1, 0, 1, 0, |P7 - P5|) {See Note 4}
    END FIGURE
```

NOTE 4 P7 is the mid-point of the line segment between P5 and P6.

Figure 9c shows the closed figure resulting from interpretation of the elements listed below.

```
    BEGIN FIGURE
        CIRCULAR ARC CENTRE (P1, 1, 0, 1, 0, |P3 - P1|)
        NEW REGION
        CIRCULAR CENTRE (P1, 1, 0, 1, 0, |P2 - P1|)
    END FIGURE
```

Figure 9c could also be the result of interpreting the following sequence of elements which include fill area elements.

```
    BEGIN FIGURE
        CIRCLE (P1, |P3 - P1|)
        CIRCLE (P1, |P2 - P1|)
    END FIGURE
```

Figure 9d shows the use of ELLIPTICAL ARC to draw a box with rounded corners and is the result of interpreting the sequence of elements shown below.

```
    EDGE VISIBILITY (ON)
    BEGIN FIGURE
        ELLIPTICAL ARC (P1, P2, P3, (1,0), (0,1))
        CONNECTING EDGE
        ELLIPTICAL ARC (P4, P5, P6, (0,1), (-1,0)) {see note 5}
        CONNECTING EDGE
        ELLIPTICAL ARC (P7, P8, P9, (-1,0), (0,-1))
        CONNECTING EDGE
        ELLIPTICAL ARC (P10, P11, P12, (0,-1), (1,0))
        CONNECTING EDGE
    END FIGURE {see note 6}
```

NOTE 5 Visible edge portion P2..P5 generated; edge portions P6.8 and P9.11 are drawn with the next two arcs.

NOTE 6 Visible edge portion P12..P3 generated.

Figure 9e shows the use of CIRCULAR ARC 3 POINT to create an ‘S' shape and is the result of interpreting the sequence of elements shown below.

```
    EDGE VISIBILITY (ON)
    BEGIN FIGURE
        CIRCULAR ARC 3 POINT (P1, P2, P3)
```

```
        CIRCULAR ARC 3 POINT (P3, P4, P5)
        CONNECTING EDGE
        CIRCULAR ARC 3 POINT (P6, P7, P8) {see note 7}
        CIRCULAR ARC 3 POINT (P8, P9, P10)
        CONNECTING EDGE
    END FIGURE {see note 8}
```

NOTE 7 Visible edge portion P5..P6 generated.

NOTE 8 Visible edge portion P10..P1 generated.

Figure 9f shows the closed figure resulting from interpretation of the elements listed below. It is similar to figure 9d, but makes use of changing the edge attributes between successive occurrences of CONNECTING EDGE.

```
    EDGE VISIBILITY (ON)
    BEGIN FIGURE
        ELLIPTICAL ARC(P1, P2, P3, (1,0), (0,1))
        CONNECTING EDGE
        EDGE TYPE(SOLID)
        ELLIPTICAL ARC(P4, P5, P6, (0,1), (-1,0)) {see note 9}
        EDGE TYPE(DASHED)
        CONNECTING EDGE
        ELLIPTICAL ARC(P7, P8, P9, (-1,0), (0,-1)) {see note 10}
        EDGE TYPE(SOLID)
        CONNECTING EDGE
        ELLIPTICAL ARC(P10, P11, P12, (0,-1), (1,0))
        EDGE TYPE(DASHED)
        CONNECTING EDGE
    END FIGURE {see note 11}
```

NOTE 9 No edge portion P2..P5 generated.

NOTE 10 Visible (dashed) edge portion P6..P8 generated; solid edge portion P9..P11 drawn with the next arc.

NOTE 11 Visible (dashed) edge portion P12..P3 generated.

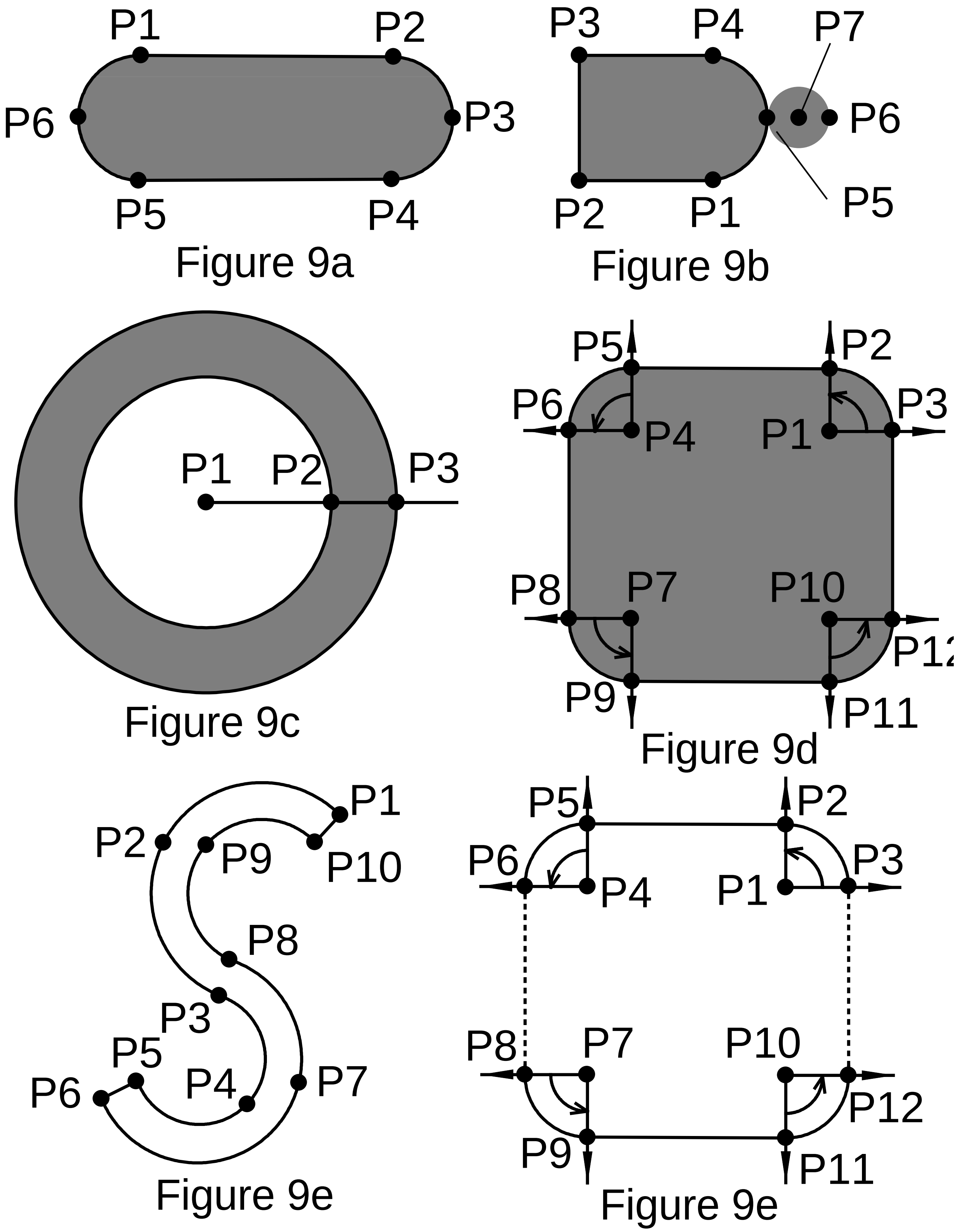


Figure 9 — Examples of closed figures

6.6.12 Symbol elements

6.6.12.1 Description

This part of ISO/IEC 8632 defines mechanisms to access external symbol libraries and include their symbols in the metafile by reference. There is one symbol primitive element.

POLYSYMBOL	generates a symbol which will be sized and oriented according to the symbol attributes and placed with its reference point coinciding with each point in a specified list of position points.

6.6.12.2 Attributes

The selection, sizing and placement of symbols is specified by the attribute elements SYMBOL SIZE, SYMBOL COLOUR, SYMBOL ORIENTATION, and SYMBOL LIBRARY INDEX. These are further described in 6.7.9.

6.6.12.3 Transformation of symbol elements

The vectors specified by the SYMBOL ORIENTATION element (see 6.7.9) are subject to the VDC-to-Device mapping (see 6.4.7) as well as to both segment and copy transformation (see 6.10.4.2 and 6.10.5).

6.7 Attribute elements

Attribute elements determine the appearance of graphical primitive elements. Attributes are classified as either individual attributes or attributes that may be bundled. Table 2 lists the attributes by this classification.

Bundled selection of attributes implies that the appearances of graphical primitive elements are distinguishable from one another when different bundles are specified. The method of specification of the aspects of a graphical primitive element that may be bundled may be chosen separately for each aspect. A further group of attributes called ASPECT SOURCE FLAGS (ASFs) takes the one of values ‘individual' or ‘bundled' to specify the choice. There is one ASF for each aspect that may be bundled.

There is a current modal value for every attribute. Elements are provided to change these modal values. The modal value established by setting an attribute remains until it is explicitly changed. All attributes return to their default values when the BEGIN PICTURE element is encountered.

There is at least one bundle index associated with each of the graphical primitive element types line, marker, filled area, and text. Line, marker, and text elements have a single associated bundle index. Filled-area elements have two associated bundle indexes, one for interior attributes and one for edge attributes.

The value of each bundle index attribute is modally bound to subsequent graphical primitive elements of the associated type. Distinct values of the bundle index correspond to distinct appearances of the graphical primitive element.

For individual attributes, the current modal value is used to display a graphical primitive element. For attributes that may be bundled a graphical primitive element is displayed as follows:

a) if the ASF for an aspect is ‘individual', the value used is the current modal value (which is set only by the individual aspect-setting elements);

b) if the ASF for an aspect is ‘bundled', the value used is obtained via the bundle table for that primitive; the corresponding component of the bundle, which is pointed to by the bundle index, is used.

Table 2 — Individual attributes and attributes that may be bundled

Individual	**May Be Bundled**
CHARACTER HEIGHT CHARACTER ORIENTATION TEXT PATH TEXT ALIGNMENT	LINE TYPE LINE WIDTH LINE COLOUR MARKER TYPE

Individual	May Be Bundled
CHARACTER SET INDEX ALTERNATE CHARACTER SET INDEX EDGE VISIBILITY FILL REFERENCE POINT PATTERN SIZE PICK IDENTIFIER LINE CAP LINE JOIN LINE TYPE CONTINUATION LINE TYPE INITIAL OFFSET TEXT SCORE TYPE RESTRICTED TEXT TYPE INTERPOLATED INTERIOR EDGE CAP EDGE JOIN EDGE TYPE CONTINUATION EDGE TYPE INITIAL OFFSET SYMBOL LIBRARY INDEX SYMBOL COLOUR SYMBOL SIZE SYMBOL ORIENTATION	MARKER SIZE MARKER COLOUR TEXT FONT INDEX TEXT PRECISION CHARACTER EXPANSION FACTOR CHARACTER SPACING TEXT COLOUR INTERIOR STYLE FILL COLOUR HATCH INDEX PATTERN INDEX EDGE TYPE EDGE WIDTH EDGE COLOUR

The resulting appearance is interpreter dependent, but the intent is that the interpreter render distinct appearances of graphical primitive elements for distinct values of the associated bundle index (or indexes) by manipulation of the attributes that may be bundled. For example, LINE BUNDLE INDEX designates visually distinct combinations of the polyline attributes LINE WIDTH, LINE TYPE, and LINE COLOUR. Table 3 lists the aspects of each bundle.

Because inquiry of bundle representations is not generally possible in a metafile environment, mixing of 'individual' and 'bundled' ASF values within a bundle will compromise the guarantee of distinguishability of different bundle indexes within that bundle at interpretation time.

In addition to the attribute elements listed in table 3, there are elements permitting the definition of "compound attributes" from several of the graphical primitive elements. The following compound attribute is defined: compound text path. The elements that may be used to specify this compound attribute are listed in table 4.

Table 3 — Aspects of the bundle and affected primitives

Bundle	Aspects	Affected primitives
LINE	LINE TYPE LINE WIDTH LINE COLOUR	POLYLINE DISJOINT POLYLINE CIRCULAR ARC 3 POINT CIRCULAR ARC CENTRE ELLIPTICAL ARC CIRCULAR ARC CENTRE REVERSED CONNECTING EDGE HYPERBOLIC ARC PARABOLIC ARC NON-UNIFORM B-SPLINE NON-UNIFORM RATIONAL B-SPLINE POLYBEZIER
MARKER	MARKER TYPE MARKER SIZE MARKER COLOUR	POLYMARKER

Bundle	Aspects	Affected primitives
FILL	INTERIOR STYLE FILL COLOUR HATCH INDEX PATTERN INDEX	POLYGON POLYGON SET RECTANGLE CIRCLE CIRCULAR ARC 3 POINT CLOSE CIRCULAR ARC CENTRE CLOSE ELLIPSE ELLIPTICAL ARC CLOSE
EDGE	EDGE TYPE EDGE WIDTH EDGE COLOUR	POLYGON POLYGON SET RECTANGLE CIRCLE CIRCULAR ARC 3 POINT CLOSE CIRCULAR ARC CENTRE CLOSE ELLIPSE ELLIPTICAL ARC CLOSE
TEXT	TEXT FONT INDEX TEXT PRECISION CHARACTER EXPANSION FACTOR CHARACTER SPACING TEXT COLOUR	TEXT RESTRICTED TEXT APPEND TEXT

Table 4 — Contributing primitives to compound attributes

Compound Attribute	First Element	Primitives Included	Other Elements	Final Element
Compound Text Path	BEGIN COMPOUND TEXT PATH	Line primitives (1) GDP(2)	None	END COMPOUND TEXT PATH

NOTE 1 All primitives of the identified classes may be included.

NOTE 2 Whether a GDP element may contribute to the compound text path and whether or how it specifies that the compound path state be opened, maintained, or closed is specified with the definition of the GDP in the International Register of Graphical Items.

6.7.1 Line attributes

6.7.1.1 Line bundle

The LINE BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) LINE TYPE: determines the type of the line (for example, ‘dotted', ‘dashed', etc.) with which the line is rendered;

b) LINE WIDTH: determines the width of the line with which the line is rendered;

c) LINE COLOUR: determines the colour in which the line is rendered.

6.7.1.2 Individual line attributes

In addition to the line attributes which may be bundled there are a number of individual line attributes.

a) LINE CAP specifies the appearance of the endpoints of line elements as well as the endpoints of individual dashes when dashed lines are rendered. The following cap styles are supported:

unspecified	no specific treatment is required;
butt	the line is squared off at the endpoint, there is no projection beyond the endpoint;
round	a semicircular arc with diameter equal to the line width is drawn around the endpoint and filled in (the drawn line thus projects beyond the endpoint);
projecting square	the line is squared off at a distance equal to half the line width beyond the endpoint;
triangle	a cap is added to the line which is an equilateral triangle, the length of whose side equals the line width.

NOTE 1 The LINE CAP element is only permissible in Version 3 and Version 4 metafiles, therefore only the 'unspecified' style is available in Version 1 and Version 2 metafiles.

These styles may be applied to the open endpoints of line elements those endpoints which demark the beginning or ending of the entire line primitive or to interior endpoints, which correspond to the endpoints of individual dashes when a non-solid line type is in effect. The interior caps must either match the caps on the open endpoints, have butt style, or be interpreter dependent. Figure 10 illustrates the styles of LINE CAP.

b) LINE JOIN specifies the appearance of the interior corners of line elements. Interior corners correspond either to the interior vertices of polyline elements or to the junctions between distinct elements comprising a compound line element. The following styles are supported:

Unspecified	no specific treatment is required;
Mitre	the outer edges of the two adjoining line segments are extended until they meet at a point;
Round	a circular arc with diameter equal to the line width is drawn around the vertex between the adjoining segments and is filled in, producing a rounded corner;
Bevel	the adjoining line segments are terminated with a butt cap, and the resulting triangular notch is filled in.

NOTE 2 The LINE JOIN element is only permissible in Version 3 and Version 4 metafiles, therefore only the 'unspecified' style is available in Version 1 and Version 2 metafiles.

For the style 'mitre', the rendering of the line join is affected by the MITRE LIMIT Control Element. Figure 10 illustrates the defined styles of LINE JOIN.

Both line caps and line joins behave as does line width with respect to transformation. If the value of LINE WIDTH SPECIFICATION MODE is absolute then conceptually the line cap and join are applied to the line in VDC space before any transformations are applied and they are subject to all transformations associated with the line. Otherwise, they are applied to the line in device space, conceptually after all associated transformations have been applied, and are immune to all transformations.

c) LINE TYPE CONTINUATION provides control of the behaviour of non-solid line types at interior vertices and junctions of line elements. The following behaviours may be selected:

Unspecified	no specific treatment is required;
Continue	the style is continued without interruption across vertices;
Restart	the style is restarted at each vertex;

adaptive continue the style is continued, but each vertex must be "inked" including vertices at the ends of the line primitive which might otherwise not be drawn because of a non-solid line type.

NOTE 3 The LINE TYPE CONTINUATION element is only permissible in Version 3 and Version 4 metafiles, therefore only the 'unspecified' style is available in Version 1 and Version 2 metafiles.

d) LINE TYPE INITIAL OFFSET allows control of how much of the first cycle of a non-solid line type to omit before drawing commences for a line primitive. It is specified as a fraction of one full cycle.

The Picture Descriptor element LINE AND EDGE TYPE DEFINITION allows the precise definition of the solid/gap sequences which comprise a line or edge type. A definition is associated with an index by this element, and this index may be refered to within the picture by LINE TYPE and EDGE TYPE elements.

6.7.2 Marker attributes

6.7.2.1 Marker bundle

The MARKER BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) MARKER TYPE: determines the marker symbol that is drawn at the marker position (for example, 'dot', 'plus', etc.);

b) MARKER SIZE: determines the size of the marker symbol;

c) MARKER COLOUR: determines the colour in which the marker symbol is drawn.

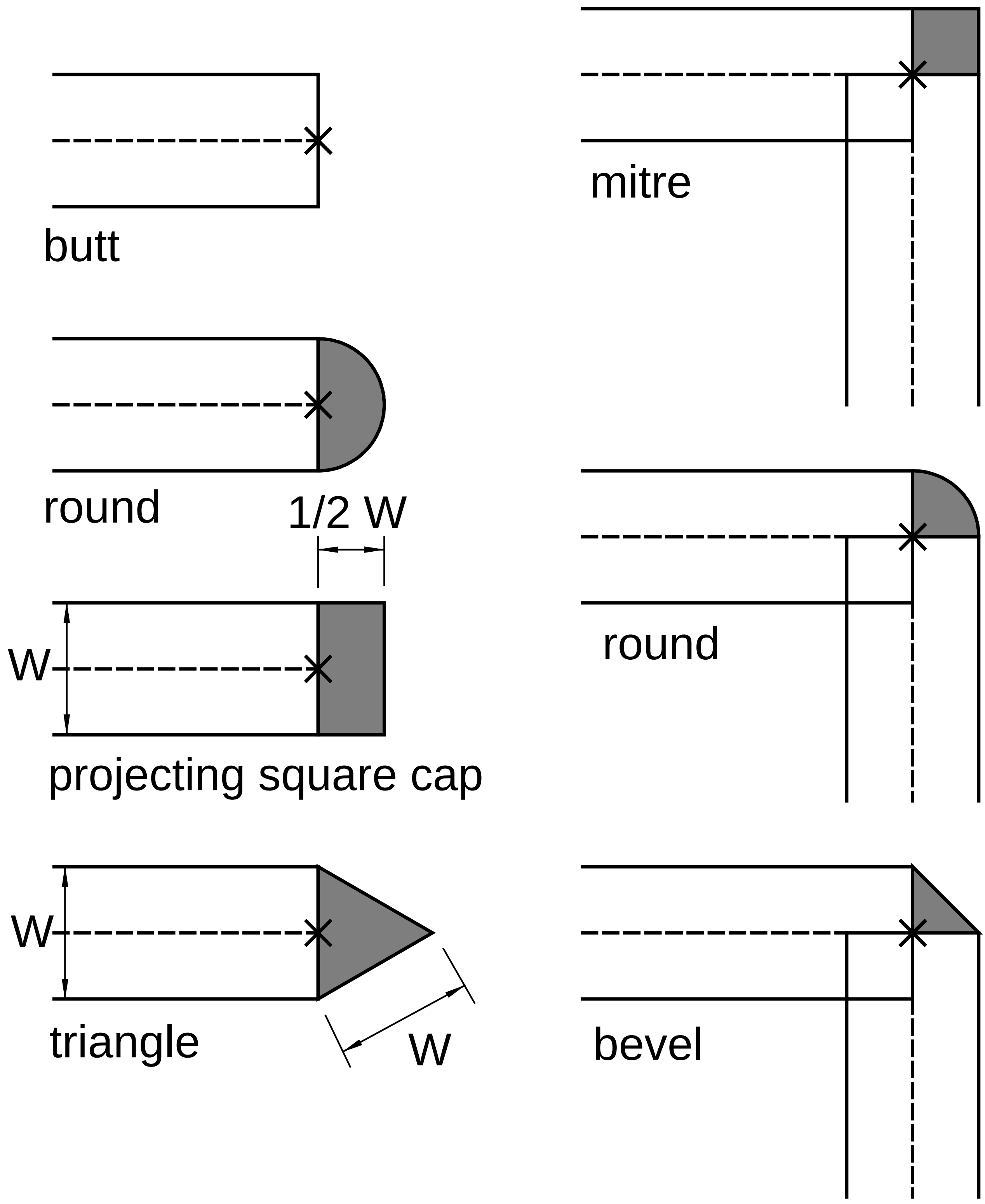


Figure 10 — Examples of LINE CAP and LINE JOIN

6.7.2.2 Individual marker attributes

There are no individual marker attributes in metafiles of Versions 1, 2, 3 and 4 — all marker elements are bundled.

6.7.3 Text attributes

6.7.3.1 Text bundle

The TEXT BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) TEXT FONT INDEX: determines the style of the graphical display of the text characters;

b) TEXT PRECISION: determines the fidelity with which characters need be displayed and positioned;

c) CHARACTER EXPANSION FACTOR: determines the deviation of the character width/height ratio from the ratio established by the font designer;

d) CHARACTER SPACING: determines the amount of blank space added between characters in a string;

e) TEXT COLOUR: determines the colour in which the text characters are drawn.

6.7.3.2 Individual text attributes and usage of text attributes

The representation and placement of text characters on a device is controlled by the attribute elements TEXT FONT INDEX, CHARACTER SET INDEX, ALTERNATE CHARACTER SET INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOUR, CHARACTER HEIGHT, TEXT SCORE TYPE, and by the control elements AUXILIARY COLOUR and TRANSPARENCY. The placement and orientation of text strings is controlled by the attribute elements CHARACTER ORIENTATION, TEXT PATH, TEXT ALIGNMENT, RESTRICTED TEXT TYPE, and by the compound text path and the control element GENERALIZED TEXT PATH MODE. TEXT BUNDLE INDEX is an index into the text bundle table, each entry of which contains values for the attributes that may be bundled. Although the placement and size of text can be precisely specified by the attributes mentioned, the fidelity of rendering depends on the current TEXT PRECISION.

The choice of character font (that is, the style of the characters to be displayed) is determined independently of the character set. However, the specified font will only have meaning if it is related to the character set being used. Times Roman and Courier Bold Oblique are examples of commonly used fonts for Latin-based alphabets.

The attributes in the character representation and placement group (above) and TEXT BUNDLE INDEX may be changed within a string. A TEXT element or RESTRICTED TEXT element is tagged to show whether or not it is complete. If not complete, the element provides only the first portion of the string. The TEXT element or RESTRICTED TEXT element may be followed by the desired text attribute element(s) and then by an APPEND TEXT element, which provides the next portion of the string. This may be repeated as often as necessary, with the final APPEND TEXT tagged to indicate that the string is complete. Note that a metafile interpreter generally cannot display any of the text until the string is complete because of TEXT ALIGNMENT and the way in which attribute changes affect the definition of the text extent rectangle (see below). Text may be displayed before the string is complete only in the cases shown in table 5.

GENERALIZED PATH TEXT MODE has the possible values ‘off', ‘non-tangential', and ‘axis-tangential'. If the mode is ‘off', then the writing direction will be as specified by the TEXT PATH element ‘right', ‘left', ‘up', or ‘down'.

Table 5 — Cases allowing display of partial text

Path	Vertical Alignment	Horizontal Alignment
Right	normal vertical or baseline	normal horizontal, left or continuous (0,0)
Left	normal vertical or baseline	normal horizontal, right, or continuous (1,0)

down	top, capline, normal vertical, or continuous (0,1)	normal horizontal or centre
up	baseline, bottom, normal vertical, or continuous (0,0)	normal horizontal or centre

NOTE 1 The four values of TEXT PATH define four special cases of the generalized text path, with paths which are straight lines pointing from the text position point in four indicated directions. For any of the four values of TEXT PATH when the generalized text path mode is ‘off', the same effect can be achieved with a straight path and appropriate attribute settings in ‘axis-tangential' mode.

When GENERALIZED TEXT PATH MODE is ‘non-tangential' or ‘axis-tangential', the string is laid out along the current text path as specified between the preceding BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements and positioned at the text position point according to the TEXT ALIGNMENT. The orientation of the characters along the path will depend on the mode. If the mode is ‘non-tangential', the characters are positioned along the path and oriented as indicated by the character orientation vectors each character has the same orientation regardless of the path direction. If the mode is ‘axis-tangential', the characters are positioned along the path and for each character the character orientation vectors are rotated by the angle defined by the tangent to the path at the character's position the orientation of each character depends upon the path direction at the character's placement point. In all cases, the angle between the up and base vectors of CHARACTER ORIENTATION is preserved and both the angle and the ratio of the vector lengths have the same effect as for mode ‘off' (see figure 21, figure 22, and figure 23).

When the GENERALIZED TEXT PATH MODE is ‘off' the escapement between characters will depend on TEXT PATHand will be computed by adding the width of the character to the character spacing if TEXT PATH is ‘left' or ‘right' and by adding the distance between the top and bottom lines of the character to the character spacing if the TEXT PATH is ‘up' or ‘down'.

When the GENERALIZED TEXT PATH MODE is ‘non-tangential', or ‘axis-tangential' the escapement will be computed by adding the width of the character to the character spacing. This escapement will be along the compound text path.

The characters are sized according to the CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR and are oriented according to CHARACTER ORIENTATION. The direction of character placement in the string relative to CHARACTER ORIENTATION is along the path defined within the scope of the preceding compound path definition. If the string length exceeds the length of the path, the characters of the string will continue to be placed along the path defined by a vector whose tail is the last point of the path and whose direction is the direction of the path at the last point.

TEXT ALIGNMENT is applied as follows for generalize text path. Conceptually the text is positioned along the defined path, determining the text alignment relative to the defined path, and applying other relevant text attributes. The path/text assembly is then aligned with the position point of the text element. This is illustrated in figure 24 and figure 25. A "text extent band" is defined whose length is the larger of the arc length of the defined path and the length of the text as it has been placed along the defined path, and whose height in the direction of the character up vector has the character body height and alignment reference points defined in figure 11. This tube is then aligned relative to the text position point.

NOTE 2 If anisotropic transformations are in effect, then the height of the text will change for paths other than straight lines.

There are several methods for inclusion of characters from different character sets within a string. The method used is determined by the CHARACTER CODING ANNOUNCER Metafile Descriptor element. The default or normal technique is to use the CHARACTER SET INDEX element and restrict the contents of the text strings to printing characters and spaces (format effector control codes such as CR and LF are permitted, but their interpretation is implementation dependent). Other settings of the CHARACTER CODING ANNOUNCER or use of the ALTERNATE CHARACTER SET INDEX element permit standardized use of 8-bit characters and the SI, SO, and ESC control codes within the text string, in accordance with ISO 2022.

NOTE 3 SI and SO are only defined and usable in 7-bit coding.

The ALTERNATE CHARACTER SET INDEX element is used to select a character set to be used as both the G1 set and the G2 set. The G1 set is used both for 8-bit characters in columns 10 to 15 of the code table, and with the SO control code. The assignment of meaning to the index parameter of both CHARACTER SET INDEX and ALTERNATE CHARACTER SET INDEX is done with the CHARACTER SET LIST Metafile Descriptor element.

Selection of fonts from font tables is done by the TEXT FONT INDEX element. The assignment of meaning to the index values of TEXT FONT INDEX is done with the Metafile Descriptor element FONT LIST.

The font coordinate system is illustrated in figure 11. The character body encloses all of the drawn parts (kerning excepted) of all characters in the font (that is, no descender extends lower than ‘bottom', and no accent mark or oversized glyph extends higher than ‘top'). The left and right edges of the character body may be defined on a per-character basis to accommodate variable widths and proportional spacing. It is expected that font designers will specify some fonts having kerns extending beyond the character body. The body exceeds the actual character width and height as necessary to provide adequate white space between characters, such that text is readable and adequately separated when adjacent character bodies are flush (that is, when CHARACTER SPACING is 0). The character body is defined in this way to permit alignment of multi-line text without overlaps in the metafile environment. The CHARACTER HEIGHT specifies the VDC distance between the capline and baseline of the font (see figure 11). The CHARACTER EXPANSION FACTOR specifies the deviation of the width to height ratio of the characters from the ratio indicated by the font designer (see figure 12). CHARACTER SPACING specifies how much additional space is to be inserted between two adjacent character bodies (see figure 13). If the value of CHARACTER SPACING is zero, the character bodies are arranged one after the other along the TEXT PATH with only the intercharacter spacing designated by the font designer. If the value of CHARACTER SPACING is positive, additional space is inserted between character bodies. If the value of CHARACTER SPACING is negative, adjacent character bodies overlap although the characters themselves might not. Character spacing is specified as a fraction of the CHARACTER HEIGHT.

CHARACTER ORIENTATION specifies the character up vector and base vector, which fix the orientation, skew, and distortion of the characters, and also determine the sense of ‘right', ‘left', ‘up', and ‘down' for TEXT PATH and TEXT ALIGNMENT (see figure 14).

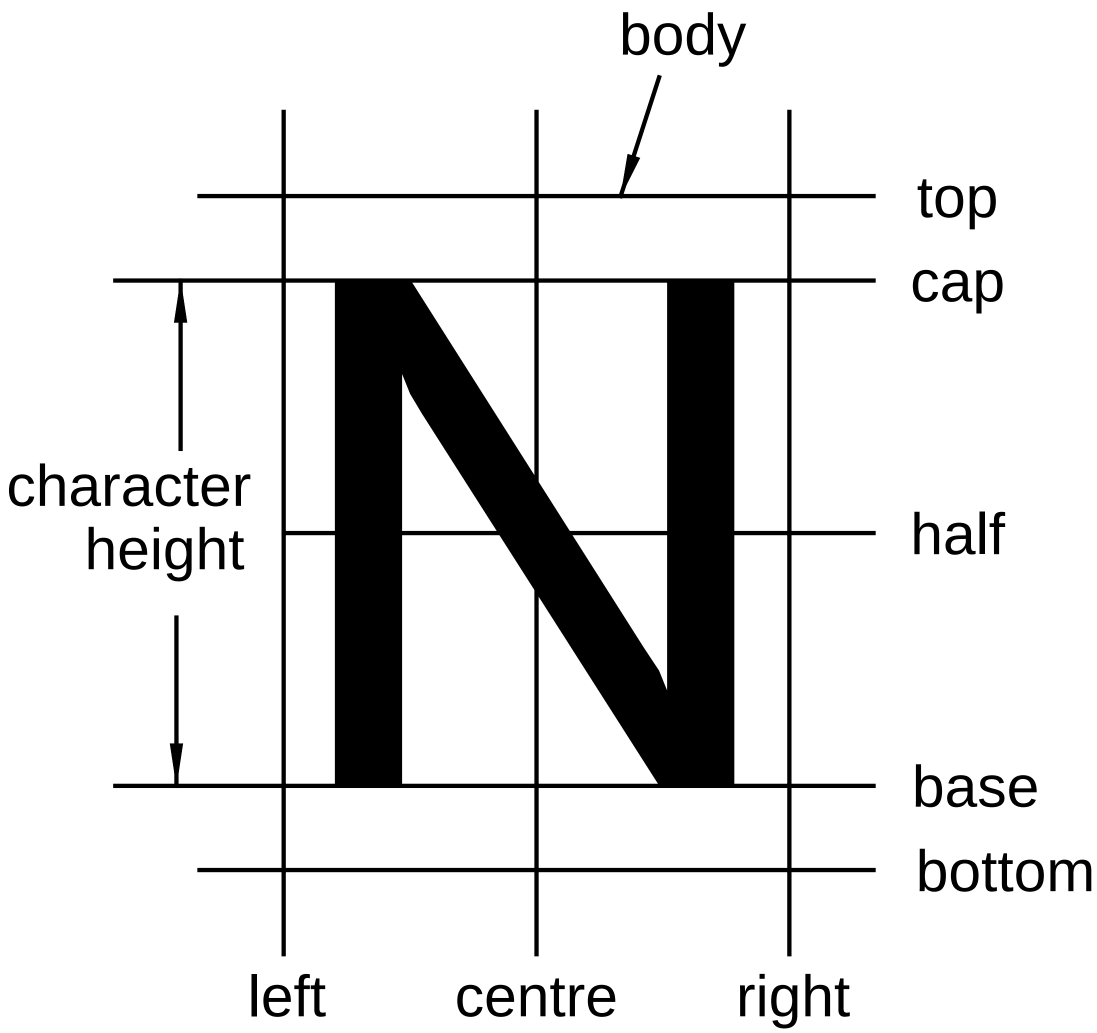


Figure 11 — Font description coordinate system

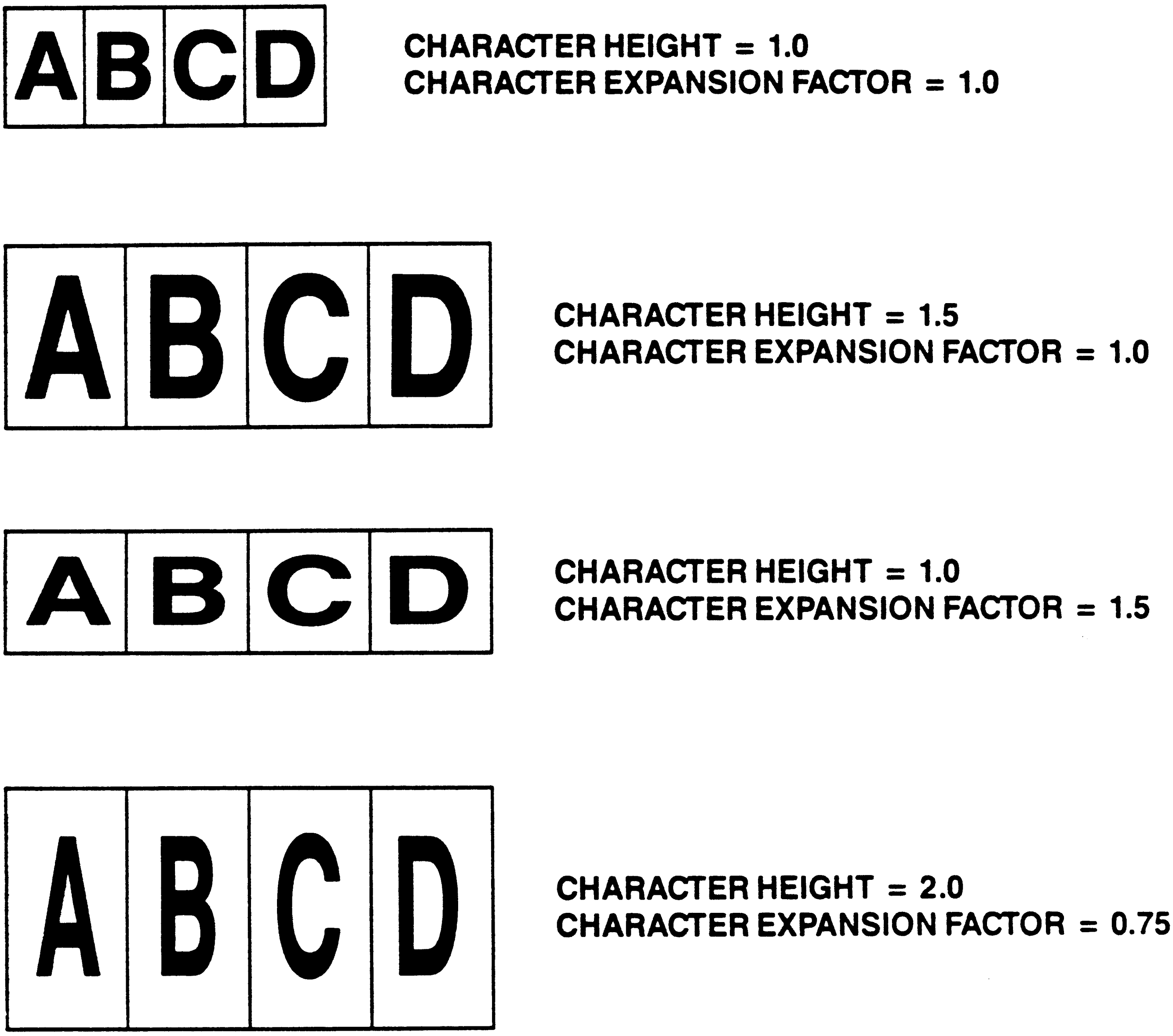


Figure 12 — CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR

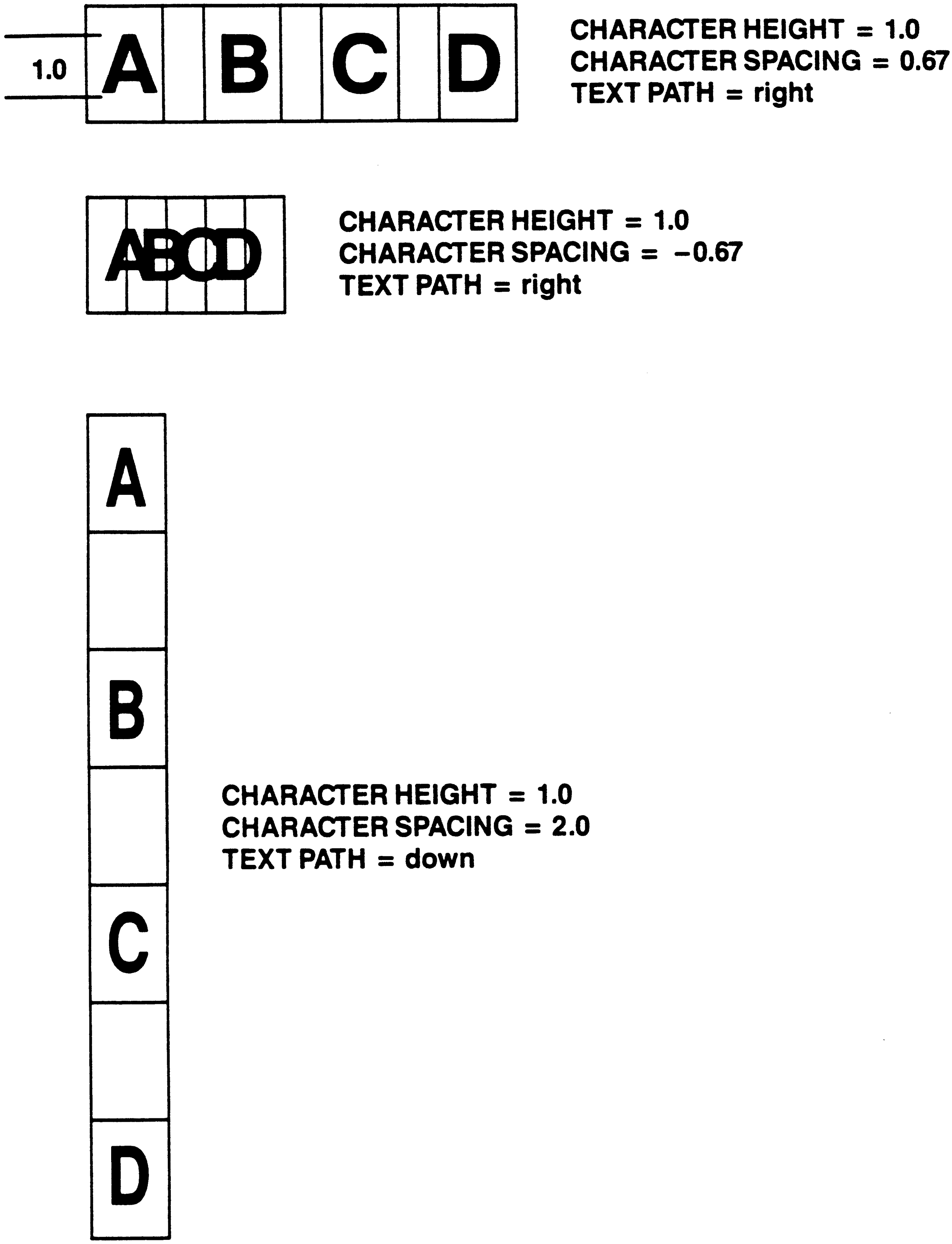


Figure 13 — CHARACTER SPACING

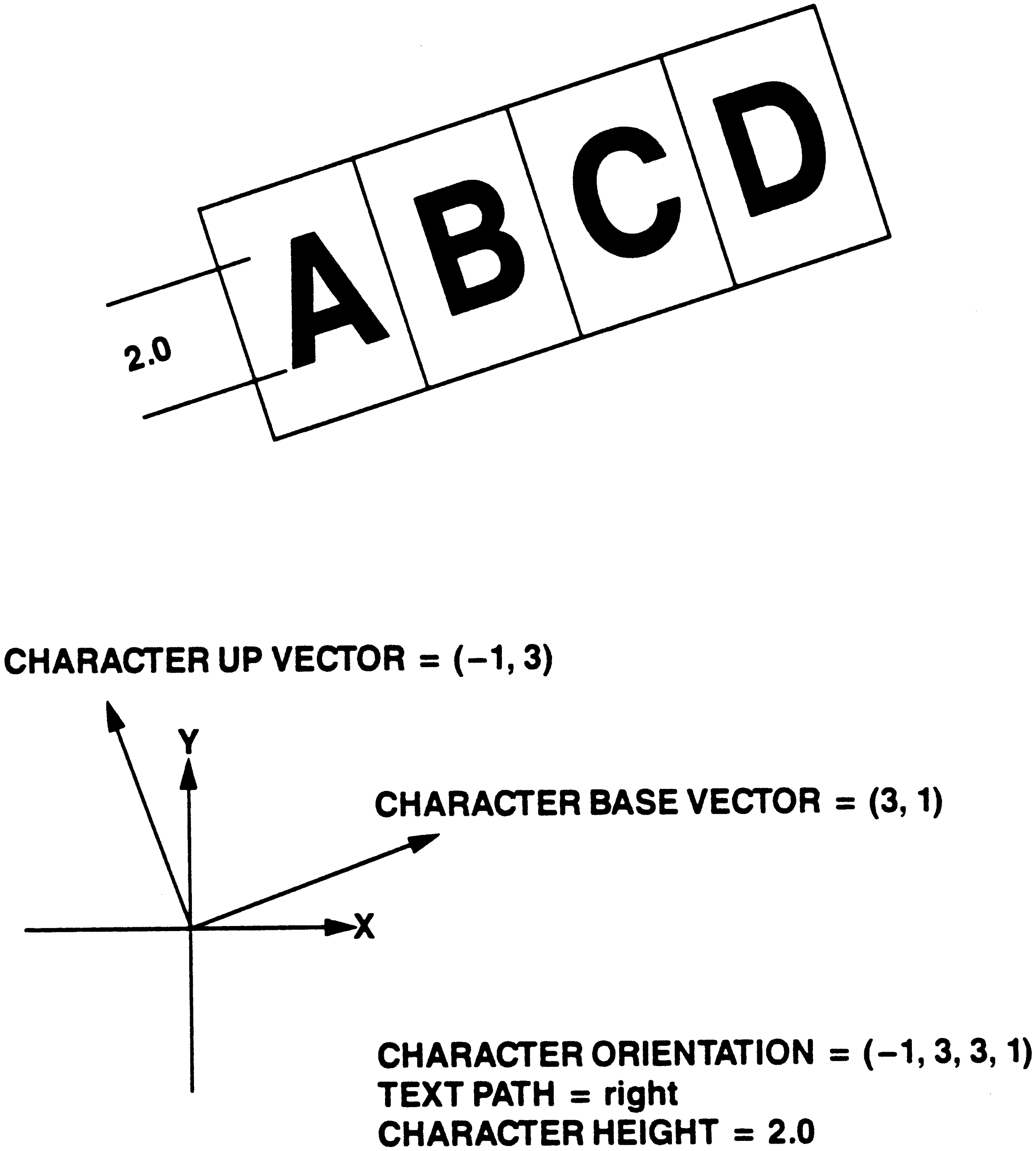


Figure 14 — CHARACTER ORIENTATION

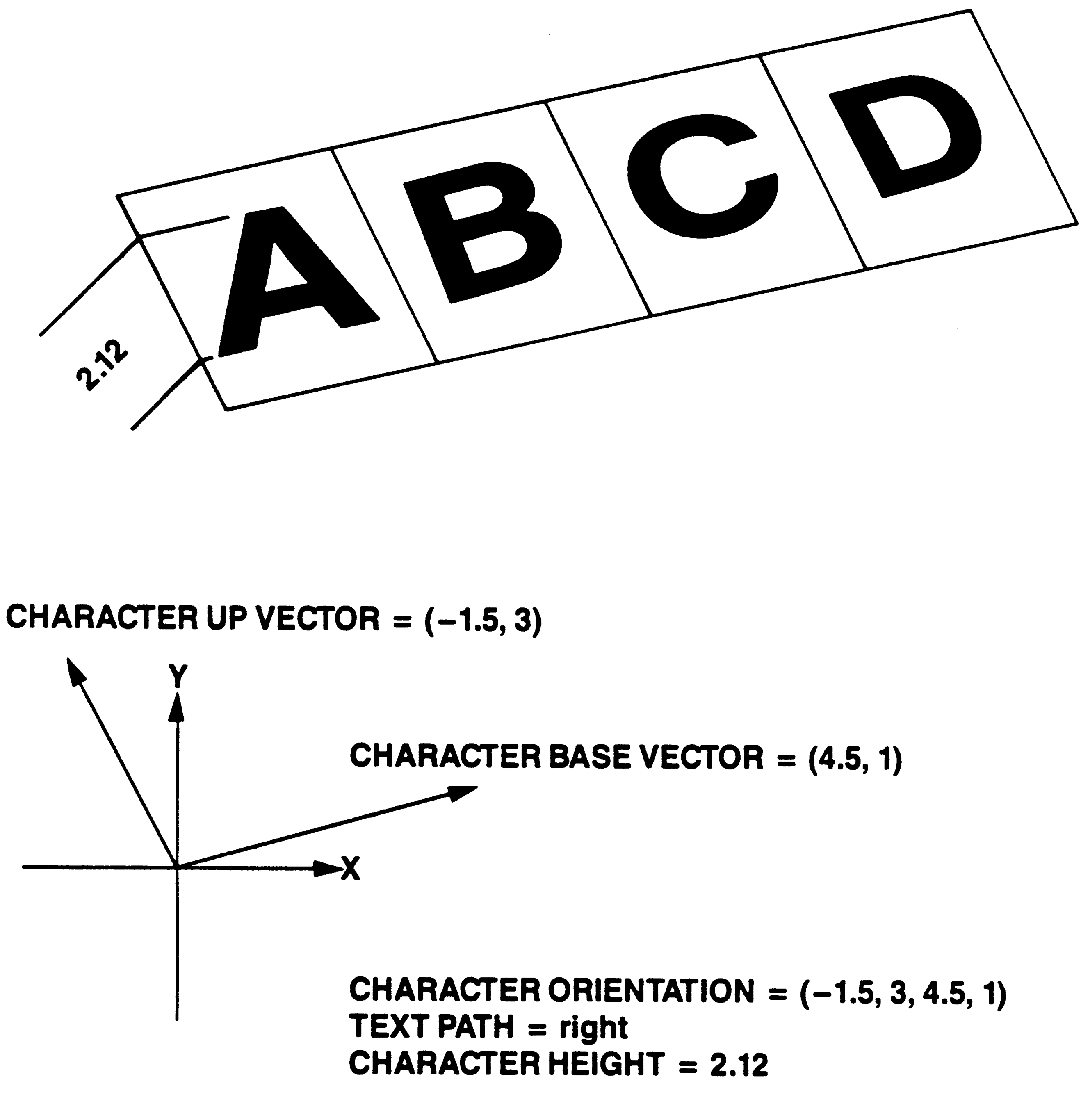


Figure 15 — CHARACTER HEIGHT and CHARACTER ORIENTATION after anisotropic transformation

The way in which software above the metafile generator and/or the metafile generator itself may use CHARACTER ORIENTATION is described. To generate the CHARACTER ORIENTATION and CHARACTER HEIGHT elements a vector whose length is the character height (baseline-to-capline) and whose direction is the desired character up vector is created. A second vector is also created with the same length, whose direction is negative 90° from the up vector. This pair of vectors may be transformed before being given to the metafile generator as the parameters to CHARACTER ORIENTATION. The length of the transformed up vector may then be used to generate the CHARACTER HEIGHT element. If an anisotropic transformation is in effect above the metafile generator, the character height must be respecified by the metafile generator for each change in orientation (see figure 15). The CHARACTER HEIGHT and CHARACTER ORIENTATION are decoupled to permit changing character height (but not orientation) within a string. Thus, to the metafile interpreter the absolute lengths of the vectors in CHARACTER ORIENTATION are not significant; only their directions and the ratio of their lengths are significant.

The ratio of the length of the width vector to the length of the height vector is used to scale the CHARACTER SPACING for text paths ‘right' and ‘left', and the CHARACTER EXPANSION FACTOR in all cases, before these are used to display the text.

TEXT PATH has the possible values ‘right', ‘left', ‘up', and ‘down'. It specifies the writing direction of the text string as follows:

Right	means the direction of the character base vector;
Left	means 180° from the character base vector;
Up	means the direction of the character up vector;
Down	means 180° from the character up vector.

For the ‘up' and ‘down' text path directions, the characters are arranged so that the centres of the character bodies are on a straight line in the direction of the up vector of CHARACTER ORIENTATION. For the ‘left' and ‘right' text path directions, the characters are arranged so that the baselines of the characters are on a straight line parallel to the direction of the character base vector. These composition rules also hold true when characters of different heights, expansion factors, fonts, or precisions are intermixed in a string by means of attribute changes between non-final TEXT elements and subsequent APPEND TEXT elements.

Alignment of text is done with respect to a text extent rectangle, which is derived by joining the character bodies of the characters in the string according to the current status of the attributes and the composition rules described. Alignment is performed according to the highest precision in the string.

For TEXT PATH = ‘left' or ‘right',

TOPLINE:	topline farthest from the baseline
CAPLINE:	capline farthest from the baseline
HALFLINE:	halfline farthest from the baseline
BOTTOMLINE:	bottomline farthest from the baseline
LEFT:	leftmost edge of leftmost character body
RIGHT:	rightmost edge of rightmost character body
CENTRE:	halfway between left and right edges

For TEXT PATH = ‘up' or ‘down',

TOPLINE:	topline of topmost character
CAPLINE:	capline of topmost character
HALFLINE:	halfway between halflines of topmost and bottommost character
BASELINE:	baseline of bottommost character
BOTTOMLINE:	bottomline of bottommost character
LEFT:	left edge farthest from the centreline
RIGHT:	right edge farthest from the centreline

Note that the relationship of topline to capline, bottomline to baseline, and the placement of the halfline are font-dependent (see figure 16). It is for this reason that the various defining lines of the text extent rectangle need not be derived from the same character body. This is a function of the text height, text font, text precision and character expansion factor changes within a string.

The TEXT ALIGNMENT attribute controls the positioning of the text extent rectangle in relation to the text position (see figure 17).

The horizontal component of TEXT ALIGNMENT has five possible values: 'left', 'centre', 'right', 'normal horizontal' and 'continuous horizontal'. If the horizontal component is 'left', the left side of the text extent rectangle passes through the text position. Similarly, if the value is 'right', the right side of the text extent rectangle passes through the text position. If the horizontal component is 'centre', the text position lies midway between the left and right sides of the text extent rectangle. In this case, if TEXT PATH = 'up' or 'down', the straight line passing through the centrelines of the characters also passes through the text position.

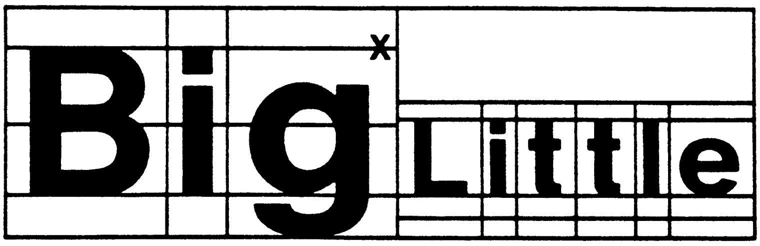
The vertical component of TEXT ALIGNMENT has seven possible values: 'top', 'cap', 'half', 'base', 'bottom', 'normal vertical', and 'continuous vertical'. A vertical alignment value of 'top', 'cap', 'half', 'base', or 'bottom' causes the text to be moved such that the corresponding defining line of the text extent rectangle passes through the text position.

For both horizontal and vertical alignment, normal values are converted to the appropriate value, as indicated in clause 5, at text element elaboration time and thereafter treated as above. For all values of TEXT ALIGNMENT the alignment value applies to the complete text string, which may be comprised of non-final partial strings and a final partial string.

If the value of the horizontal component of TEXT ALIGNMENT is 'continuous horizontal', an additional value, 'continuous horizontal alignment' (a real number normalized so that 1.0 corresponds to the width of the text extent rectangle) is used as an offset from the text position to the left side of the text extent rectangle. Figure 18 illustrates the sense of positive and negative values of 'continuous horizontal alignment'.

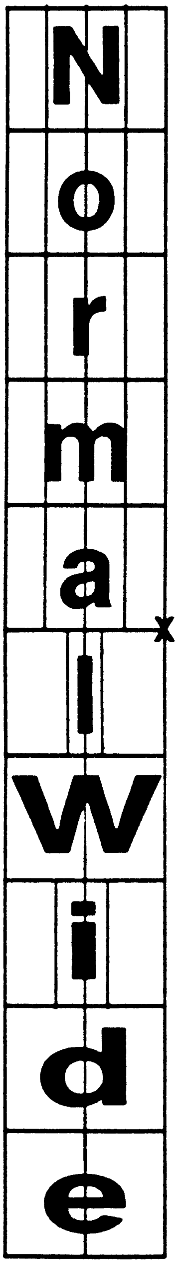
If the value of the vertical component of TEXT ALIGNMENT is 'continuous vertical', an additional value, 'continuous vertical alignment' (a real number normalized so that 1.0 corresponds to the height of the text extent rectangle) is used as an offset from the text position to the bottom side of the text extent rectangle. Figure 18 illustrates the sense of positive and negative values of 'continuous vertical alignment'.

The foregoing examples have been illustrated for the case of the character up vector and the character base vector being orthogonal. When they are not, the text extent rectangle becomes a parallelogram, with the sides remaining parallel to the two orientation vectors. The centreline skews to remain parallel with the left and right edges of the text extent parallelogram. The height of the text extent rectangle is measured along the skewed edge (not perpendicular to the baseline), and the distance to be moved for alignment is done along the angle made by the appropriate orientation vector (see figure 19). Right is in the direction of the character base vector, and left is in the opposite direction.



TEXT ALIGNMENT = (centre, cap, 0, 0)

TEXT PATH = right
CHARACTER HEIGHT = 2.0
String = Big
CHARACTER HEIGHT = 1.0
Appended String = Little



TEXT ALIGNMENT = (right, half, 0, 0)

TEXT PATH = down
CHARACTER HEIGHT = 1.0
CHARACTER EXPANSION FACTOR = 1.0
String = Normal
CHARACTER EXPANSION FACTOR = 2.0
Appended String = Wide

Figure 16 — Discrete text alignment with appended text and proportional spacing

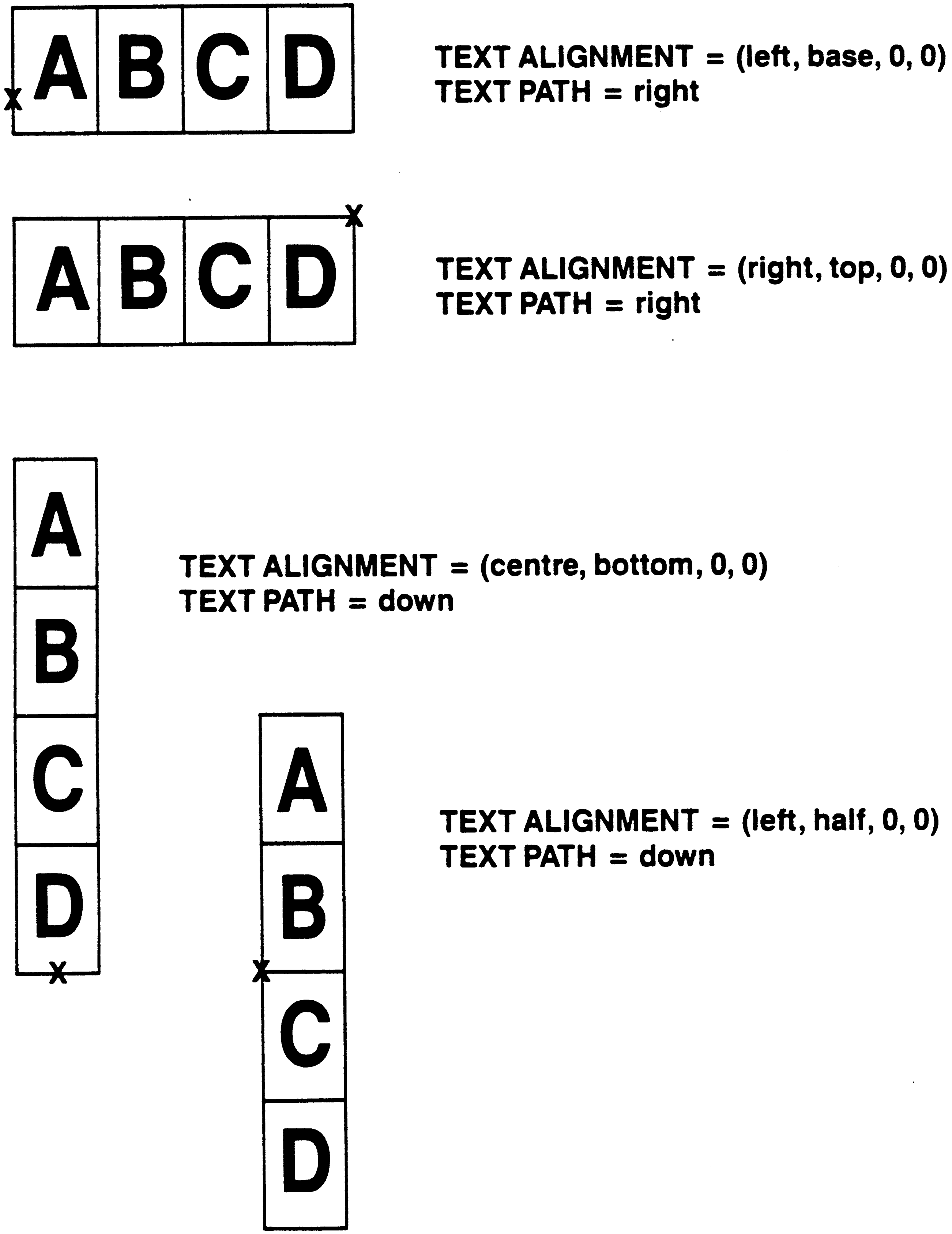
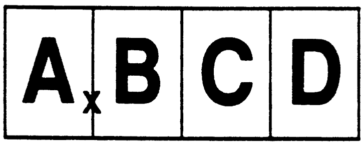
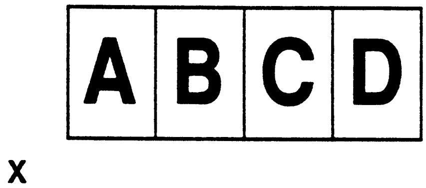


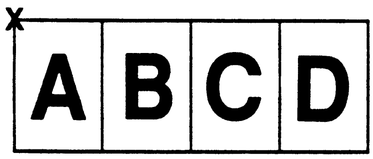
Figure 17 — Discrete text alignment



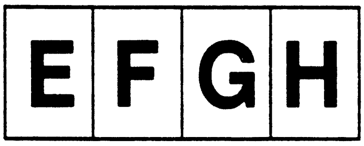
TEXT ALIGNMENT = (continous horizontal, base, 0.25, 0)
TEXT PATH = right



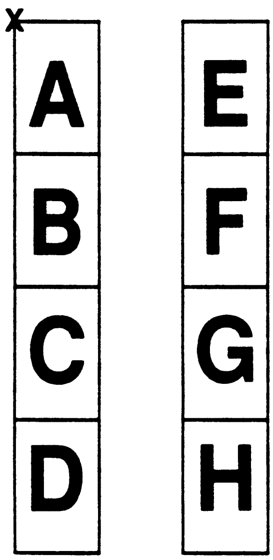
TEXT ALIGNMENT = (continuous horizontal, continuous vertical, −0.25, −0.25)
TEXT PATH = right



TEXT ALIGNMENT = (left, continuous vertical, 0, 1)
TEXT PATH = right
String 1 = ABCD



TEXT ALIGNMENT = (left, continuous vertical, 0, 2.5)
TEXT PATH = right
String 2 = EFGH



TEXT ALIGNMENT = (continuous horizontal, top, 0, 0)
TEXT PATH = down
String 1 = ABCD

TEXT ALIGNMENT = (continuous horizontal, top, -2.0, 0)
TEXT PATH = down
String 2 = EFGH

Figure 18 — Continuous text alignment

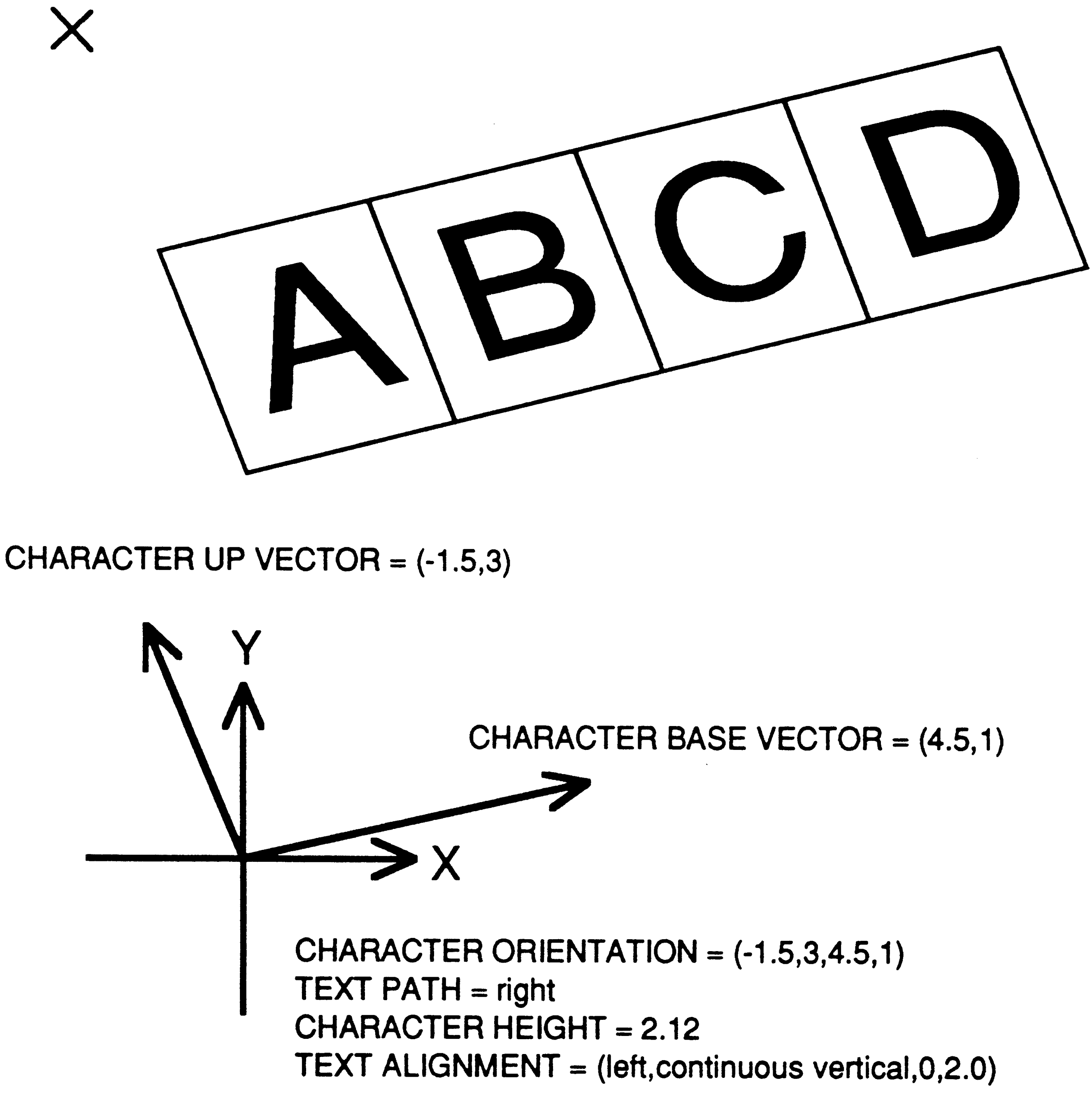


Figure 19 — Continuous text alignment after anisotropic transformation

The continuous values of alignment allow proper positioning of multiple rows or columns of text relative to each other, using a single text position for all of the rows or columns. Rows might typically consist of several lines from a horizontally written character set displayed with ‘right' path. Columns might consist of strings of a vertically written alphabet displayed with ‘down' path.

Such positioning would not otherwise be achievable in a metafile environment, because inquiry of the dimensions of the text extent rectangle cannot be provided at metafile generation time.

EXAMPLE — As an example of the set of the continuous alignment attributes, consider the display of four rows of left-justified text, each consisting of a single string specified by a single TEXT element. To ensure that ascenders and descenders do not interfere between rows and that, in addition, there is a space of at least one-half the maximum size of a character between the descenders of one row with raised accent marks or oversized glyphs of another row, TEXT ALIGNMENT should be set to [‘left', ‘continuous vertical'], and the continuous vertical value should be set to 0.0. Then, output the first row with the text position equal to the lower-left corner of the string. Now, set the continuous vertical value to 1.5, and output the second string with the same text position. This places the second row below the first because of the change in alignment. The last two rows are output in the same way with the continuous vertical value set to 3.0 and 4.5, and the text position parameters to TEXT unchanged. A value of 1.0 assures no overlap between rows; anything greater than 1.0 guarantees additional non-printing space.

TEXT PRECISION is used to specify the ‘closeness' of the text representation at metafile interpretation in relation to that defined by the other metafile text attributes and the clipping currently applicable. The following precision values are defined:

String: The complete text string is generated in the requested text font and is positioned by aligning the string at the given text position. Text height and CHARACTER EXPANSION FACTOR are evaluated as closely as possible given the capabilities of the metafile interpreter. The CHARACTER ORIENTATION (text vectors), TEXT PATH, TEXT ALIGNMENT, and CHARACTER SPACING need not be used. Clipping is done in an implementation dependent fashion.

Char: The complete text string is generated in the requested text font. For the representation of each individual character the aspects text height, the text vectors, and CHARACTER EXPANSION FACTOR are evaluated as closely as possible, given the capabilities of the metafile interpreter. The spacing used between character bodies is evaluated exactly; the character body, for this purpose, is an ideal character body, calculated precisely from the text aspects and the font dimensions. The position of the resulting text extent parallelogram is determined by the TEXT ALIGNMENT and the text position. Clipping is performed at least on a character by character basis.

Stroke: The complete text string in the requested text font is displayed at the text position by applying all text aspects. The text string is clipped exactly at the clipping rectangle.

‘Stroke' precision does not necessarily mean vector strokes; as long as the representation adheres to the rules governing stroke precision, the font may be realized in any form, for example by raster fonts.

TEXT SCORE TYPE is used to specify methods of scoring text strings. Each score type may be ‘off' or ‘on'. Any number of score types may be ‘on' simultaneously.

The TEXT attributes also apply to the RESTRICTED TEXT primitive. Because determination of the text extent of a string is generally not possible in a metafile environment, the RESTRICTED TEXT element has as a parameter the size of a text restriction box. The text restriction box is a parallelogram which is derived from this parameter and the current values of the CHARACTER ORIENTATION and TEXT ALIGNMENT elements.

The simplest allowable interpretation of the restriction box is that all of the specified text string (from the RESTRICTED TEXT element and any associated APPEND TEXT elements) shall fit within the text restriction box, and the text extent of the displayed string shall not exceed the box.

This is not necessarily the most useful interpretation. For example, the common "boxed text" model requires character descenders to exceed the bottom of the box. Version 3 and Version 4 metafiles allow selection of one of several specific ways in which the text shall fit the box (see below). This element is not available in Versions 1 and

2. Any one of these defined methods is a valid interpretation of the RESTRICTED TEXT element for Version 1 metafiles.

If the text string as displayed with the current text attributes would not fit the box as specified, then the values of the text attributes CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT FONT INDEX, TEXT PRECISION, and CHARACTER HEIGHT which are used for the display of this string are adjusted in an implementation dependent manner to achieve the required restriction. The adjustment of attributes pertains only to the restricted string, and is applied conceptually to the "realized" attribute values, i.e., those values that are actually used for the display of the string.

The RESTRICTED TEXT TYPE element specifies the manner in which the string specified with the RESTRICTED TEXT will be restricted. The restriction area may be a box, a parallelogram, or a band according to the character orientation and the generalized text path mode.

When the generalized text path mode is ‘non-tangential', the text restriction band is defined by the area swept out by a vector of length *delta height*, as specified in the RESTRICTED TEXT element, in the direction of the character up vector which traverses along the compound text path. When the generalized text path mode is ‘axis-tangential', the text restriction band is defined by the area swept out by a vector of length *delta height* in the direction of the character up vector rotated by the angle defined by the tangent to the path at that point, which traverses along the compound text path.

For the purpose of describing the effect of RESTRICTED TEXT TYPE, the following terminology is used.

The *horizontal direction* is defined to be: the direction specified by the character base vector of the CHARACTER ORIENTATION element, if the generalized text path mode is ‘off'; along the compound text path (the direction of the tangent to the path at each point), if the generalized text path mode is ‘non-tangential' or ‘axis-tangential'.

The *vertical direction* is defined to be the direction specified by the character up vector of the CHARACTER ORIENTATION element.

The *escapement direction* is defined to be: the horizontal direction, if the generalized text path mode is ‘off' and the value of the TEXT PATH element is ‘left' or ‘right'; the vertical direction, if the generalized text path mode is ‘off' and the value of the TEXT PATH element is ‘up' or ‘down'; along the compound path, if the generalized text path mode is ‘non-tangential' or ‘axis-tangential'.

BOTTOMLINE, BASELINE, CAPLINE, and TOPLINE are defined above in the description of TEXT PATH and TEXT ALIGNMENT. The following text restriction methods are defined:

basic: The text string is constrained not to exceed the text restriction area, and any implementation-dependent realization of this requirement is acceptable.

boxed-cap: The BASELINE to CAPLINE distance of the text string exactly fills the text restriction area in the vertical direction and the width of the string exactly fits the area in the horizontal direction.

boxed-all: As ‘boxed-cap', but the BOTTOMLINE to TOPLINE distance is used for the vertical measurement.

isotropic-cap: The text string is displayed as large as possible within the text restriction area without altering the ratio of the height to the width of the string. The text string will exactly fill the text restriction area in either the horizontal or vertical direction and the characters will have the same proportions as if no adjustments had been made. The BASELINE to CAPLINE distance of the text is the measurement which is matched to the vertical dimension of the area.

isotropic-all: As ‘isotropic-cap' but BOTTOMLINE to TOPLINE is used for the vertical measurement.

justified: The text string exactly fits the text restriction area in the escapement direction without changing the proportions of the characters. That is, the height of the characters and their aspect ratio (expansion factor) are not altered. Any extra space that needs to be added to accomplish justification may be added between characters, between words, or both.

These are illustrated in figure 20.

NOTE 4 The RESTRICTED TEXT TYPE element, which defines the way in which the text string is to fit the box, is only defined and permitted in Version 3 and Version 4 metafiles. Any of these styles is valid in Version 1 and Version 2 metafiles, however there is no element to select amongst them.

NOTE 5 For certain combinations of generalized text path mode and character orientation the text string will not be confined exactly to the restricted text area. These combinations are: when generalized text path mode is ‘axis-tangential' and the character base vector of the CHARACTER ORIENTATION element is not in the same direction as the default direction. In these circumstances the character cell may be skewed with respect to the escapement direction and so parts (e.g., corners) of the character may appear outside the text restriction area.

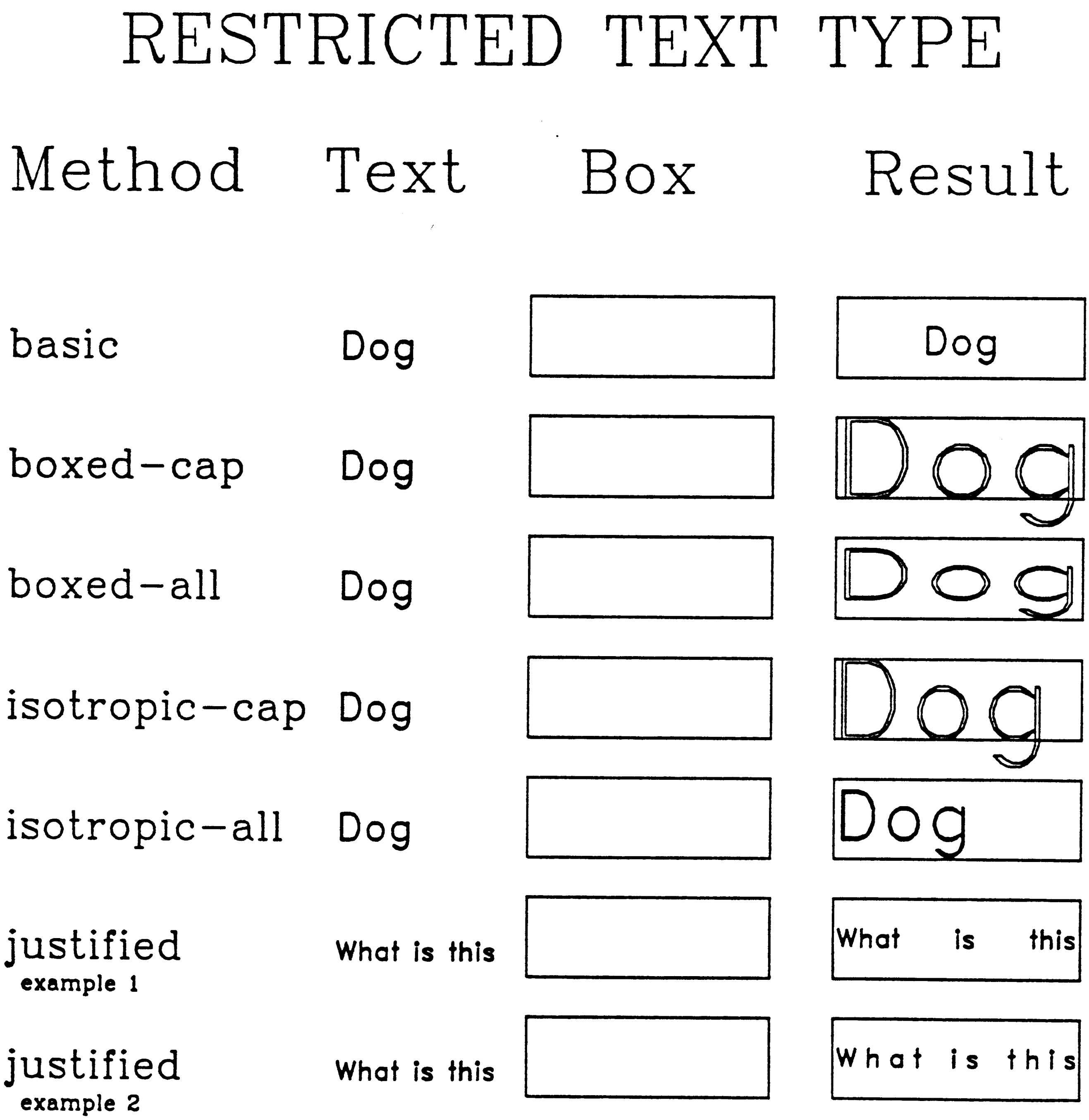


Figure 20 — Examples of RESTRICTED TEXT TYPE

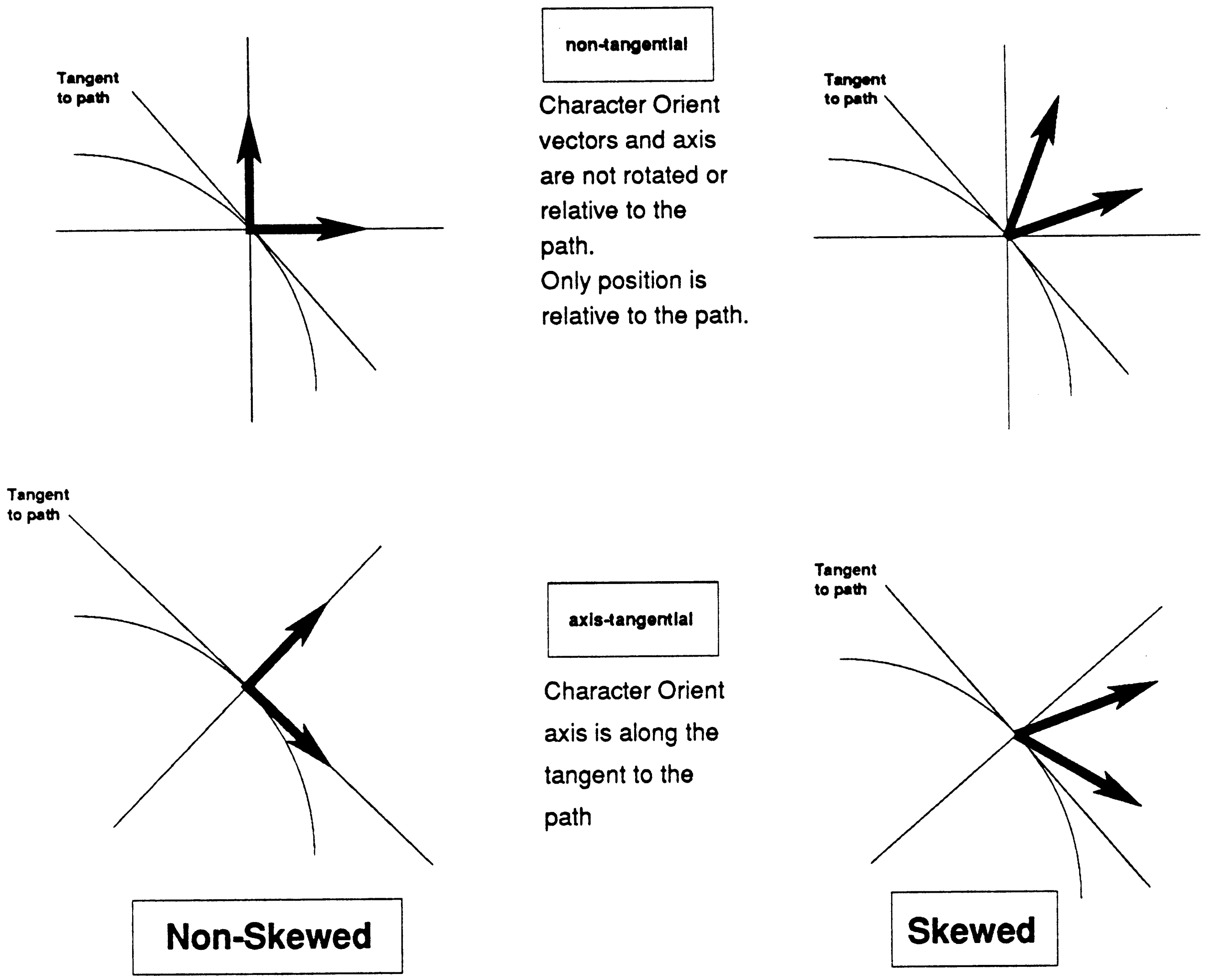


Figure 21 — Illustration of the modes of GENERALIZED TEXT PATH MODE

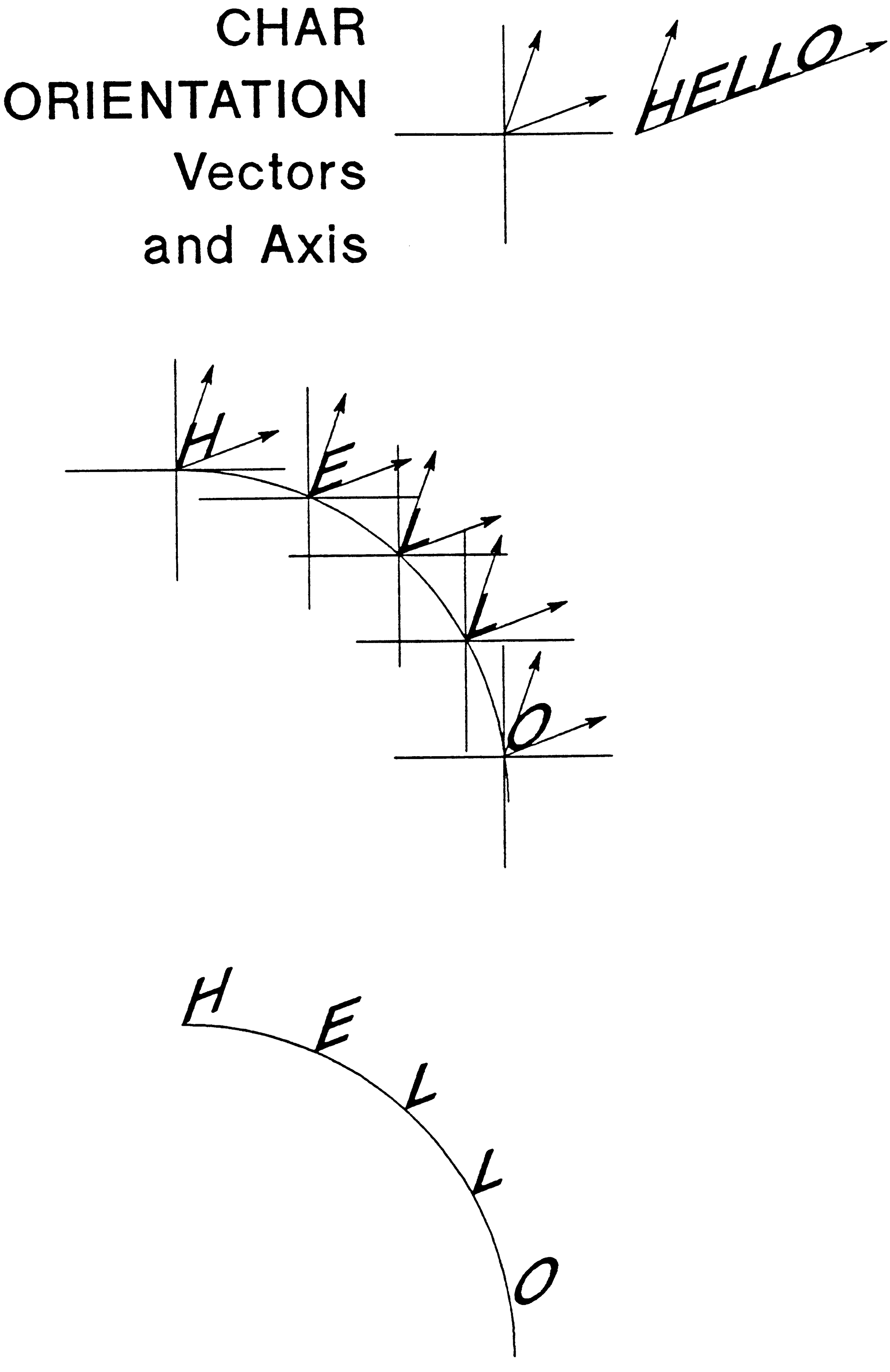


Figure 22 — Examples of GENERALIZED TEXT PATH MODE, non-tangential

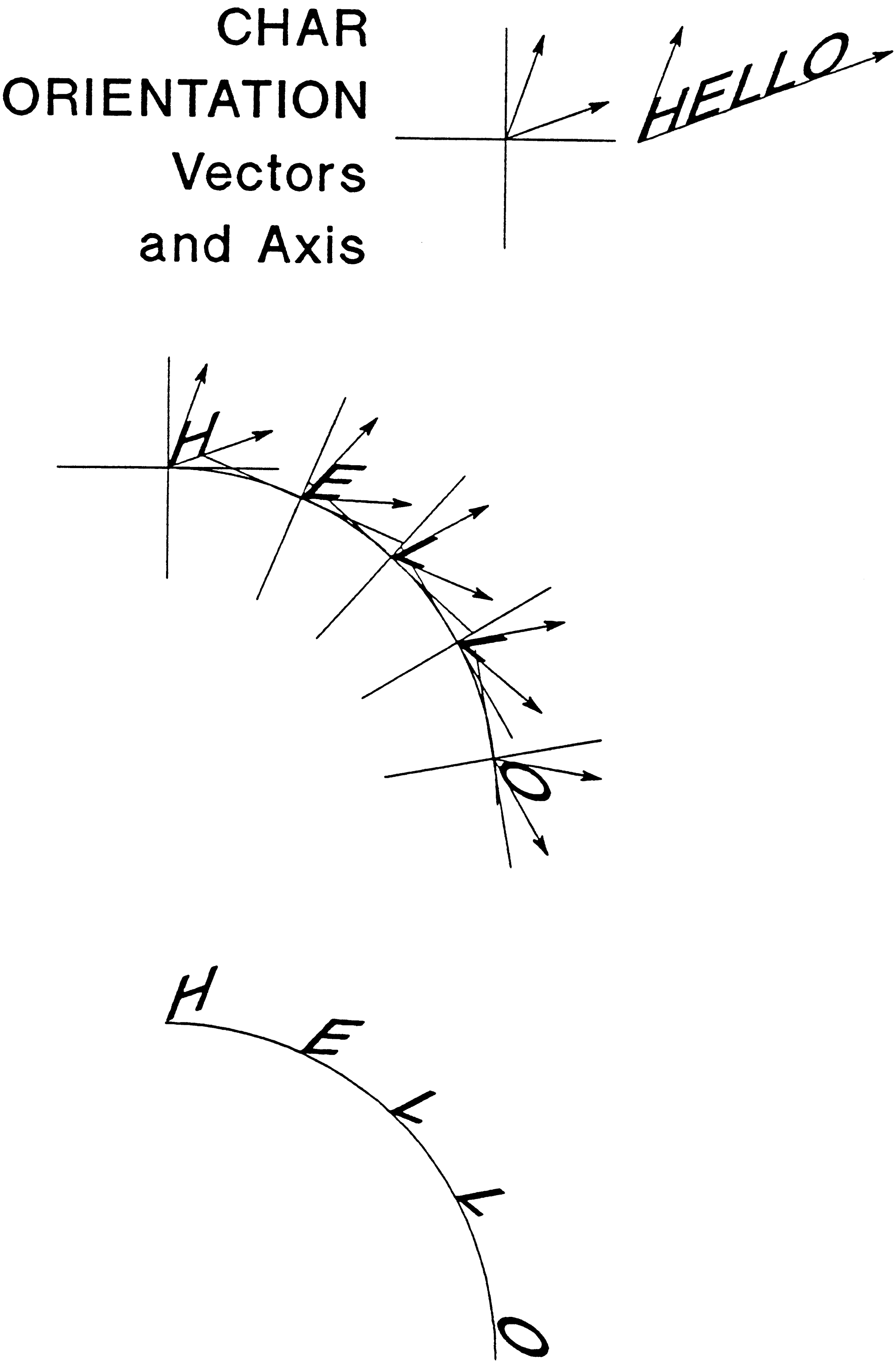


Figure 23 — Examples of GENERALIZED TEXT PATH MODE, axis-tangential

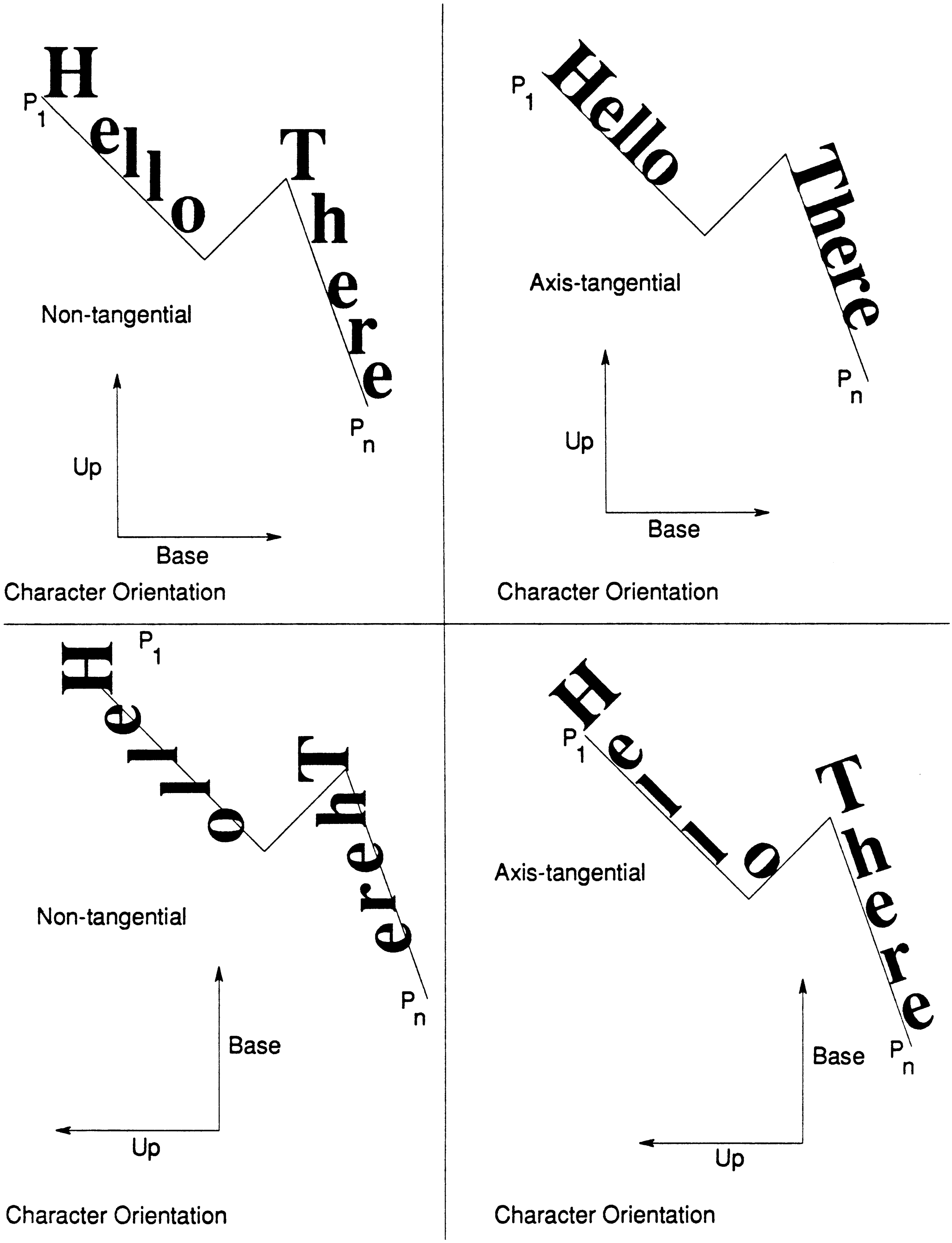


Figure 24 — GENERALIZED TEXT PATH MODE and unsmooth paths

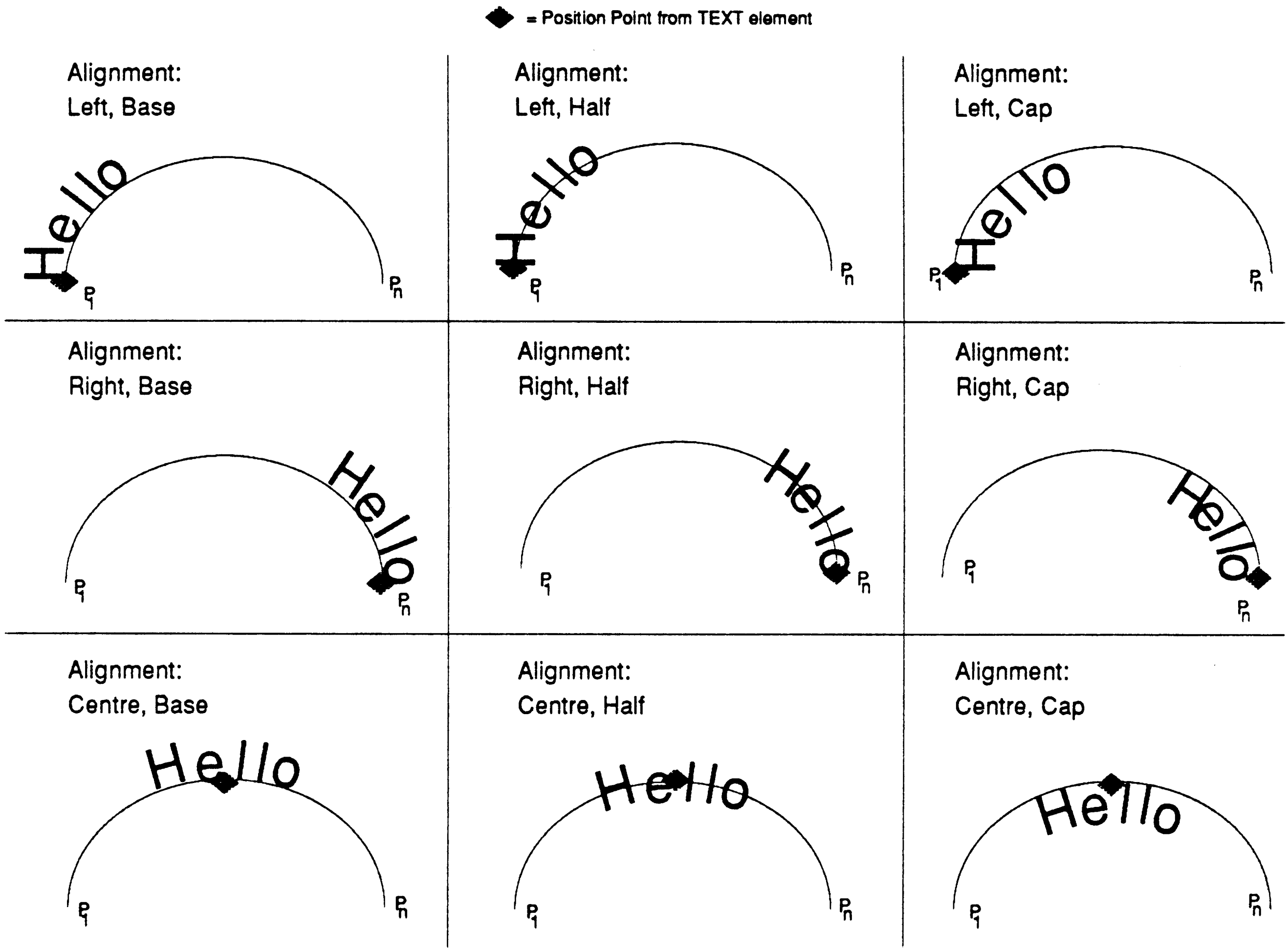


Figure 25 — Effect of TEXT ALIGNMENT and path text

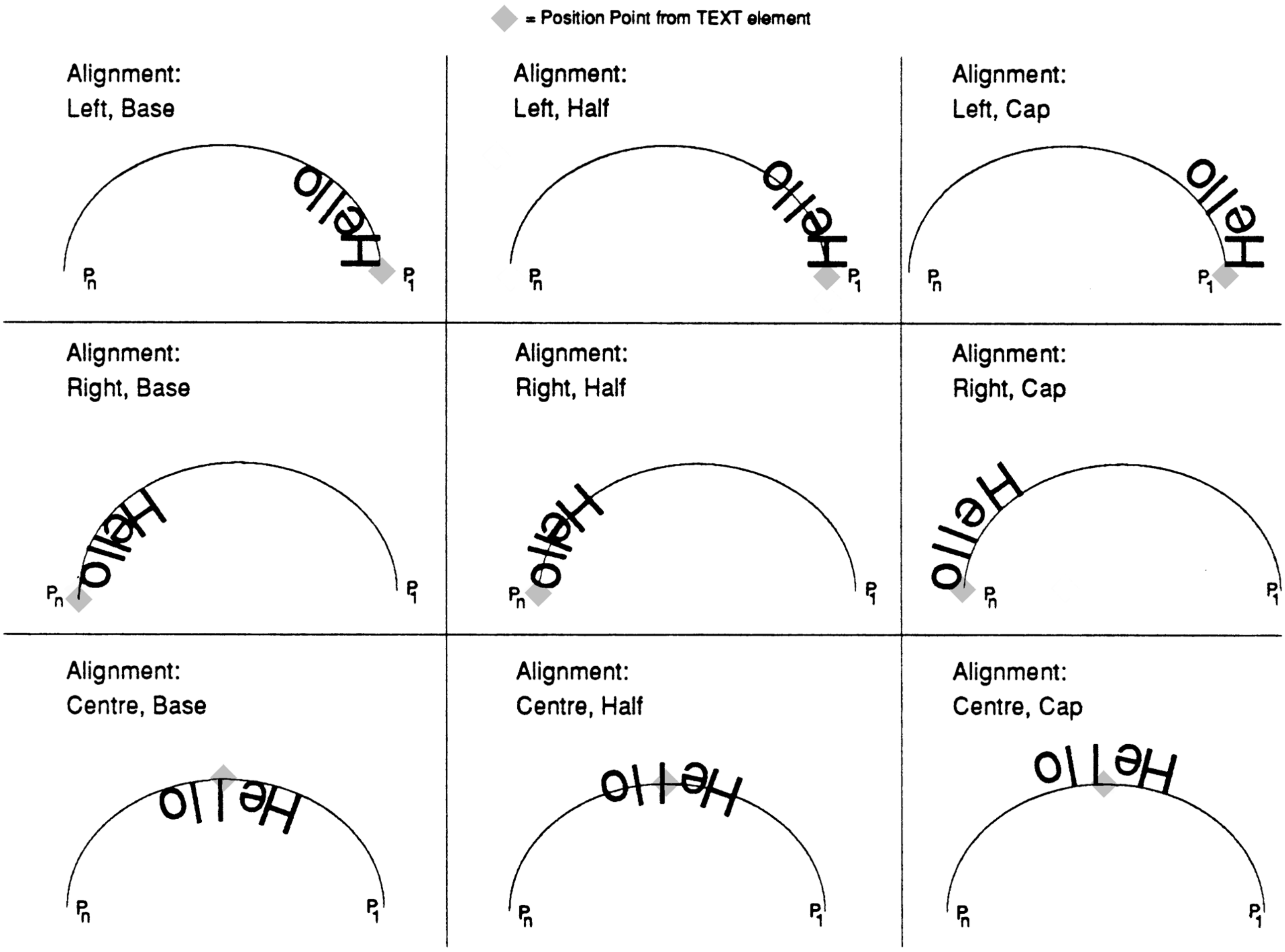


Figure 26 — Effect of path direction and TEXT ALIGNMENT with path text

6.7.4 Filled-area attributes

Separate control is provided over the appearances of the interior and the edge of filled-area primitives. There are two bundles associated with filled-area elements.

6.7.4.1 Fill bundle

The FILL bundle is associated with the interior attributes of filled-area elements. The FILL BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) INTERIOR STYLE: determines which of the classes of interior (‘hollow', ‘solid', ‘pattern', ‘hatch', ‘empty', ‘geometric pattern', or ‘interpolated') is used to draw a filled-area element;

b) FILL COLOUR: determines the colour in which the interior of a filled-area primitive is drawn. This applies only to interior styles ‘hollow', ‘solid' and ‘hatch' (the drawn boundary of a ‘hollow' area is considered as part of the representation of the interior);

c) HATCH INDEX: determines which hatch style is used if ‘hatch' interior style is selected;

d) PATTERN INDEX: determines which entry in the pattern table is used if ‘pattern' interior style is selected or the geometric pattern to be used if ‘geometric pattern’ interior style is selected.

6.7.4.2 Edge bundle

The EDGE bundle is associated with the edge attributes of filled-area elements. The EDGE BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

a) EDGE TYPE: determines the line type with which the edges are drawn;

b) EDGE WIDTH: determines the width of the edge;

c) EDGE COLOUR: determines the colour in which the edge is drawn.

6.7.4.3 Individual fill attributes and usage of fill attributes

6.7.4.3.1 Interior styles

The INTERIOR STYLE attribute selects one of the styles in which the interiors of filled-area elements are rendered:

hollow: No filling, but the boundary (bounding line) of the filled area is drawn using the fill colour currently selected (either via FILL BUNDLE INDEX or FILL COLOUR depending on the corresponding FILL COLOUR ASF). The boundary of a ‘hollow' filled area is considered to be the representation of the interior. The boundary is distinct from the edge, and is drawn only for ‘hollow' filled areas. The linetype and linewidth of the boundary are implementation dependent.

solid: Fill the interior using the fill colour currently selected (either via FILL BUNDLE INDEX or FILL COLOUR depending on the corresponding FILL COLOUR ASF).

pattern: Fill the interior using the pattern index currently selected (either via FILL BUNDLE INDEX or PATTERN INDEX, depending on the corresponding PATTERN INDEX ASF) as an index into the pattern table or the geometric pattern table.

hatch: Fill the interior using the fill colour and the hatch index currently selected (either via FILL BUNDLE INDEX or individual attributes FILL COLOUR and HATCH INDEX, depending on the corresponding ASFs). Hatch styles may be defined in the CGM. An arbitrarily complex arrangement of hatch lines and inter-line spaces may be defined, and direction vectors for either single hatch or cross hatch may be specified. The colour and line type of the lines in the hatch may be defined as a part of the hatch style definition using the HATCH STYLE DEFINITION element. Hatch styles are associated with an index when defined, and invoked as a filling interior with the normal HATCH INDEX element when the interior style is ‘hatch'.

empty: No filling is done and no boundary is drawn, i.e., nothing is done to represent the interior. The only potentially visible component of an ‘empty' filled area is the edge, subject to EDGE VISIBILITY and the other edge attributes.

interpolated: Fill the interior using the interpolated colour gradient defined by the INTERPOLATED INTERIOR element.

geometric pattern: Fill the interior using the geometric pattern associated with the pattern index currently selected (either via PATTERN INDEX or FILL BUNDLE INDEX, depending on the corresponding PATTERN INDEX ASF). The GEOMETRIC PATTERN DEFINITION element associates the index with a segment which is used to fill the interior.

6.7.4.3.2 Interpolated interiors

The metafile provides a general purpose element for defining several different styles of interpolated interiors. A solid but continuously graded colour interior is defined for filled-area primitives. Conceptually a graded-colour plane of infinite extent is defined and this is "extruded" through the interior of filled area primitives to define the appearance of the interior.

A number of styles are defined. For each style, the parameterization of the element defines a "reference geometry" two parallel lines, an ellipse, or a triangle and then defines colour interpolation points whose positions are defined relative to the reference geometry. Colours in the parameter list of the element are assigned to these latter points. For some of the styles (parallel and elliptical), multiple parallel or concentric bands (stages) of independently interpolated colour may be defined.

The following styles are defined:

parallel: This is a multi-stage style. Each stage is a infinite band bounded by two parallel lines. A reference colour is defined on each parallel line. Colour is constant along any parallel line within the band and is equal to the linear interpolant of the reference colours. Outside of the outer-most defined bands (in the two semi-infinite half planes) the colour is constant and equal to the corresponding outer-most reference colours.

elliptical: This is a multi-stage style. Each stage is an elliptical annulus with a reference colour defined on each of the inner and outer bounding ellipses. Colours are constant along ellipses which are concentric to the reference ellipse. The colour at any ellipse within a band is constant and equal to the linear interpolant of the reference colours on the two bounding ellipses. Outside the outer-most defined ellipse the colour is constant and equal to the corresponding outer-most reference colour. Inside the inner-most defined ellipse the colour is constant and equal to the corresponding inner-most reference colour.

triangular: Colours are associated with three points defining a triangle. The unique bi-linear interpolated colour is defined at each point in the interior of the triangle. Outside of the triangle the colour is defined to be constant on rays from the centre of the triangle to infinity and equal to the interpolated colour value on the boundary of the triangle.

The parameterization of the element defining interpolated interiors is consistent with application of the rules of INTERIOR STYLE SPECIFICATION MODE regarding transformation of interiors. The reference geometry and style defined by the element transform or do not transform according to the value of the mode.

Colour interpolation is performed in the colour space specified by the COLOUR MODEL element.

6.7.4.3.3 Geometric patterns

The metafile provides for the definition of geometric patterns which are constructed from the set of all picture elements which are available within segments. Geometric pattern definition associates either a global segment, or a local segment defined in the Picture Descriptor, with a pattern index. The segment is clipped to the rectangle defined by a pattern extent. This rectangle is used to fill the pattern box as described further under the PATTERN SIZE element in clause 5, and the geometric pattern fills the interiors of filled areas as described under that element.

6.7.4.4 Individual edge attributes

In addition to the edge attributes which may be bundled there are a number of individual edge attributes.

a) EDGE CAP specifies the appearance of the endpoints of edges, such as might occur due to the turning off and on of edge visibility in POLYGON SET elements or Closed Figure elements. The supported styles and their definitions are as for the LINE CAP element, with the endpoints of visible edge segments corresponding to the "open endpoints" of the line elements definition.

b) EDGE JOIN specifies the appearance of edges at the vertices of filled-area elements or at junctions between distinct elements in compound filled area elements. The supported styles and their definitions are as for the

LINE JOIN element. For the style ‘mitre', the rendering of the edge join is affected by the MITRE LIMIT Control Element.

Both edge caps and edge joins behave as does edge width with respect to transformation. The behaviour is determined by the value of EDGE WIDTH SPECIFICATION MODE, and is as described for LINE CAP and LINE JOIN.

c) EDGE TYPE CONTINUATION provides control of the behaviour of non-solid edge types at interior vertices and junctions of the edges of filled-area elements. The supported styles and their definitions are as for the LINE TYPE CONTINUATION element.

d) EDGE TYPE INITIAL OFFSET allows control of how much of the first cycle of a non-solid line type to omit before drawing commences for a line primitive. It is specified as a fraction of one full cycle.

The Picture Descriptor element LINE AND EDGE TYPE DEFINITION allows the precise definition of the solid/gap sequences which comprise a line or edge type. A definition is associated with an index by this element, and this index may be refered to within the picture by LINE TYPE and EDGE TYPE elements.

6.7.5 Specification modes and transformation of aspects

The CGM provides a mechanism for selecting different modes by which geometric information related to line width, line type, edge width, edge type, marker size, and fill interiors is specified. The following specification modes are defined for aspects relating to size and distance:

absolute: Specification units are VDC;

scaled: Specification units are a scale factor to be applied by the interpreter to a device-dependent "nominal" measure;

fractional: Specification units are interpreted as a fraction of the horizontal dimension of the default device viewport;

mm: Specification units are millimetres.

All of these modes are permitted in Version 3 and Version 4 metafiles. Only ‘absolute' and ‘scaled' modes are permitted in Version 1 and 2 metafiles.

Some primitives (those in segments) may have a transformation associated with their VDC definition. The application of this transformation gives the actual appearance of the primitive in VDC. Also all primitives in a picture may be subject to a transformation of VDC to the display device, which defines their final displayed size and appearance. This transformation may be explicitly specified by the DEVICE VIEWPORT element or partially specified by the SCALING MODE element. Mode ‘absolute' means that the affected aspects (e.g., line width, line caps and joins, line dash and gap lengths, etc) are subject to any and all transformations. In the other three modes none of these aspects are subject to any transformations.

In mode ‘absolute', the interpreter conceptually renders the associated aspects in VDC space before it applies any associated transformations. For the other three modes, the aspect is conceptually rendered after all transformations have been applied to the geometry of the primitive, in the drawing space of the device.

The three non-transformable modes are distinguished by these properties:

— ‘scaled' gives a result which is completely device and interpreter dependent neither the final displayed sizes nor their relationship to the rest of the displayed picture are precisely controllable;

— ‘fractional' gives a precisely controllable way of specifying the sizes, according to their relationship to the rest of the picture but not in terms of actual physical measurements; it is device independent in giving a picture whose components maintain a fixed relationship to each other at all display sizes, but it does not provide for invariant actual sizes across a range of picture sizes.

— ‘mm' gives a precisely controllable way of specifying the sizes in terms of their actual physical measurements which remain invariant as display size varies; it is device dependent in that the request may not make sense at some display sizes on some devices.

6.7.6 Colour attributes

In CGM, colours are described by a colour model together with a specification of colour coordinates in the colour space of that model. Colour models define a colour coordinate system and a subspace, within which each describable colour is represented by a point. The CGM provides the following colour models: RGB (the default), CIELAB, CIELUV, CMYK, and RGB-related. The selection of one of these models is made in the Metafile Descriptor.

RGB is the only colour model available in Version 1 and Version 2 metafiles.

The RGB system is an additive colour mixture system, i.e., the red (R), green (G) and blue (B) stimuli additively combine their radiant intensity together to form the complete range of colours. In case RGB data are non-linear in the radiant intensity, a look-up table may be used in the colour calibration (see below) to transform into (linear) RGB tristimulus values.

The RGB system is used by a number of different types of devices, such as colour monitors, film writers, and colour input scanners.

Two CIE recommended uniform colour spaces, CIELAB and CIELUV, are allowed in the CGM. These colour spaces are non-linear transformations of the CIE 1931 XYZ tristimulus space, providing approximate correlates of hue, lightness and chroma. CIELAB and CIELUV closely approximate a uniform colour space over small distances, and provide an approximately uniform measure of perceived colour differences. CIELUV is commonly used for applications involving self-luminous displays where the additivity provided by its associated chromaticity diagram is important. CIELAB is more commonly used in surface colour applications and for the paints, plastics and textile industries.

The CMYK colour model is based on the subtractive colour mixture of cyan (C), magenta (M) and yellow (Y) primaries with the inclusion of black (K). This model is used primarily in the printing industry. In the CGM, the quantities C, M, Y, and K represent the relative area occupied by a colorant at a particular point in order to produce a final image. In theory, three colorants cyan (C), magenta (M), and yellow (Y) should be sufficient to reproduce all desired colours. In practice, black colorant is added to increase the colour gamut (lower L^* values), i.e. higher density blacks are possible than with usual cyan, magenta and yellow colorants.

RGB-related colour spaces are colour spaces derived through linear transformations (3x3 matrix) from the RGB colour space. These colour spaces usually relate to non-linear RGB values. Examples are luminance/chrominance signals in television and video devices, such as YUV (PAL, SECAM), YIA (NTSC), YC_RC_B (CCITT video codecs and ITU 601 studio standard).

NOTE 1 The fact that ISO/IEC 8632 allows 3- and 4-dimensional colour spaces results in a 3-tuple or 4-tuple data type for direct colour specification, depending on the COLOUR MODEL element.

The meaning of each colour space allowed in the CGM (RGB, CIELAB, CIELUV CMYK, and RGB-related) is defined by the transformation necessary to convert a colour specification expressed in a reference colour space to or from the specification of the same colour in the CGM colour space (colour calibration). The reference colour space is the CIE 1931 XYZ space, which is defined in CIE Publication 15.2. Conceptually, colour values specified by one of the CGM colour spaces are interpreted by converting them into the reference colour space, and then converting them from the reference colour space to the device space of the interpreter.

NOTE 2 CIE Publication 15.2 defines the CIE 1931 XYZ space in terms of the CIE 1931 2° Standard Observer (CIE S002). The space is colorimetrically precise and covers all perceivable colours. It is based on properties of the human visual system, determined by extensive experiments in colour matching, rather than on the properties of any particular device. In 1964, the CIE defined a new 10° standard observer for matching colour fields from 4° to 10° in angular subtense. As smaller colour fields are expected in computer graphics, the 2° 1931 Standard Observer was selected.

NOTE 3 It is recognized that the general problem of appearance matching has not been completely solved. However, this standard uses the best available and internationally recognized approach, which is the CIE system of colorimetry.

NOTE 4 The specification of a colour in the reference colour system can be made by reporting its XYZ tristimulus values and/or its x, y, z chromaticity coordinates, which are given by x = X/T, y = Y/T, where T = X+Y+Z. Plotting the y chromaticity coordinate versus the x chromaticity coordinate for spectral colours results in a horseshoe shaped curve known as the spectral locus. If, for example, the colours available from an RGB system are transformed into chromaticity coordinates, they will fall within a triangle whose vertices are defined by the RGB primaries of that system.

The reference colour space is normalized such that the Y tristimulus value is 1 for the reference white to allow for simplicity of conversions from colorimetric values to other colour spaces.

NOTE 5 This differs from the CIE recommendation of normalizing Y of the perfect reflecting (or transmitting) diffuser (reference white) to exactly 100.

The colour calibration, as provided by the COLOUR CALIBRATION element, depends on the colour space specification. The calibration data for all colour spaces contain a reference white value to allow for how colours are perceived in relation to the viewing environment. This is the set of CIEXYZ values of the reference white (X_n, Y_n, Z_n).

The reference white value is the only calibration data applicable to the CIELAB and CIELUV colour spaces.

Additionally, the calibration data for the RGB colour space consists of a 3x3 matrix, specifying the CIEXYZ values of each of the RGB primaries, as well as look-up tables for non-linear RGB values.

The calibration data for the CMYK colour space consist of CMYK grid locations and corresponding CIEXYZ values. No calibration data are available for standard ink sets.

For RGB-related colour spaces, the calibration data include a second 3x3 matrix in addition to the calibration data for the RGB colour space.

For details of conversion between each CGM colour space and the CIE reference colour space, see annex G, which has been extracted from annexes I, J, and K of Addendum 2 to CCITT Recommendation T.412/ISO 8613-2.

The CGM provides two mechanisms for colour selection: 'direct' and 'indexed'. In 'direct' colour selection, the colour is defined by providing values for the normalized weights of the colour components for the selected colour model. In 'indexed' colour selection, the colour is defined by an index into a table of direct colour values. Selection of one of these mechanisms is specified by the COLOUR SELECTION MODE element.

For 'indexed' colour selection the COLOUR TABLE attribute element is provided to set the contents of the colour table. COLOUR TABLE may appear in the picture body for metafiles of Version 1, 2, 3 and 4, as well as in the Picture Descriptor for Version 2, 3 and 4 metafiles. Conceptually, for each colour selection mode there is an associated current colour in that mode. This is set by, either the direct colour specified, for 'direct', or that defined by the index into the colour table, for 'indexed'. When the colour selection mode is changed the current colour takes the current colour value for the new colour selection mode.

Redefinition of a colour index using the COLOUR TABLE element shall have no effect on any graphical primitive elements which have already been displayed using the given index (or bound to it).

For direct colour specification in RGB and CMYK colour spaces, normalized weights for the colour components are specified. For example, in the default situation, these are the red, green, and blue components of the desired colour. In the abstract, each component of the 3-tuple or 4-tuple is normalized to the continuous range of real numbers [0,1]; the normalization also has the property that any 3-tuple or 4-tuple with identical components represents equal weights of the colour components. For any given component, one end of the range indicates that none of that component is included, and the other end indicates that the maximum intensity of that component is included in the colour, with an infinite number of component values in between. For the RGB colour space, for example, (0,0,0) thus represents black, (1,1,1) represents white, and (x,x,x) with x between 0 and 1 represent shades of gray. For the CMYK colour space the opposite holds: (0,0,0,0) represents white (i.e., substrate with no ink) and (1,1,1,1) represents black (i.e., maximum ink).

For direct colour specification in CIELAB, CIELUV and RGB-related colour spaces, each component of a 3-tuple is represented through a scale and offset factor.

NOTE 6 For example, a defined colour component value labelled x is represented by $sx + o$, s and o being the scale and offset.

No range is specified as in the RGB and CMYK colour spaces. This is due to the fact that the visible colours either do not fill, or are not normalized to the unit cube as for RGB and CMYK colour spaces.

The COLOUR CALIBRATION element provides for the selection of proper combinations of its parameters, including the choice that no calibration should be applied.

There is a Metafile Descriptor element, COLOUR VALUE EXTENT, which allows metafile generators to specify the minimum and maximum metafile colour values for RGB and CMYK colour spaces; these correspond with the abstract (0,0,0) and (1,1,1) for RGB, and (0,0,0,0), (1,1,1,1) for CMYK. For CIELAB, CIELUV, and RGB-related colour spaces, this element specifies the scale and offset parameters.

6.7.7 Pick identifier

The pick identifier is associated with graphical primitive elements within segments (see 6.10). It is the only attribute element which does not affect the appearance of a graphical primitive element. It merely establishes a means of identification of primitives within segments at metafile interpretation. The PICK IDENTIFIER element has no graphical effect.

6.7.8 Compound text path

The BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements delimit a compound text path. These elements permit the definition of a path that consists of a number of distinct elements which serves as a reference path for laying out subsequent text strings (and is not drawn). If two line segments which are adjacent in the definition are not physically contiguous, i.e., the end point of one line does not coincide with the first point of the next one, then the path includes the straight line segment joining these two points.

The permissible elements in the definition of a compound text path definition are identical to those in a compound line definition.

The compound text path permits arbitrary, complex placement of text. Each glyph in a text string is placed with its reference point on the compound text path according to the alignment. When the generalized text path mode is ‘axis-tangential', the tangent at the point at which the character is to be drawn is the logical base line for the character cell. If a glyph's reference point aligns with the junction of two line elements of the compound text path, it is implementation dependent whether the final direction of the first segment or the initial direction of the second segment will be used. Positioning of subsequent glyphs is accomplished so that the arc length between the glyph position points is the same as the distance between glyph position points when they are laid out on a straight line. When the path ends before all glyphs have been placed, the path for the excess text is the straight line described by the tangent at the end of the compound text path.

NOTE Using paths with sharp curvature is likely to cause overlapping tops or bottoms of characters unless care is taken in adjusting the text attributes.

6.7.9 Symbol Attributes

There are no bundled symbol attributes — all symbol attributes are individual.

Selection of the current symbol library from the list of available libraries is specified by the SYMBOL LIBRARY INDEX element. The Metafile Descriptor element SYMBOL LIBRARY LIST associates index values with symbol library names. Access to symbol libraries and symbols is analogous to access to text fonts and glyphs. The SYMBOL LIBRARY LIST associates the names of external libraries with indexes for internal reference, just as FONT LIST associates font names with internal indexes; SYMBOL LIBRARY INDEX selects the current symbol library, just as FONT INDEX selects the current font for text display; and POLYSYMBOL selects the particular symbol, just as the character codes within text strings select glyphs.

The symbol coordinate system is illustrated in figure 27. The symbol extent box is the design size of the symbol. It is used to define the transformation to scale the symbol to the size specified in SYMBOL SIZE. The symbol need not be entirely contained within the symbol extent box. Each symbol has a reference point (though all symbols in a symbol library need not have the same reference point). The symbol's reference point is aligned with each point in the list of position points specified in the POLYSYMBOL element.

The SYMBOL SIZE specifies the VDC sizes to which the design height of the symbol (the design distance between the top and the bottom of the symbol extent box) and the design width of the symbol (the design distance between the left and right side of the symbol extent box) shall be scaled for symbol display.

SYMBOL ORIENTATION specifies a symbol up vector and a base vector, which define the orientation, skew, and distortion of the symbol.

The way in which software invoking the metafile generator and/or the metafile generator itself may use SYMBOL ORIENTATION is described in this subclause. To generate the SYMBOL ORIENTATION and SYMBOL SIZE elements, a vector whose length is the symbol height and whose direction is the desired symbol up vector is created. A second vector is also created whose length is the symbol width and whose direction is negative 90° from the up vector. This pair of vectors may be transformed before being passed to the metafile generator to generate the SYMBOL ORIENTATION and SYMBOL SIZE elements. If the resultant vectors are not orthogonal, the symbol extent box becomes a parallelogram, and the symbol is skewed. If the positive angle from the up vector to the base vector is less than 180°, the symbol is mirror imaged. The height and width parameters of the SYMBOL SIZE element are derived from the transformed length and width vectors, and the indicator as to how the interpreter is to scale the symbol is generated from the application requirements.

The SYMBOL SIZE and SYMBOL ORIENTATION are decoupled. Thus the lengths of the vectors in SYMBOL ORIENTATION are not significant; only their directions are significant. The SYMBOL SIZE allows the generator to request that the symbol height is to be scaled without distortion of the symbol aspect ratio, or the symbol width is to be scaled without distortion of the aspect ratio, or both are to be scaled with possible distortion of the aspect ratio.

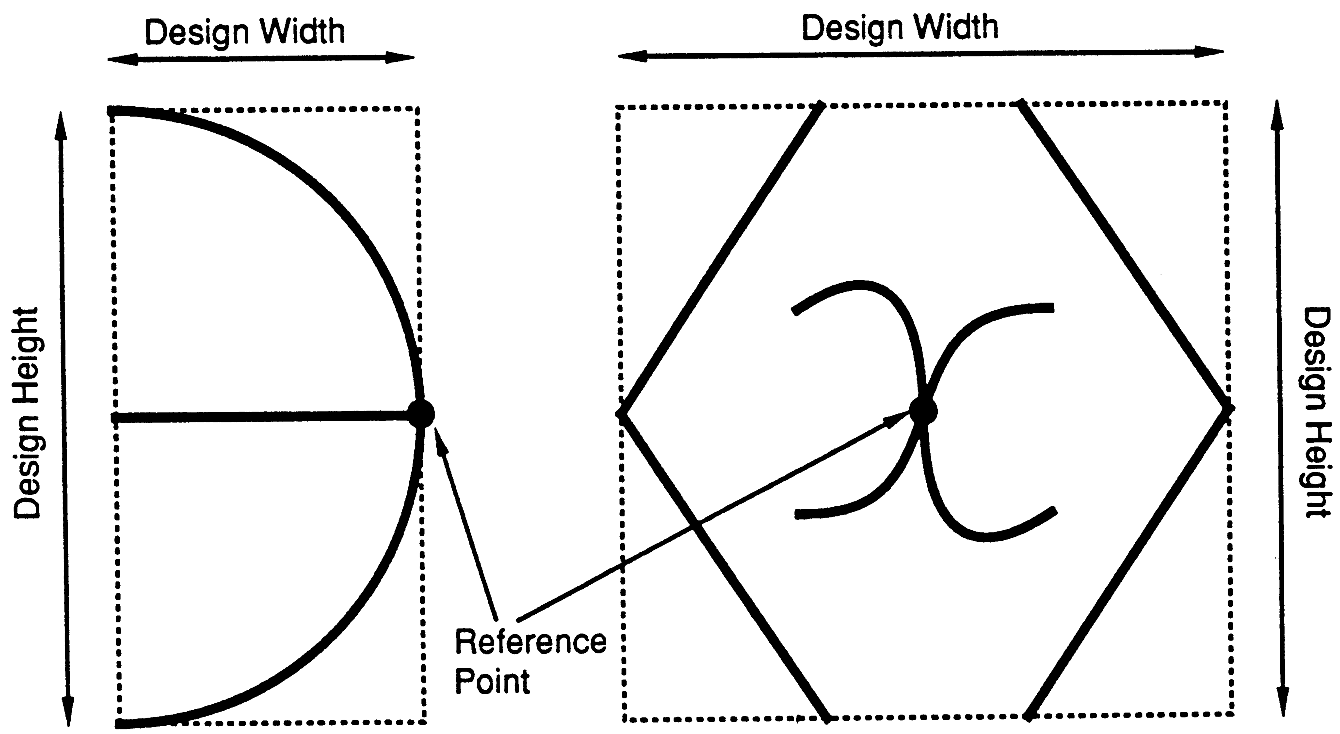


Figure 27 — Symbol coordinate system

6.8 Escape elements

ESCAPE elements describe device- or system-dependent data in the CGM. ESCAPEs may be included in the metafile at the discretion of the user, but direct effects and side effects of the use of non-standardized elements are beyond the scope of ISO/IEC 8632. ISO/IEC 8632 imposes no constraints on the functional intent or content of data passed by the ESCAPE mechanism.

6.9 External elements

External elements communicate information not directly related to the generation of a graphical image. They may appear anywhere in the CGM.

The MESSAGE element specifies a string of characters used to communicate information to operators at CGM interpretation time. This element is intended to be used to provide special device-dependent information necessary to process a CGM. Control over the position and appearance of the character string is not provided.

The APPLICATION DATA element allows applications to store and access private data. This element is not a graphical element and its interpretation will have no effect on the picture produced by an interpreter.

For specification of non-standardized graphical effects, the ESCAPE and GENERALIZED DRAWING PRIMITIVE elements are provided. These elements may have an effect on the picture produced by an interpreter.

6.10 Segment elements

6.10.1 Introduction

In the CGM Versions 2 and 3, graphic objects may be grouped in segments. Each segment is identified by a unique segment identifier. Segments may have the attributes:

a) transformation;

b) highlighting;

c) display and pick priority.

These may be defined at segment definition time, before the first graphical primitives of the segment. Once defined, they shall not be changed.

Only elements inside segments are affected by the segment attributes.

The segment elements are divided into two groups, the segment control elements and the segment attribute elements.

The segment control elements are:

COPY SEGMENT
INHERITANCE FILTER
CLIP INHERITANCE

The segment attribute elements are:

SEGMENT TRANSFORMATION
SEGMENT HIGHLIGHTING
SEGMENT DISPLAY PRIORITY
SEGMENT PICK PRIORITY

Segments are delimited by BEGIN SEGMENT and END SEGMENT. Segment attribute elements, if used, shall appear immediately after BEGIN SEGMENT, and before the first elements of any other type. The *segment identifer* parameter of segment attribute elements shall refer to the segment in which the elements are contained.

6.10.2 Local and global segments

There are two types of segments: local segments and global segments. Both contain primitives and attributes that can be manipulated in the manner described above. Local segments have no existence beyond the bounds of the picture in which they are defined they may be defined either in the Picture Descriptor or in the picture body.

Defining a local segment in a picture body automatically includes that segment in the picture's image. Local segments defined within the Picture Descriptor are not automatically included in the picture's image but may be referenced from within that picture. Global segments can be referenced by any of the pictures in the metafile in which they are defined.

6.10.2.1 Location of, and access to, global segments

A global segment is delimited by the BEGIN SEGMENT and END SEGMENT elements. Global segments are defined in the Metafile Descriptor. They are not a part of any picture within the metafile. They shall be accessed from within individual pictures by the COPY SEGMENT (see 6.10.5) element. The COPY SEGMENT element incorporates the segment into the open picture in the same way for both local and global segments.

6.10.2.2 Segment related elements permitted in the metafile or picture descriptors

BEGIN SEGMENT is the only segment-related element that is allowed within the Metafile Descriptor State (MDS) or the Picture Descriptor State (PDS). BEGIN SEGMENT changes the state to Global Segment State (GSS) or Picture Descriptor Segment State (DSS) respectively.

NOTE In Version 2 metafiles segments are not allowed to be defined in Picture Descriptor State.

6.10.2.3 References to global segments

Within pictures, no elements are allowed that would modify the contents or default appearance of global segments. This restriction preserves the logical independence of pictures and the ability to randomly access pictures. The only references to global segments within pictures shall be by using the COPY SEGMENT element.

6.10.2.4 Association of control and attribute elements with primitives inside segments

The current modal values of control and attribute elements are associated with the primitives inside local segments which occur within the picture body (i.e., after the BEGIN PICTURE BODY element). The modal values established by setting control or attribute elements within a segment remain in effect outside the segment until they are explicitly changed.

Control and attribute elements are bound in global segments (which are only defined in the Metafile Descriptor), and local segments which are defined in the Picture Descriptor, as they are in local segments which are defined in the picture body. Upon the occurrence of BEGIN METAFILE, every element that is modally defined and bound to primitives (Metafile Descriptor elements defining modes and precisions, Picture Descriptor elements, Control elements, Attribute elements and Segment Control elements) has a default value. Conceptually the set of all of these define a "modal state list".

The Metafile Descriptor (MD) and Picture Descriptor (PD) are processed sequentially. Throughout the MD, modal MD elements modify the MD entries in the state list and occurrences (possibly multiple) of the Metafile Defaults Replacement element allow manipulation (outside of GSS state) of the rest of the modal elements (as well as explicitly changing the defaults). Throughout the PD, modal PD elements modify the PD entries in the state list.

Within GSS, PDS, and LSS states the allowable modal (control, attribute, and segment attribute) elements also alter the contents of the modal state list. The values of modal elements that are in effect upon BEGIN PICTURE are the default values for that picture, whether they are implicit (defined in clause 8 of this part of ISO/IEC 8632) or explicit (that is, by values set in the Metafile Defaults Replacement).

6.10.3 Delimiting and naming segments

The contents of a segment are delimited by the elements BEGIN SEGMENT and END SEGMENT which are delimiter elements. The elements in between these two delimiters are a part of that segment. Each segment has an identifier associated with it. No two global segments shall have the same identifier and no local segment shall have an identifier which is the same as either a local segment in the same picture or the same as a global segment.

6.10.4 Segment attributes

6.10.4.1 Introduction

The segment attributes associated with each segment control its display. Segment attributes shall be set only after the segment has been opened with the BEGIN SEGMENT element. When a segment is opened, the segment's attributes are set to their default values. Segment attributes, if set, shall be set immediately after the BEGIN SEGMENT element and before any other type of element. This structure is shown below:

BEGIN SEGMENT (Segment identifier)
    Segment attributes
    Allowed primitives, attributes and control elements in any order
END SEGMENT

6.10.4.2 Segment transformation

The segment transformation is a coordinate transformation associated with each segment and applies to all graphical objects in the identified segment and will be used on interpretation. Clipping rectangles are not transformed by the segment transformation. It allows scaling, translation, and rotation of segments to be defined during segment definition.

The segment transformation is a transformation of VDC space to VDC space and is distinct from the VDC-to-Device mapping which is a transformation of VDC space to device coordinate space.

The transformation attribute of a segment may be defined by the SEGMENT TRANSFORMATION element during the segment definition. A segment transformation is represented by a 2 x 3 matrix, comprising a 2 x 2 scaling and rotation portion, and a 2 x 1 translation portion. If the SEGMENT TRANSFORMATION element is not stored in the metafile, then all coordinate data is mapped using only the VDC-to-Device mapping. If the SEGMENT TRANSFORMATION is stored in the metafile, it is applied before the application of the VDC-to-Device mapping.

NOTE The use of segment transformations may produce coordinates that cannot be expressed within the VDC range. This is handled in an interpretation dependent way.

6.10.4.3 Segment highlighting

Segment highlighting can take one of two values, NORMAL or HIGHLIGHTED. The setting of this attribute selects one of these two states for the segment.

6.10.4.4 Segment display priority

The display priority attribute of a segment determines how overlapping segments are displayed. During interpretation segments with higher display priorities will be displayed as if they were in front of segments with lower display priorities. In the abstract, the display priority is a real number in the range zero to one. In the metafile, the display priority is an integer in the range defined by the Metafile Descriptor element SEGMENT PRIORITY EXTENT. The integer display priority may be normalized onto the continuous range of real numbers, zero to one, by mapping the minimum extent and maximum extent values provided in the SEGMENT PRIORITY EXTENT element onto zero and one respectively.

6.10.4.5 Segment pick priority

The pick priority attribute of a segment is used to resolve the picking of segments which overlap. In the abstract, the pick priority is a real number in the range zero to one. In the metafile, the pick priority is an integer in the range defined by the Metafile Descriptor element SEGMENT PRIORITY EXTENT. The integer pick priority may be normalized onto the continuous range of real numbers, zero to one, by mapping the minimum extent and maximum extent values provided in the SEGMENT PRIORITY EXTENT element onto zero and one respectively. Interpretation of SEGMENT PICK PRIORITY has no graphical effect.

6.10.5 Copy segment and inheritance

The COPY SEGMENT element inserts the elements of the referenced segment into the picture at the point of occurrence of the element. The elements copied may be altered in a variety of ways:

a) The inheritance filter mechanism controls whether individual attribute values are reapplied to the elements.

b) The clip inheritance mechanism controls whether the primitives in the segment are clipped to the current clip rectangle or to a combination of the current and the segment clipping rectangles.

c) The primitive elements are transformed by the copy transformation and optionally by the segment transformation of the copied segment according to the rules for transformation.

COPY SEGMENT has a transformation matrix as a parameter. The copy transformation is applied to graphical objects before they are copied. This also applies to clipping rectangles in the segment (see below). Graphical objects may be transformed to alter their location, size, and orientation.

A segment may be referenced by the COPY SEGMENT element, either within a picture or in a global segment. The attributes associated on interpretation can be those bound to the segment being copied or can be imposed by the inclusion of the INHERITANCE FILTER element.

The clipping associated with a segment can be that associated with the picture at the time of the copy or can be a combination of the current clipping and the segment clipping when the CLIP INHERITANCE element is used.

The inheritance filter mechanism allows the use of the current values of attributes and controls to be associated with the copied segment in place of the attributes and controls bound to the primitives when the segment was created. The attributes and controls to be associated with the segment can be all attributes or can be a subset of attributes. The attributes and controls are selected using the INHERITANCE FILTER element. The attributes and controls can be selected using individual or group names for attributes, controls and ASFs. The elements that can be selected are shown in table 6 for attributes and controls (both individual element names and group names) and in table 7 for ASFs.

If an attribute or group of attributes designated in the filter selection list is set to ‘state list', graphic objects inherit that attribute or group of attributes from the current modal values when a segment is copied.

If an attribute or group of attributes designated in the filter selection list is set to ‘segment', that attribute or group of attributes is unaffected (in all graphic objects employing them) by the corresponding current state list when a segment is copied.

The default inheritance filter setting value is ‘segment' for all attributes and controls.

Table 6 — Inheritance filter selection names for attributes

Attribute Group Name	**Individual Attribute Name**
LINE ATTRIBUTES	LINE BUNDLE INDEX LINE TYPE LINE WIDTH LINE COLOUR LINE CLIPPING MODE LINE CAP LINE JOIN LINE TYPE CONTINUATION LINE TYPE INITIAL OFFSET
MARKER ATTRIBUTES	MARKER BUNDLE INDEX MARKER TYPE MARKER SIZE MARKER COLOUR MARKER CLIPPING MODE
TEXT PRESENTATION AND PLACEMENT ATTRIBUTES	TEXT BUNDLE INDEX TEXT FONT INDEX TEXT PRECISION CHARACTER EXPANSION FACTOR CHARACTER SPACING TEXT COLOUR CHARACTER HEIGHT TEXT SCORE TYPE RESTRICTED TEXT TYPE
TEXT PLACEMENT AND ORIENTATION ATTRIBUTES	CHARACTER ORIENTATION TEXT PATH TEXT ALIGNMENT
FILL ATTRIBUTES	FILL BUNDLE INDEX INTERIOR STYLE FILL COLOUR HATCH INDEX PATTERN INDEX INTERPOLATED INTERIOR
EDGE ATTRIBUTES	EDGE BUNDLE INDEX EDGE TYPE EDGE WIDTH EDGE COLOUR EDGE VISIBILITY EDGE CLIPPING MODE EDGE CAP EDGE JOIN EDGE TYPE CONTINUATION EDGE TYPE INITIAL OFFSET
PATTERN ATTRIBUTES	FILL REFERENCE POINT PATTERN SIZE
OUTPUT CONTROL	AUXILIARY COLOUR TRANSPARENCY MITRE LIMIT

Table 6 — Inheritance filter selection names for attributes (concluded)

Attribute Group Name	Individual Attribute Name
PICK IDENTIFIER	PICK IDENTIFIER
SYMBOL ATTRIBUTES	SYMBOL LIBRARY INDEX SYMBOL COLOUR SYMBOL SIZE SYMBOL ORIENTATION
ALL ATTRIBUTES AND CONTROL ALL	All attributes and control elements All attributes, control elements and ASFs

Table 7 — Inheritance filter selection names for Aspect Source Flags

ASF Group Name	Individual ASF Name
LINE ASFS	LINE TYPE ASF LINE WIDTH ASF LINE COLOUR ASF
MARKER ASFS	MARKER TYPE ASF MARKER SIZE ASF MARKER COLOUR ASF
TEXT ASFS	TEXT FONT INDEX ASF TEXT PRECISION ASF CHARACTER EXPANSION FACTOR ASF CHARACTER SPACING ASF TEXT COLOUR ASF
FILL ASFS	INTERIOR STYLE ASF FILL COLOUR ASF HATCH INDEX ASF PATTERN INDEX ASF
EDGE ASFS	EDGE TYPE ASF EDGE WIDTH ASF EDGE COLOUR ASF
ALL ASFS	All aspect source flags

An example of the COPY SEGMENT element with the INHERITANCE FILTER element is as follows:

```
BEGIN METAFILE "…"
  .
  .
BEGIN SEGMENT (1)
      LINE COLOUR (blue)
      POLYLINE                                                        blue solid line
END SEGMENT

BEGIN DEFAULTS REPLACEMENT
      LINE TYPE (dash)
```

```
END DEFAULTS REPLACEMENT

BEGIN SEGMENT (2)
      LINE COLOUR (red)
      INHHERITANCE FILTER (LINE ATTRIBUTES,STATE LIST)
      COPY SEGMENT (1)                                         red dashed line
      POLYLINE                                                 red dashed line
      INHERITANCE
      FILTER (LINE ATTRIBUTES,SEGMENT)
      COPY SEGMENT (1)                                         blue solid line

      POLYLINE                                                 red dashed line
END SEGMENT

BEGIN PICTURE "…"
BEGIN PICTURE BODY
LINE COLOUR (green)
INHERITANCE FILTER (LINE ATTRIBUTES,SEGMENT)
COPY SEGMENT (2)                                               red dashed line
                                                               red dashed line
                                                               blue solid line
                                                               red dashed line
POLYLINE                                                       green dashed line
INHERITANCE FILTER (LINE ATTRIBUTES,STATE LIST)
COPY SEGMENT (2)                                               green dashed line
                                                               green dashed line
                                                               green dashed line
                                                               green dashed line
BEGIN SEGMENT (3)
      LINE COLOUR (red)
      COPY SEGMENT (1)                                         red dashed line
      INHERITANCE FILTER (LINE ATTRIBUTES,SEGMENT)
      COPY SEGMENT (1)                                         blue solid line
END SEGMENT
  .
  .
LINE COLOUR (green)
COPY SEGMENT (3)                                               red dashed line
                                                               blue solid line
INHERITANCE FILTER (LINE ATTRIBUTES,STATE LIST)
COPY SEGMENT (3)                                               green dashed line
                                                               green dashed line
  .
  .
  .
END PICTURE
END METAFILE
```

The description of clipping in this subclause also applies to the protection region used for clipping and shielding of areas. Clipping is not included in the INHERITANCE FILTER. There is a separate element that controls clipping behaviour CLIP INHERITANCE. Its values may be either ‘state list' or ‘intersection'.

If the value is ‘state list', then the clip rectangle associated with primitives in the copied segment is that of the last CLIP RECTANGLE encountered during interpretation in the metafile element sequence prior to the COPY SEGMENT element, that is, the value in the "modal state list".

If the value is ‘intersection' and if both the modal state list clip indicator and the clip indicator associated with the primitives of the copied segment are ‘on', then the resulting clipping boundary is the intersection of the modal state list clip rectangle with the clipping boundary resulting from the application of the copy transformation to the clip rectangle associated with the primitives. If either indicator is ‘off', then there is no contribution from its associated clip rectangle. To illustrate: if TA and TB are copy transformations:

```
BEGIN SEGMENT A
    CLIP INDICATOR(ON)
    CLIP RECTANGLE R1
    POLYLINE P1
END SEGMENT

CLIP INHERITANCE (INTERSECTION)
CLIP INDICATOR(ON)
CLIP RECTANGLE R2
POLYLINE P2
COPY SEGMENT (A,TA)
POLYLINE P3
```

P2 and P3 are clipped by R2, P1 is clipped by R2 (intersected with) TA(R1). This clipping region may turn out to be an 8-sided convex polygon, if TA causes rotation and skewing.

The composition of clipping rectangles continues however many levels the segment hierarchy is nested. For example:

```
BEGIN SEGMENT A
    CLIP RECTANGLE R0
    POLYLINE P0
    CLIP RECTANGLE  R1
    POLYLINE P1
END SEGMENT

BEGIN SEGMENT B
    CLIP RECTANGLE R2
    POLYLINE P2
    CLIP INHERITANCE (INTERSECTION)
    COPY SEGMENT (A,TA)
END SEGMENT

CLIP RECTANGLE R3
CLIP INHERITANCE (INTERSECTION)
COPY SEGMENT (B,TB)
POLYLINE P3
```

The effective clipping "rectangles" are:

for P0:	TB(R2 intersection TA(R0)) intersection R3
for P1:	TB(R2 intersection TA(R1)) intersection R3
for P2:	TB(R2) intersection R3
for P3:	R3

From this example, it can be seen that the effective clipping "rectangle" can in fact be an arbitrary convex polygon. Annex D contains recommended fallback procedures for interpreters which cannot perform such clipping.

Segment Transformations are never applied to clipping boundaries. The default value for CLIP INHERITANCE is ‘state list'.

6.11 Metafile states

There are a number of required sequential relationships between metafile elements, which determine whether it is syntactically correct. For example, the Metafile Descriptor (which is the first sequence of consecutive elements classified as Metafile Descriptor elements) shall occur in a metafile after the BEGIN METAFILE element and before any other elements (disregarding external and escape elements).

Conceptually, any metafile generator or interpreter may consider that the sequence of pictures and actions represented by the metafile imply changes of state in a virtual device. The valid sequences of metafile elements can be therefore documented by means of a state diagram.

For the purposes of illustrating state relationships, the only significant capability of this hypothetical device is the ability to traverse the metafile data structure from beginning to end and to identify or comprehend (as opposed to interpret, render, or display) the metafile elements. The only significant structural component of the machine is a "state register". The identification by this abstract machine of various metafile elements in the sequential data structure causes the state register to assume certain values.

The defined set of metafile "states" can then be considered to be the values of the state register of this abstract device. For Version 1 and Version 2 metafiles, the major states and the elements causing state transitions are illustrated in the state diagrams, figure 28 and figure 29. The state transitions for the minor states are defined in table 8b.

The states in which each element is allowed for Version 1, Version 2, Version 3, and Version 4 metafiles are described in table 8a and table 8b. These tables also show the lowest metafile version for which each element is defined. Whereas figure 28 and figure 29 define the elements causing state transitions for major metafile states, these definitions are contained in the text of clause 7 for the minor metafile states (see 7.1).

This presentation of metafile states and explanation in terms of abstract interpreters is solely for the purpose of illustrating and clarifying the rules of sequentiality of metafile elements. It is in no way intended to mandate the behaviour or structure of actual metafile generators and interpreters.

Table 8a — CGM Elements by their allowed states – Major States

		CGM Major States								
CGM Element	ver$_1$	PCS	MDS	DR$_3$	GSS DSS	PDS	POS	LSS	SDS	SOS
		v1$_{(2)}$	v1	v1	v2	v1	v1	v2	v4	v4
BEGIN METAFILE$_4$	1									
END METAFILE	1	X	X							
BEGIN PICTURE	1	X	X							
BEGIN PICTURE BODY	1					X				
END PICTURE;	1						X			
BEGIN SEGMENT$_{(5)}$, v2	2		X				X			X
BEGIN SEGMENT, v3/4	2		X			X	X			X
END SEGMENT	2				X			X		
BEGIN FIGURE	2				X		X	X		X
END FIGURE	2									
BEGIN PROTECTION REGION	3				X		X	X		X
END PROTECTION REGION	3									
BEGIN COMPOUND TEXT PATH	3				X		X	X		X
END COMPOUND TEXT PATH	3									
BEGIN TILE ARRAY	3						X			X
END TILE ARRAY	3									
BEGIN APPLICATION STRUCTURE	4						X			X
BEGIN APPLICATION STRUCTURE BODY	4								X	
END APPLICATION STRUCTURE	4									X
METAFILE VERSION	1		X							
METAFILE DESCRIPTION	1		X							
VDC TYPE	1		X							
INTEGER PRECISION	1		X							
REAL PRECISION	1		X							
INDEX PRECISION	1		X							
COLOUR PRECISION	1		X							
COLOUR INDEX PRECISION	1		X							
MAXIMUM COLOUR INDEX	1		X							
COLOUR VALUE EXTENT	1		X							
METAFILE ELEMENT LIST	1		X							
METAFILE DEFAULTS REPLACEMENT	1		X							
FONT LIST	1		X							
CHARACTER SET LIST	1		X							
CHARACTER CODING ANNOUNCER	1		X							
NAME PRECISION	2		X							
MAXIMUM VDC EXTENT	2		X							
SEGMENT PRIORITY EXTENT	2		X							
COLOUR MODEL	3		X							
COLOUR CALIBRATION	3		X							
FONT PROPERTIES	3		x							
GLYPH MAPPING	3		X							
SYMBOL LIBRARY LIST	3		X							
PICTURE DIRECTORY	4		X							
SCALING MODE	1			X		X				
COLOUR SELECTION MODE$_{(5)}$, v1	1			X		X				

Table 8a — CGM Elements by their allowed states – Major States (continued)

		CGM Major States								
CGM Element	ver$_{(1)}$	PCS	MDS	DR$_{3}$	GSS DSS	PDS	POS	LSS	SDS	SOS
		v1$_{(2)}$	v1	v1	v2	v1	v1	v2	v4	v4
COLOUR SELECTION MODE$_{(5)}$, v2/3/4	1			X	X	X	X	X		X
LINE WIDTH SPECIFICATION MODE$_{(5)}$, v1	1			X		X				
LINE WIDTH SPECIFICATION MODE, v2/3/4	1			X	X	X	X	X		X
MARKER SIZE SPECIFICATION MODE$_{(5)}$, v1	1			X		X				
MARKER SIZE SPECIFICATION MODE, v2/3/4	1			X	X	X	X	X		X
EDGE WIDTH SPECIFICATION MODE$_{(5)}$, v1	1			X		X				
EDGE WIDTH SPECIFICATION MODE$_{(5)}$,v2/3/4	1			X	X	X	X	X		X
VDC EXTENT	1			X		X				
BACKGROUND COLOUR	1			X		X				
DEVICE VIEWPORT	2			X		X				
DEVICE VIEWPORT MAPPING	2			X		X				
DEVICE VIEWPORT SPECIFICATION MODE	2			X		X				
LINE REPRESENTATION	2			X		X				
MARKER REPRESENTATION	2			X		X				
TEXT REPRESENTATION	2			X		X				
FILL REPRESENTATION	2			X		X				
EDGE REPRESENTATION	2			X		X				
INTERIOR STYLE SPECIFICATION MODE	3			X	X	X	X	X		X
LINE AND EDGE TYPE DEFINITION	3			X		X				
HATCH STYLE DEFINITION	3			X		X				
GEOMETRIC PATTERN DEFINITION	3			X		X				
APPLICATION STRUCTURE DIRECTORY	4					X				
VDC INTEGER PRECISION	1			X	X		X	X		X
VDC REAL PRECISION	1			X	X		X	X		X
AUXILIARY COLOUR	1			X	X		X	X		X
TRANSPARENCY	1			X	X		X	X		X
CLIP RECTANGLE	1			X	X		X	X		X
CLIP INDICATOR	1			X	X		X	X		X
LINE CLIPPING MODE	2			X	X		X	X		X
MARKER CLIPPING MODE	2			X	X		X	X		X
EDGE CLIPPING MODE	2			X	X		X	X		X
NEW REGION	2									
SAVE PRIMITIVE CONTEXT	2				X		X	X		X
RESTORE PRIMITIVE CONTEXT	2				X		X	X		X
PROTECTION REGION INDICATOR	3			X	X		X	X		X
GENERALIZED TEXT PATH MODE	3			X	X		X	X		X
MITRE LIMIT	3			X	X		X	X		X
TRANSPARENT CELL COLOUR	3			X	X	X	X	X		X
POLYLINE	1				X		X	X		X
DISJOINT POLYLINE	1				X		X	X		X
POLYMARKER	1				X		X	X		X
TEXT	1				X		X	X		X
RESTRICTED TEXT	1				X		X	X		X
APPEND TEXT	1									
POLYGON	1				X		X	X		X
POLYGON SET	1				X		X	X		X
CELL ARRAY	1				X		X	X		X
GDP	1				X		X	X		X
RECTANGLE	1				X		X	X		X
CIRCLE	1				X		X	X		X

Table 8a — CGM Elements by their allowed states – Major States (continued)

		CGM Major States								
CGM Element	ver[1]	PCS	MDS	DR[3]	GSS DSS	PDS	POS	LSS	SDS	SOS
		v1[2]	v1	v1	v2	v1	v1	v2	v4	v4
CIRCULAR ARC 3 POINT	1				X		X	X		X
CIRCULAR ARC 3 POINT CLOSE	1				X		X	X		X
CIRCULAR ARC CENTRE	1				X		X	X		X
CIRCULAR ARC CENTRE CLOSE	1				X		X	X		X
ELLIPSE	1				X		X	X		X
ELLIPTICAL ARC	1				X		X	X		X
ELLIPTICAL ARC CLOSE	1				X		X	X		X
CIRCULAR ARC CENTRE REVERSED	2				X		X	X		X
CONNECTING EDGE	2									
HYPERBOLIC ARC	3				X		X	X		X
PARABOLIC ARC	3				X		X	X		X
NON-UNIFORM B-SPLINE	3				X		X	X		
NON-UNIFORM RATIONAL B-SPLINE	3				X		X	X		X
POLYBEZIER	3				X		X	X		X
POLYSYMBOL	3				X		X	X		X
BITONAL TILE	3									
TILE	3									
LINE BUNDLE INDEX	1			X	X		X	X		X
LINE TYPE	1			X	X		X	X		X
LINE WIDTH	1			X	X		X	X		X
LINE COLOUR	1			X	X		X	X		X
MARKER BUNDLE INDEX	1			X	X		X	X		X
MARKER TYPE	1			X	X		X	X		X
MARKER SIZE	1			X	X		X	X		X
MARKER COLOUR	1			X	X		X	X		X
TEXT BUNDLE INDEX	1			X	X		X	X		X
TEXT FONT INDEX	1			X	X		X	X		X
TEXT PRECISION	1			X	X		X	X		X
CHARACTER EXPANSION FACTOR	1			X	X		X	X		X
CHARACTER SPACING	1			X	X		X	X		X
TEXT COLOUR	1			X	X		X	X		X
CHARACTER HEIGHT	1			X	X		X	X		X
CHARACTER ORIENTATION	1			X	X		X	X		X
TEXT PATH	1			X	X		X	X		X
TEXT ALIGNMENT	1			X	X		X	X		X
CHARACTER SET INDEX	1			X	X		X	X		X
A]LTERNATE CHARACTER SET INDEX	1			X	X		X	X		X
FILL BUNDLE INDEX	1			X	X		X	X		X
INTERIOR STYLE	1			X	X		X	X		X
FILL COLOUR	1			X	X		X	X		X
HATCH INDEX	1			X	X		X	X		X
PATTERN INDEX	1			X	X		X	X		X
EDGE BUNDLE INDEX	1			X	X		X	X		X
EDGE TYPE	1			X	X		X	X		X
EDGE WIDTH	1			X	X		X	X		X
EDGE COLOUR	1			X	X		X	X		X
EDGE VISIBILITY	1			X	X		X	X		X
FILL REFERENCE POINT	1			X	X		X	X		X
PATTERN TABLE[5], v1	1			X			X			X
PATTERN TABLE[5], v2/3/4	1			X		X	X			X
PATTERN SIZE	1			X	X		X	X		X
COLOUR TABLE[5], v1	1			X			X			X
COLOUR TABLE[5], v2/3/4	1			X		X	X			X
ASPECT SOURCE FLAGS	1			X	X		X	X		X
PICK IDENTIFIER	2			X	X		X	X		X

Table 8a — CGM Elements by their allowed states – Major States (concluded)

		CGM Major States								
CGM Element	ver(1)	PCS	MDS	DR3	GSS DSS	PDS	POS	LSS	SDS	SOS
		v1(2)	v1	v1	v2	v1	v1	v2	v4	v4
LINE CAP	3			X	X		X	X		X
LINE JOIN	3			X	X		X	X		X
LINE TYPE CONTINUATION	3			X	X		X	X		X
LINE TYPE INITIAL OFFSET	3			X	X		X	X		X
TEXT SCORE TYPE	3			X	X		X	X		X
RESTRICTED TEXT TYPE	3			X	X		X	X		X
INTERPOLATED INTERIOR	3			X	X		X	X		X
EDGE CAP	3			X	X		X	X		X
EDGE JOIN	3			X	X		X	X		X
EDGE TYPE CONTINUATION	3			X	X		X	X		X
EDGE TYPE INITIAL OFFSET	3			X	X		X	X		X
SYMBOL LIBRARY INDEX	3			X	X		X	X		X
SYMBOL COLOUR	3			X	X		X	X		X
SYMBOL SIZE	3			X	X		X	X		X
SYMBOL ORIENTATION	3			X	X		X	X		X
ESCAPE	1	X	X	X	X	X	X	X		X
MESSAGE	1	X	X	X	X	X	X	X		X
APPLICATION DATA	1	X	X	X	X	X	X	X		X
APPLICATION STRUCTURE ATTRIBUTE	4								X	
COPY SEGMENT	2				X		X	X		X
INHERITANCE FILTER	2			X	X		X	X		X
CLIP INHERITANCE	2			X	X		X	X		X
SEGMENT TRANSFORMATION	2			X	X			X		
SEGMENT HIGHLIGHTING	2			X	X			X		
SEGMENT DISPLAY PRIORITY	2			X	X			X		
SEGMENT PICK PRIORITY	2			X	X			X		

NOTE 1 The "ver" column in the table indicates the lowest metafile version in which the element may appear.

NOTE 2 These entries define the lowest metafile version for which this state is defined. Therefore "v1" indicates the state is defined for Version 1 metafiles (hence also for Version 2, Version 3, and Version 4 metafiles); "v2" indicates that the state is defined for Version 2 metafiles (hence also for Version 3 and Version 4 metafiles); and "v3" indicates that the state is defined for Version 3 metafiles (hence also for Version 4); .and "v4" indicates that the state is defined only for Version 4 metafiles.

NOTE 3 Defaults replacement mode is not really a metafile state, but for implementation purposes it behaves as one and so has been included in this table.

NOTE 4 The Metafile Closed State is not included in this table BEGIN METAFILE is the only element allowed in this state.

NOTE 5 These elements have state rules which are different depending upon the metafile version. For example, COLOUR TABLE is not allowed in the picture body in Version 1 metafiles, but it is allowed in the picture body in Version 2, Version 3 and Version 4 metafiles.

Table 8b — CGM Elements by their allowed states – Minor States

	CGM Minor States				
CGM Elements	FOS v2$_{(1)}$	TOS v1	CPS v3	PRS v3	TAS v3
BEGIN METAFILE					
END METAFILE					
BEGIN PICTURE					
BEGIN PICTURE BODY					
END PICTURE					
END SEGMENT					
BEGIN FIGURE					
END FIGURE	X				
BEGIN PROTECTION REGION					
END PROTECTION REGION				X	
BEGIN COMPOUND LINE					
END COMPOUND LINE			X$_{(2)}$		
BEGIN COMPOUND TEXT PATH					
END COMPOUND TEXT PATH			X$_{(3)}$		
BEGIN TILE ARRAY					
BEGIN APPLICATION STRUCTURE BODY					
END APPLICATION STRUCTURE					
METAFILE VERSION					
METAFILE DESCRIPTION					
VDC TYPE					
INTEGER PRECISION					
REAL PRECISION					
INDEX PRECISION					
COLOUR PRECISION					
COLOUR INDEX PRECISION					
MAXIMUM COLOUR INDEX					
COLOUR VALUE EXTENT					
METAFILE ELEMENT LIST					
METAFILE DEFAULTS REPLACEMENT					
FONT LIST					
CHARACTER SET LIST					
CHARACTER CODING ANNOUNCER					
NAME PRECISION					
MAXIMUM VDC EXTENT					
SEGMENT PRIORITY EXTENT					
COLOUR MODEL					
COLOUR CALIBRATION					
FONT PROPERTIES					
GLYPH MAPPING					
SYMBOL LIBRARY LIST					
PICTURE DIRECTORY					
SCALING MODE					
COLOUR SELECTION MODE					
LINE WIDTH SPECIFICATION MODE					
MARKER SIZE SPECIFICATION MODE					
EDGE WIDTH SPECIFICATION MODE					
VDC EXTENT					
BACKGROUND COLOUR					
DEVICE VIEWPORT					
DEVICE VIEWPORT MAPPING					

Table 8b — CGM Elements by their allowed states – Minor States (continued)

	CGM Minor States				
CGM Elements	FOS v2(1)	TOS v1	CPS v3	PRS v3	TAS v3
DEVICE VIEWPORT SPECIFICATION MODE					
LINE REPRESENTATION					
MARKER REPRESENTATION					
TEXT REPRESENTATION					
FILL REPRESENTATION					
EDGE REPRESENTATION					
INTERIOR STYLE SPECIFICATION MODE;					
LINE AND EDGE TYPE DEFINITION					
HATCH STYLE DEFINITION					
GEOMETRIC PATTERN DEFINITION					
APPLICATION STRUCTURE DIRECTORY					
VDC INTEGER PRECISION	X		X	X	
VDC REAL PRECISION	X		X	X	
AUXILIARY COLOUR	X	X			
TRANSPARENCY	X	X			
CLIP RECTANGLE					
CLIP INDICATOR					
LINE CLIPPING MODE					
MARKER CLIPPING MODE					
EDGE CLIPPING MODE					
NEW REGION	X			X	
SAVE PRIMITIVE CONTEXT					
RESTORE PRIMITIVE CONTEXT					
PROTECTION REGION INDICATOR					
GENERALIZED TEXT PATH MODE					
MITRE LIMIT					
TRANSPARENT CELL COLOUR					
POLYLINE	X		X	X	
DISJOINT POLYLINE	X		X	X	
POLYMARKER					
TEXT					
RESTRICTED TEXT					
APPEND TEXT		X			
POLYGON	X			X	
POLYGON SET	X			X	
CELL ARRAY					
GDP	X		X	X	
RECTANGLE	X			X	
CIRCLE	X			X	
CIRCULAR ARC 3 POINT	X		X	X	
CIRCULAR ARC 3 POINT CLOSE	X			X	
CIRCULAR ARC CENTRE	X		X	X	
CIRCULAR ARC CENTRE CLOSE	X			X	
ELLIPSE	X			X	
ELLIPTICAL ARC	X		X	X	
ELLIPTICAL ARC CLOSE	X			X	
CIRCULAR ARC CENTRE REVERSED	X		X	X	
CONNECTING EDGE	X				
HYPERBOLIC ARC	X		X	X	
PARABOLIC ARC	X		X	X	

Table 8b — CGM Elements by their allowed states – Minor States (continued)

	CGM Minor States				
CGM Elements	FOS v2(1)	TOS v1	CPS v3	PRS v3	TAS v3
NON-UNIFORM B-SPLINE	X		X	X	
NON-UNIFORM RATIONAL B-SPLINE	X		X	X	
POLYBEZIER	X		X	X	
POLYSYMBOL					
BITONAL TILE					X
TILE					X
LINE BUNDLE INDEX					
LINE TYPE					
LINE WIDTH					
LINE COLOUR					
MARKER BUNDLE INDEX					
MARKER TYPE					
MARKER SIZE					
MARKER COLOUR					
TEXT BUNDLE INDEX		X			
TEXT FONT INDEX		X			
TEXT PRECISION		X			
CHARACTER EXPANSION FACTOR		X			
CHARACTER SPACING		X			
TEXT COLOUR		X			
CHARACTER HEIGHT		X			
CHARACTER ORIENTATION					
TEXT PATH					
TEXT ALIGNMENT					
CHARACTER SET INDEX		X			
ALTERNATE CHARACTER SET INDEX		X			
FILL BUNDLE INDEX					
INTERIOR STYLE					
FILL COLOUR					
HATCH INDEX					
PATTERN INDEX					
EDGE BUNDLE INDEX	X				
EDGE TYPE	X				
EDGE WIDTH	X				
EDGE COLOUR	X				
EDGE VISIBILITY	X				
FILL REFERENCE POINT					
PATTERN TABLE					
PATTERN SIZE					
COLOUR TABLE					
ASPECT SOURCE FLAGS	X				
PICK IDENTIFIER					
LINE CAP					
LINE JOIN					
LINE TYPE CONTINUATION					

Table 8b — CGM Elements by their allowed states – Minor States (concluded)

	CGM Minor States				
CGM Elements	FOS v2(1)	TOS v1	CPS v3	PRS v3	TAS v3
LINE TYPE INITIAL OFFSET					
TEXT SCORE TYPE		X			
RESTRICTED TEXT TYPE					
INTERPOLATED INTERIOR					
EDGE CAP					
EDGE JOIN					
EDGE TYPE CONTINUATION					
EDGE TYPE INITIAL OFFSET					
SYMBOL LIBRARY INDEX					
SYMBOL COLOUR					
SYMBOL SIZE					
SYMBOL ORIENTATION					
ESCAPE	X	X	X	X	X
MESSAGE	X		X	X	X
APPLICATION DATA	X		X	X	X
APPLICATION STRUCTURE ATTRIBUTE					
COPY SEGMENT					
INHERITANCE FILTER					
CLIP INHERITANCE					
SEGMENT TRANSFORMATION					
SEGMENT HIGHLIGHTING					
SEGMENT DISPLAY PRIORITY					
SEGMENT PICK PRIORITY					

NOTE 1 These entries define the lowest metafile version for which this state is defined. Therefore "v1" indicates the state is defined for Version 1 metafiles (hence also for Version 2, Version 3, and all higher version metafiles); "v2" indicates that the state is defined for Version 2 metafiles (hence also for Version 3 and all higher version metafiles); "vn" indicates that the state is defined for Version n metafiles (hence also for any higher version metafiles).

NOTE 2 END COMPOUND LINE is only allowed in Compound Path State when that state was entered by the BEGIN COMPOUND LINE element.

NOTE 3 END COMPOUND TEXT PATH is only allowed in Compound Path State when that state was entered by the BEGIN COMPOUND TEXT PATH element.

Major States :

PCS Picture Closed State
MDS Metafile Descriptor State
DR Defaults Replacement Mode
GSS Global Segment State
DSS Picture Descriptor Segment State
PDS Picture Descriptor State
POS Picture Open State
LSS Local Segment State
SDS Structure Descriptor State
SOS Structure Open State

Minor States :

FOS	Figure Open State
TOS	Text Open State
CPS	Compound Path State
PRS	Protection Region State
TAS	Tile Array State

Table 9 — CGM state transitions into and out of minor states

Element	Original state	Final state
BEGIN FIGURE END FIGURE	GSS,LSS,POS,SOS FOS	FOS previous state
TEXT (not-final) RESTRICTED TEXT (not-final) APPEND TEXT (final)	GSS,LSS,POS,SOS GSS,LSS,POS,SOS TOS	TOS TOS previous state
BEGIN COMPOUND LINE END COMPOUND LINE BEGIN COMPOUND TEXT PATH END COMPOUND TEXT PATH	GSS,LSS,POS,SOS CPS GSS,LSS,POS,SOS CPS	CPS previous state CPS previous state
BEGIN PROTECTION REGION END PROTECTION REGION	GSS,LSS,POS,SOS PRS	PRS previous state
BEGIN TILE ARRAY END TILE ARRAY	POS,SOS TAS	TAS previous state

6.12 Registration

For certain elements, the CGM defines value ranges of parameters as being reserved for registration. The meanings of these values will be defined using the established procedures of the ISO International Registration Authority for Graphical Items. These procedures do not apply to values and value ranges defined as being reserved for implementation-dependent or private use; these values and ranges are not standardized.

Applications therefore shall not use parameter values in the reserved ranges for implementation-dependent or private use. Those metafile elements that will be affected by registration of graphical items are

COLOUR MODEL	LINE CAP	EDGE CAP
COLOUR CALIBRATION	LINE JOIN	EDGE JOIN
FONT PROPERTIES	LINE TYPE CONTINUATION	EDGE TYPE CONTINUATION
GLYPH MAPPING	MARKER TYPE	GENERALIZED DRAWING PRIMITIVE
SYMBOL LIBRARY LIST	TEXT SCORE TYPE	ESCAPE
BITONAL TILE	RESTRICTED TEXT TYPE	BEGIN APPLICATION STRUCTURE
TILE	HATCH STYLE	APPLICATION STRUCTURE ATTRIBUTE
LINE TYPE	EDGE TYPE	

Registration of character sets for use with the CHARACTER SET LIST element is according to the procedures established by ISO 2375. Registration of glyph collections is according to the procedures and registry defined by ISO/IEC 10036. ISO/IEC 9541-1 defines procedures for registering naming authorities for fonts, and suggests standard naming conventions to use in specifying or registering fonts (it does not itself, however, establish font registration procedures, a registration authority, or a font registry). A template will be required for BEGIN APPLICATION STRUCTURE and APPLICATION STRUCTURE ATTRIBUTE which describes the particular registered usage of the elements.

6.13 Application Structure Elements

6.13.1 Introduction

Application structures (APS), which provide access to metafiles for applications such as text/graphics integration at levels of granularity finer than the picture level, may be defined in Version 4 metafiles. An APS is defined within a picture body as follows:

```
BEGIN APPLICATION STRUCTURE
        APPLICATION STRUCTURE ATTRIBUTE
        .
        .(Arbitrary number of application structure attributes)
        .
BEGIN APPLICATION STRUCTURE BODY
        .
        .(Graphic primitives, attributes, and control elements in any order)
        .
END APPLICATION STRUCTURE
```

The BEGIN APPLICATION STRUCTURE element has three parameters which define the type, indentifier, and inheritance mechanism for the APS. The APPLICATION STRUCTURE ATTRIBUTE provides the capability for applications to associate non-graphical information with APSs. This element has two parameters which define the type of APS attribute, and the data record containing the actual data.

APSs defined in this manner enable applications to structure and manage the metafile in a manner meaningful to the application.

If a BEGIN APPLICATION STRUCTURE element is encountered with an ‘identifier’ parameter which matches the identifier of another BEGIN APPLICATION STRUCTURE in the same picture, then the second (and any subsequent) occurrence shall be construed as continuing the definition of the APS begun by the first occurrence. The location of the APS in any possible hierarchy of Application Structures shall be defined by the first occurrence of a BEGIN APPLICATION STRUCTURE element with the given identifier. An APS with a given identifier may not be nested within an APS with the same identifier. An APS with a given identifier shall not match the identifier of an APS in another picture in the same metafile. The 'type' and 'inheritance' parameters of the BEGIN APPLICATION STRUCTURE element of any continued APS shall match the values present on the first occurrence of the APS.

6.13.2 Location of and access to Application Structures

One or more APSs may occur within a picture body, or totally within another APS. APSs may be classified into classes or categories meaningful to a particular application by using the application structure type parameter. An APS shall be uniquely identified (using the application structure identifier) within the metafile (see 6.13.1 regarding the mechanism of continuing the definition of Application Structures). An APS may occur only within a picture body but may not occur within a local segment. However, an APS may contain local segments as long as the segments begin and end within the APS body.

6.13.3 Nesting of Application Structures

Application structures may be nested to any level as long as a nested APS is contained completely within the APS in which it is contained.

6.13.4 Graphical Context of Application Structures

The graphic primitive context at the beginning of an APS is determined by the value of the inheritance flag parameter of the BEGIN APPLICATION STRUCTURE element of the first occurrence of the APS with a given identifier. If the value is 'statelist', the APS inherits the current graphic primitive context (context-dependent). If it is 'application structure' the graphic primitive context is reset to the state at the beginning of the picture body (context-independent) in which the APS is defined. The graphic primitive context at the beginning of subsequent occurrences of a context-independent APS with the same value of the identifier parameter shall be the graphic primitive context at the occurrence of the END APPLICATION STRUCTURE element of the immediately prior APS with the same identifier value. At the time its END APPLICATION STRUCTURE element is encountered, a context-independent APS automatically restores the graphic primitive context to the state which existed at the time its BEGIN APPLICATION STRUCTURE element was encountered. The END APPLICATION STRUCTURE element of a context-dependent APS has no effect on the graphic primitive context.

6.13.5 Application Structure Attributes

The APPLICATION STRUCTURE ATTRIBUTE element describes and defines the behaviour of application structures. One or more of these elements may occur immediately after a BEGIN APPLICATION STRUCTURE element and before its corresponding BEGIN APPLICATION STRUCTURE BODY element. APPLICATION STRUCTURE ATTRIBUTE elements may not occur elsewhere in a metafile. The application structure attribute type may be used to classify the information stored in the attribute's structured data record. The value of the application structure attribute type need not be unique. The values are intended to provide important information to the application before it parses the information in the structured data record. The value of the application structure attribute type and the structured data record are unrestricted. It is anticipated that application constituencies will specify the content and structure of the information in the application structure attributes in a profile.

APPLICATION STRUCTURE ATTRIBUTE elements are the only elements that shall occur between the BEGIN APPLICATION STRUCTURE element and its corresponding BEGIN APPLICATION STRUCTURE BODY element.

In the case that the definition of an Application Structure is continued across several disjoint APS fragments in the metafile (see 6.13.1), APPLICATION STRUCTURE ATTRIBUTE elements may occur in the header sections of any of the APS fragments. Such continued or fragmented APS definitions shall ultimately define a single logical application structure. APS attributes shall be construed as associated with the entire structure, i.e., no APPLICATION STRUCTURE ATTRIBUTE element shall be construed as applying only to the fragment in which it occurs.

6.13.6 Relationship between application structures and segments

An application structure shall occur only within a picture body but not within the definition of a local segment. Therefore, an application structure cannot occur within either global or local segment definitions, regardless of where the segments are defined. Copy segment elements may occur within application structures. These copy segment elements may reference either local or global segments. However, a COPY SEGMENT element which occurs within a context independent APS (inheritance flag value of 'application structure') shall only reference segments defined within the metafile descriptor or the picture descriptor. COPY SEGMENT elements within a context independent APS shall not reference segments defined in the picture body.

When a segment is referenced from within an application structure using the COPY SEGMENT element, the graphical context used to render the segment depends on the value of the inheritance flag parameter of the BEGIN APPLICATION STRUCTURE element, the values of the filter selection list parameter of the INHERITANCE FILTER element, and the value of the clip inheritance parameter of the CLIP INHERITANCE element. The graphical context for rendering segments referenced from within an application structure are shown in the following table:

APS inheritance flag	segment inheritance	segment graphical context
state list	state list	context at COPY SEGMENT
application structure	state list	context at COPY SEGMENT
state list	segment	context at segment definition
application structure	segment	context at segment definition

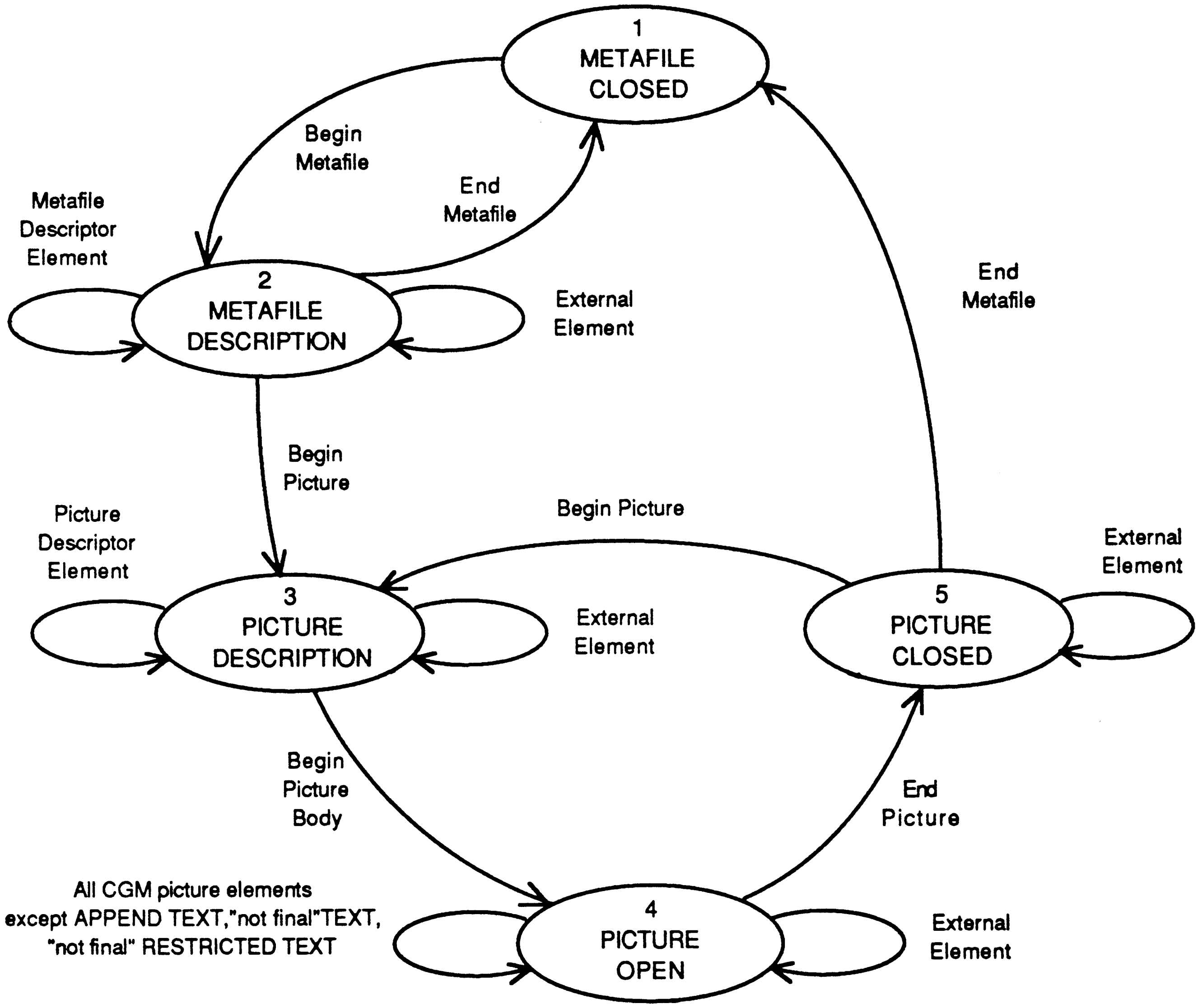
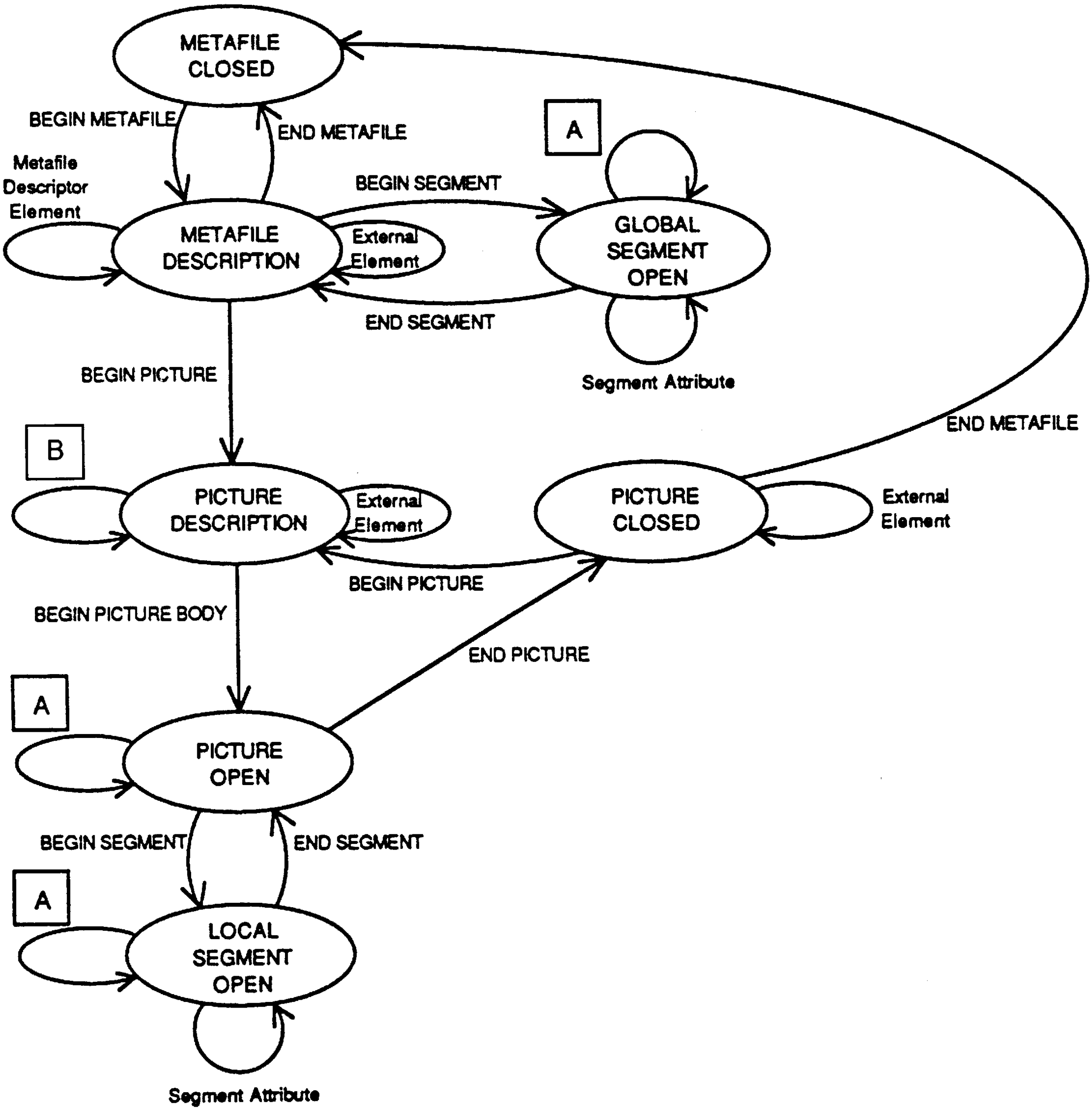


Figure 28 — State diagram for Version 1 metafiles

NOTE 1 "External Element" in figure 28 denotes the set containing both the Escape Elements and the External Elements.



A	All Control, Graphical Primitive, Attribute, Escape and External elements except NEW REGION, CONNENCTING EDGE, APPEND TEXT, “not final” TEXT, “not final” RESTRICTED TEXT, plus COPY SEGMENT, INHERITANCE FILTER, CLIP INHERITANCE, COLOUR SELECTION MODE, LINE WIDTH SPECIFICATION MODE, MARKER SIZE SPECIFICATION MODE, EDGE WIDTH SPECIFICATION MODE, and INTERIOR STYLE SPECIFICATION MODE.
B	Picture Descriptor Element, COLOUR TABLE and PATTERN TABLE.

Figure 29 — State diagram for Version 2 metafiles.

NOTE 2 "External Element" in figure 29 denotes the set containing both the Escape Elements and the External Elements.

7 Abstract specification of elements

7.1 Introduction

The metafile elements are discussed in this clause.

The Delimiter Elements (see 7.2) delimit significant structures within the Metafile.

The Metafile Descriptor Elements (see 7.3) describe the functional content, default conditions, identification, and characteristics of the CGM.

The Picture Descriptor Elements (see 7.4) define the extent of VDC space and declare the parameter modes of attribute elements for the entire picture.

The Control Elements (see 7.5) specify size and precision of coordinate space and format descriptions of the CGM elements.

The Graphical Primitive Elemenüs (see 7.6) describe geometric objects in the CGM.

The Attribute Elements (see 7.7) describe the appearance of graphical primitive elements.

The Escape Elements (see 7.8) describe device- and system-dependent elements used to construct a picture.

The External Elements (see 7.9) communicate information not directly related to the generation of a graphical image.

The Segment Elements (see 7.10) provide for the grouping and manipulation of elements.

The format used throughout this clause to define the Metafile element set separates functionality from coding. Each element is named, the parameters are described, data types are listed, and a description of implicit relationships is added to clarify how the element fits into the system.

The metafile version in which an element may appear is specified in table 8a. Figure 28 and figure 29 define the elements causing state transitions for major metafile states. These definitions are contained in the text of this clause for the minor metafile states (see table 8b). For elements whose state restrictions differ depending on the metafile version, the state restrictions are described in the individual element descriptions of this clause.

The data types used in this clause and elsewhere in this International Standard are defined in table 8b.

Type IX parameters, used as enumeration selectors in some elements, have a fixed number of values with defined and standardized meanings, and have other values available for implementation-dependent definition and use. The standardized values may be expanded in future versions of the CGM. To avoid possible conflict with user-defined values, the standardized and user-available values are assigned to distinct ranges of the IX parameter. Negative values of IX are allocated for private or implementation-dependent meanings for those Version 1 and Version 2 elements which allow private values, and non-negative values are reserved for (future) standardization.

Type SF (string fixed) parameters are not subject to the character set control and switching mechanisms of type S parameters (see 6.3.4.5).

Combinations of simple types can also be used where n is an unspecified number (for example, nP or 2R,2I). Also, lists of types can be expressed (for example, I,E,R,E).

How these data types are represented in a given encoding of the CGM is specified in the subsequent encoding parts of ISO/IEC 8632.

Table 10 — Data type definitions and abbreviations

Data Types		Meaning
CI	Colour Index	Non-negative integer pointer into a table of colour values.
CCO	Colour component	One component of a colour direct value.
CD	Colour Direct	Three-tuple or four-tuple of CCO values (as determined by COLOUR MODEL) for colour definition within one of the supported colour models.
CO	Colour	CI, if the value of COLOUR SELECTION MODE is 'indexed'; ;;CD, if the value of COLOUR SELECTION MODE is 'direct'
E	Enumerated	Set of standardized values. The set is defined by enumerating ;;the identifiers that denote the values.
I	Integer	Number with no fractional part.
IX	Index	Integer pointer into a table of values, or integer used to select from among a set of enumerated values.
P	Point	Two VDC values representing the x and y coordinates of a point in VDC space.
R	Real	Number with integer and fractional portion, only one of which need exist.
S	String	Sequence of characters.
VDC	VDC value	Single real or integer value (as determined by VDC TYPE) in VDC space.
SS	Size Specification	VDC if applicable specification mode is 'absolute', otherwise real (R). SS applies to such metafile aspects as line width and marker size. ;See table 10 for resolution of SS (to VDC or R) for each affected primitive and aspect.
D	Data Record	User-defined and otherwise non-standardized record of data that accompanies elements such as APPLICATION DATA, ESCAPE, and GENERALIZED DRAWING PRIMITIVE.
N	Name	Identifier for a segment, pick or context. Realization is integer. Range is dependent on NAME PRECISION.
VC	Viewport Coordinate	Single real or integer value as determined by the DEVICE VIEWPORT SPECIFICATION MODE: — R, fraction [0..1] of default viewport — I, millimetres (scaled) ..—.I, native device units.
VP	Viewport Point	Two VC values representing the x and y coordinates of a point in viewport specification space.
UI8	8-bit Unsigned Integer	An unsigned integer in the range 0..255, represented in each of the encodings with a precision equivalent to 8 binary bits.
UI16	16-bit Unsigned Integer	An unsigned integer in the range 0..(2^{16} - 1), represented in each of the encodings with a precision equivalent to 16 binary bits.
UI32	32-bit Unsigned Integer	An unsigned integer in the range 0..(2^{32} - 1), represented in each of the encodings with a precision equivalent to 32 binary bits.
BS	Bitstream	A binary data object, given an encoding-dependent representation in each of the encodings (part 3, part 4), which consists of a compressed stream of the binary representations of other CGM datatypes (e.g., colours), compressed according to one of a number of standardized techniques defined in this part of this International Standard.
SDR	Structured Data Record	A record of data comprised of a list of zero or more members. Each member is a typed sequence of data elements of the same data type. A typed sequence contains: a data type indicator, a data count, and that many items of the indicated type. The type may be SDR itself, or one of the above data types. See annex C for the precise definition of the structure and grammar of SDR.
SF	String Fixed	Sequence of characters, comprising string parameters of non-graphical text strings, not subject to character attributes and controls.

Data type CO is not a basic data type — it represents either CI or CD depending upon the current value of the COLOUR SELECTION MODE element. The colour selection mode is the same for all primitives and attributes at any given point in the metafile. Similarly, data type SS is not a basic data type — it represents either VDC or R, depending upon the value of an associated specification mode. The relevant specification mode is either LINE WIDTH SPECIFICATION MODE, MARKER SIZE SPECIFICATION MODE, EDGE WIDTH SPECIFICATION MODE, or INTERIOR STYLE SPECIFICATION MODE, depending on the particular attribute.

These specification modes may have the values 'scaled', 'absolute', 'fractional', or 'mm'. When the value is 'absolute', then an associated parameter of type SS resolves to the basic data type VDC. Otherwise, associated SS parameters resolve to the basic data type R.

Table 11 defines which specification mode controls the resolution of SS, to VDC or R, for each metafile parameter whose data type is SS.

Table 11 — Specification mode contolling SS resolution for each affected aspect

Specification Mode	Affected Parameters	
	Element	**Parameter**
LINE WIDTH SPECIFICATION MODE	LINE REPRESENTATION LINE AND EDGE TYPE DEFINITION LINE WIDTH	line width specifier dash cycle repeat length line width specifier
MARKER SIZE SPECIFICATION MODE	MARKER REPRESENTATION MARKER SIZE	marker size specifier marker size specifier
EDGE WIDTH SPECIFICATION MODE	EDGE REPRESENTATION EDGE WIDTH	edge width specifier edge width specifier
INTERIOR STYLE SPECIFICATION MODE	HATCH STYLE DEFINITION HATCH STYLE DEFINITION PATTERN SIZE INTERPOLATED INTERIOR	hatch direction vectors specifier duty cycle length pattern size specifier reference geometry

ISO/IEC 8632 defines the syntax and semantics of the elements that may occur in a metafile. To aid designers of metafile interpreters in achieving uniformity of results, three categories of errors and degeneracies in metafile contents are identified.

a) syntax errors;

b) geometric degeneracies;

c) mathematical singularities and ambiguities.

Annex D defines each of these, identifies some of specific occurrences and contains suggestions for reasonable interpretations.

7.2 Delimiter elements

7.2.1 BEGIN METAFILE

Parameters :

identifier (SF)

Description :

This is the first element of a metafile. It demarcates the beginning of the Metafile Descriptor. BEGIN METAFILE shall occur exactly once in a metafile. The *identifier* parameter is available for use by metafile generators and interpreters in a manner that is not further standardized.

NOTE If more than one CGM is to be recorded on the same output medium, each shall be addressed by reference to its BEGIN METAFILE element.

References :

6.2

7.2.2 END METAFILE

Parameters:

None

Description:

This is the last element of a metafile.

END METAFILE shall occur exactly once in a metafile.

References:

6.2

7.2.3 BEGIN PICTURE

Parameters:

identifier (SF)

Description:

This is the first element of a picture.

It demarcates the beginning of the Picture Descriptor. It forces all picture descriptor, control, and attribute elements which are subject to default to return to the default values. The *identifier* parameter is available for use by metafile generators and interpreters in a manner that is not further standardized.

For compatibility with ISO 2022, designating and invoking controls which may occur within the string parameters of TEXT, APPEND TEXT, RESTRICTED TEXT and GENERALIZED DRAWING PRIMITIVE elements, the way that BEGIN PICTURE forces the character set to assume its default value is as follows:

BEGIN PICTURE causes the character set selected by the default value of CHARACTER SET INDEX to be designated as the current G0 set and invoked into positions 2/1 through 7/14 of the 7-bit or 8-bit code chart.

BEGIN PICTURE also designates the character set selected by the default value of ALTERNATE CHARACTER SET INDEX as the current G1 set and also as the current G2 set.

In an 8-bit environment, BEGIN PICTURE invokes that default G1 set into code chart positions 10/1 to 15/14 (or 10/0 to 15/15 if the G1 set is a 96-character set).

Here, the terms "designate", "invoke", "G0 set", "G1 set", and "G2 set" have the meanings defined for them in ISO 2022. .

NOTE BEGIN PICTURE and END PICTURE bound the set of elements of a single picture in the CGM. Every picture in a metafile is totally independent from every other picture and always starts with a BEGIN PICTURE. This independence is enforced by returning the modal values of all elements to their default values at the start of the picture.

References:

6.2

7.2.4 BEGIN PICTURE BODY

Parameters:

None

Description:

This element demarcates the end of the Picture Descriptor and the beginning of the body of the picture. It thus informs the metafile interpreter of the transition from the Picture Descriptor to the graphical primitive, attribute, and control elements that define the picture.

If a new picture begins with a cleared view surface, the initial colour of the view surface is the colour specified by the BACKGROUND COLOUR element, if that element is present in the Picture Descriptor, or by the default background colour, if the BACKGROUND COLOUR element is not present in the Picture Descriptor.

Each picture defines a graphical image independent of the other pictures. As suggested in annex D, presentation of each picture on a cleared view surface is the most expected action. Because view surface clearing is not standardized, interpreters are free to compose images by overlaying pictures.

References:

6.2
D.4.1

7.2.5 END PICTURE

Parameters:

None

Description:

This is the last element of a picture.

Only external and escape elements may occur between END PICTURE and BEGIN PICTURE or between END PICTURE and END METAFILE.

References:

6.2
D.4.1

7.2.6 BEGIN SEGMENT

Parameters:

segment identifier (N)

Description:

This is the first element of a segment. All subsequent elements until the next END SEGMENT will belong to this segment.

References:

6.2
6.10.3

7.2.7 END SEGMENT

Parameters:

None

Description:

This is the last element of a segment. Subsequent elements will no longer belong to a segment.

References:

6.2
6.10.3

7.2.8 BEGIN FIGURE

Parameters:

none

Description:

This is the first element of a closed figure. All subsequent elements until the next END FIGURE will be part of the closed figure.

References:

6.2
6.6.11

7.2.9 END FIGURE

Parameters:

none

Description:

This element terminates the current closed figure.

If the current region has not yet been closed by a preceding NEW REGION element and if the last point of the last line element is not coincident with the current closure point, then the current subregion is closed by a line segment connecting the last point of the preceding line element to the current closure point. This line becomes a part of the implicit boundary specification. If the END FIGURE was preceded by a CONNECTING EDGE element, which was itself preceded by a line primitive, then this line also becomes part of the edge specification. If the region which has been previously closed is empty, or if the last point of the last line element is coincident with the current closure point, or if the last element was a filled-area primitive, then no line segment is generated by this element.

References:

6.2
6.6.11

7.2.10 BEGIN PROTECTION REGION

Parameters:

region index (IX)

Description:

Line and fill primitives which are present between the BEGIN PROTECTION REGION and END PROTECTION REGION are used to construct a protection region. The region is used either for clipping or for shielding, as specified by the PROTECTION REGION INDICATOR element. The defined region is associated with the region index parameter, by which it may subsequently be referenced by the PROTECTION REGION INDICATOR element. Legal values of the region index parameter are positive integers.

References:

6.2
6.5.2
6.5.4

7.2.11 END PROTECTION REGION

Parameters:

None

Description:

This is the last element of a protection region definition.

References:

6.2
6.5.2
6.5.4

7.2.12 BEGIN COMPOUND LINE

Parameters:

none

Description:

This is the first element of a compound line. All subsequent elements until the next END COMPOUND LINE will be part of the compound line. The compound line entity will have consistent line attributes and will be treated as a single line primitive. Line attributes shall not be changed while constructing a compound line.

References:

6.2
6.6.1.2

7.2.13 END COMPOUND LINE

Parameters:

None

Description:

This is the last element of a compound line definition.

References:

6.2
6.6.1.2

7.2.14 BEGIN COMPOUND TEXT PATH

Parameters:

none

Description:

This is the first element of a compound text path. All subsequent elements until the next END COMPOUND TEXT PATH will be part of the compound text path. Attributes shall not appear within the definition of the compound text path. The display of text along the compound text path is described in clause 6.

References:

6.2
6.5.5
6.7.3.2
6.7.8

7.2.15 END COMPOUND TEXT PATH

Parameters:

None

Description:

This is the last element of a compound text path definition.

References:

6.2
6.5.5
6.7.3.2
6.7.8

7.2.16 BEGIN TILE ARRAY

Parameters:

position (P)
cell path direction (one of: 0, 90, 180, 270) (E)
line progression direction (one of: 90, 270) (E)
number of tiles in path direction (I)
number of tiles in line direction (I)
number of cells/tile in path direction (I)
number of cells/tile in line direction (I)

cell size in path direction (R)
cell size in line direction (R)
image offset in path direction (I)
image offset in line direction (I)
image number of cells in path direction (I)
image number of cells in line direction (I)

Description:

A tile array is defined as follows:

The point specified by the *position* parameter is used to place the tile array. The corner of the first cell with information content in the first tile is placed at the specified point.

The *cell path direction* parameter defines the direction of progression of successive cells along a line relative to the VDC x-axis. The *line progression direction* parameter defines the direction of progression of successive cell lines and is expressed as a direction relative to the cell path direction.

The values of the *cell size in path direction* and *cell size in line direction* parameters are defined in units of number of cells per VDC unit.

The values of the *cell size in path direction* and *number of cells/tile path direction* parameters together define the length and granularity for each line in the tile, hence the tile size in the cell path direction. The *cell size in line direction* and *number of cells/tile in line direction* parameters together implicitly define the tile size in the line progression direction.

The tiles in the tile array define a rectangular region of VDC space — a "tiling space". The actual graphical image, which is that portion of the tile array with information content, need not (in fact in large tiled images probably will not) occupy the full rectangle. The image offset and image number of cells parameters specify the rectangle within the tiling space which actually has information content. See figure 5.

All cells in all tiles of the tile array are encoded, regardless of whether or not they have information content.

NOTE It is recommended that metafile generators set these "border" cells, which have no information content, identically to the background colour. Metafile interpreters should not draw these border cells.

References:

6.2
6.6.5.2
D.4.5.13

7.2.17 END TILE ARRAY

Parameters:

None

Description:

This is the last element of a tile array definition.

References:

6.2
6.6.5.2
D.4.5.13

7.2.18 BEGIN APPLICATION STRUCTURE

parameters:

application structure identifier (SF)
application structure type (SF)
inheritance flag (one of: statelist, application structure) (E)

Description:

This is the first element of an application structure. All subsequent elements until the matching END APPLICATION STRUCTURE element belong to this application structure.

The Application Structure Identifier is available for use by generators and interpreters in a manner that is not standardized except that this identifier shall uniquely identify the application structure within the scope of the metafile. See 6.13.1 regarding the mechanism for continuing the definition of Application Structures, by repeating the use of the unique APS identifier on subsequent BEGIN APPLICATION STRUCTURE elements within the same picture. The Application Structure Type is available for classifying APSs into groups meaningful to the application. There are no restrictions on the values of this parameter. See 6.13.4 regarding the inheritance of graphic primitive context in Application Structures with continued definitions.

The inheritance flag parameter determines the graphic primitive context in which the elements between the BEGIN APPLICATION STRUCTURE and END APPLICATION STRUCTURE elements will be realized. If the value of the inheritance flag parameter is 'state list', then the application structure will inherit the graphic primitive context that is current within the metafile and the END APPLICATION STRUCTURE element that completes the application structure definition will have no effect on the graphic primitive context. If the inheritance flag is 'application structure', the current graphical primitive context will be suspended and replaced by the context at the BEGIN PICTURE BODY element of the picture in which the APS occurs. The suspended context will be restored at the END APPLICATION STRUCTURE element that completes the application structure definition.

References:

6.2

7.2.19 BEGIN APPLICATION STRUCTURE BODY

Parameters:

None

Description:

This element demarcates the application structure attributes and the beginning of the graphical primitives in the application structure. It thus informs the metafile interpreter of the transition from the application structure attributes to the graphical primitive, attribute, and control elements that define the application structure.

References:

6.2

7.2.20 END APPLICATION STRUCTURE

Parameters:

None

Description:

This is the last element of an application structure. Subsequent elements will not belong to this application structure. However, the definition of an Application Structure may be continued using the mechanism described in 6.13.13. At the time this element is encountered, the graphic primitive context is automatically restored to the state which existed at the time the corresponding BEGIN APPLICATION STRUCTURE element was encountered if the inheritance flag parameter of the BEGIN APPLICATION STRUCTURE was set to 'application structure' . If the inheritance flag parameter was set to 'state list' the END APPLICATION STRUCTURE element has no effect on the graphic primitive context.

References:

6.2

7.3 Metafile descriptor elements

7.3.1 METAFILE VERSION

Parameters:

version (I)

Description:

The metafile conforms to the specified version of the CGM Standard. This element shall occur in the Metafile Descriptor of every metafile.

METAFILE VERSION shall appear exactly once in the Metafile Descriptor.

References:

6.3.1

7.3.2 METAFILE DESCRIPTION

Parameters:

description (SF)

Description:

The contents of the metafile are described in a non-standardized way by this entry.

NOTE This element allows the CGM to be identified with such descriptive text as generating product and version, time and date, author, place of origin, etc.

References:

6.3.1

7.3.3 VDC TYPE

Parameters:

vdc type (one of: integer, real) (E)

Description:

The *vdc type* parameter is an enumerative value that declares the data type, integer or real, of the Virtual Device Coordinates.

References:

6.3

7.3.4 INTEGER PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The precision for operands of data type integer (I) is specified for subsequent data of type I. The precision is defined as the field width measured in units applicable to the specific encoding.

References:

6.3

7.3.5 REAL PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The precision for operands of data type real (R) is specified for subsequent data of type R. The precision is defined as the field width measured in units applicable to the specific encoding. The precision may consist of parameters that define subfields of data type R.

References:

6.3

7.3.6 INDEX PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The precision for operands of data type index (IX) is specified for subsequent data of type IX. The precision is defined as the field width measured in units applicable to the specific encoding.

References:

6.3

7.3.7 COLOUR PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The precision for operands of datatype colour component (CCO) is specified for subsequent data of type CCO. The precision is defined as the field width measured in units applicable to the specific encoding.

References:

6.3

7.3.8 COLOUR INDEX PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The precision for operands of data type colour index (CI) is specified for subsequent data of type CI. The precision is defined as the field width measured in units applicable to the specific encoding.

References:

6.3

7.3.9 MAXIMUM COLOUR INDEX

Parameters:

maximum colour index (CI)

Description:

The *maximum colour index* parameter represents an upper bound (not necessarily the least upper bound) on colour index values that will be encountered in the metafile.

References:

6.3

7.3.10 COLOUR VALUE EXTENT

Parameters:

colour value mapping specifier
    if the colour model is RGB or CMYK,
    minimum colour value (CD)
    maximum colour value (CD)

    if the colour model is CIELAB, CIELUV, or RGB-related
    3 pairs of colour scale and colour offset (6R)

Description:

For colour models RGB and CMYK, the parameters represent an extent which normally bounds the direct colour values that will be encountered in the metafile. It need not represent the exact extent of colour values contained in the metafile. The *minimum colour value* and *maximum colour value* parameters are 3-tuples or 4-tuples giving the colour components corresponding to the normalized colour space, zero to one for each component. The values given will depend upon the colour model RGB or CMYK selected for use in the metafile.

For colour models CIELAB, CIELUV, and RGB-related the parameters represent the scale and offset that relate each component of a direct colour value to the colour value of the corresponding colour space. The three pairs apply to the first, second and third component of the colour direct value.

If the minima and maxima of the COLOUR VALUE EXTENT cannot be represented in the precision declared by an element's *local colour precision* parameter (for CELL ARRAY and PATTERN TABLE), or *cell colour precision* parameter (for TILE), then for the duration of that element the effective colour value extent for the components shall correspond to the range: $0..2^{lcp} - 1$ for the Binary Encoding, where *lcp* is the local colour precision measured in bits; and, 0..*lcp_max* for the Clear Text Encoding, where *lcp_max* is the effective value which results from interpretation of the *local colour precision* parameter.

NOTE 1 Only RGB is allowed in Version 1 and Version 2 metafiles.

NOTE 2 For example, if CD_i, i = 1,2,3, denotes the ith component of the colour direct value, then the corresponding colour value in CIELAB or CIELUV colour space is obtained through: $scale_i \cdot CD_i + offset_i$, where $scale_i$ and $offset_i$, i = 1,2,3, denote the colour scale and colour offset for the ith colour component. The motivation for this is representational rather than colorimetric. The range of values for L^*, a^* and b^*, for example, is typically $0 \leq L^* \leq 100.0$, $-128 \leq a^* \leq 127$, $-128 \leq b^* \leq 127$.

References:

6.7.6

7.3.11 METAFILE ELEMENT LIST

Parameters:

The form of the parameter is encoding dependent.

Description:

All of the elements that may be encountered in the metafile and that are not mandatory are listed. (Mandatory elements are those which shall be contained in every syntactically correct metafile.)

METAFILE ELEMENT LIST shall appear exactly once in the Metafile Descriptor of every metafile. The list represents an upper bound of functional capability. It need not be the least upper bound. Every element in the metafile shall be in the list, but the list may include elements not found in the metafile.

Shorthand names are provided for use in the metafile elements list. These names may be used in conjunction with individual element names in the element list. The shorthand names, presented according to the lowest metafile version in which they may be appear, are

Version 1	**Version 2**	**Version 3**	**Version 4**
DRAWING SET DRAWING PLUS CONTROL SET	VERSION 2 SET EXTENDED PRIMITIVES SET VERSION 2 GKSM SET	VERSION 3 SET	

The elements included in each of these sets are listed in sub-clause 6.3.2.

NOTE The information carried by this element can be used by the interpreters to determine the maximum facilities necessary for interpreting the metafile.

References:

6.3.2

7.3.12 METAFILE DEFAULTS REPLACEMENT

Parameters:

Picture Descriptor, Control, Primitive Attribute, Escape, External, and Segment elements

Description:

Each element in the element list shall have the same format, meaning, and parameter data types as it does when it occurs outside the METAFILE DEFAULTS REPLACEMENT element. Clause 8 gives default values for those CGM elements for which defaults make sense. Substitute or replacement values for the defaults may be defined with the METAFILE DEFAULTS REPLACEMENT. Any subset of the picture descriptor, control, primitive attribute, and segment elements which are given defaults in clause 6 may be included. The default values set by the Metafile Defaults Replacement, or the defaults defined in clause 8, where explicit values are not included in the Metafile Defaults Replacement, are resumed at each BEGIN PICTURE.

The parameters in the defaults replacement list are order dependent. When an element is encountered in the defaults replacement list, the value replaces the current default value for the element. If an element occurs more than once in the defaults replacement list, then the last value specified is the default value used by BEGIN PICTURE.

The content and format for elements in the default list are the same as the content and format for setting corresponding elements. The format of the parameter list is not further elaborated here in order to allow freedom for encodings to treat this complex element in the manner best suited to the encodings.

When a value has more than one specification mode, this standard defines its default for each mode. An element that sets a default value in the defaults replacement list shall set the value in the current specification mode. The current specification mode when processing the list is either the default mode defined by ISO/IEC 8632, or the mode most recently set by an element in the list. The list may contain elements that set values in more than one mode.

Elements in the list are processed sequentially. If a value is defined more than once in the list, the default that actually takes effect is the one set latest in the list.

NOTE Segment elements may appear in the METAFILE DEFAULTS REPLACEMENT only for Version 2, Version 3, and Version 4 metafiles.

References:

6.3.3

7.3.13 FONT LIST

Parameters:

font names (nSF)

Description:

This element permits selection of named fonts via TEXT FONT INDEX. The first font defined in the font list is assigned to index 1. The second to index 2, etc. The syntax and meaning of the strings comprising the font names are not specified by this International Standard.

NOTE For maximum portability of metafiles, it is recommended that the naming procedures defined by ISO/IEC 9541 be used. There currently is no central font registration authority within ISO which operates according to the conventions of ISO/IEC 9541.

References:

6.3.4
6.7.3.2

7.3.14 CHARACTER SET LIST

Parameters:

list of:

character set type (one of:
94-character G-set,
96-character G-set,
94-character multibyte G-set,
96-character multibyte G-set,
complete code) (E),

character set designation (SF)

Description:

The CHARACTER SET LIST element declares the character sets that may be named in subsequent CHARACTER SET INDEX and ALTERNATE CHARACTER SET INDEX elements and establishes the character set index value that is associated with each of these character sets.

The first character set declaration in the list names the character set whose character set index value is to be 1. Likewise, the second, third, fourth, etc., character set declarations name the character sets whose index values are to be 2, 3, 4, etc.

Each character set declaration has two parts: the *character set type* parameter and the *character set designation* parameter. The type specifies which type of character set is being declared (that is, which type of IS0 2022 designating escape sequence is associated with that character set). The character set designation consists of, optionally the character set revision sequence, and the designation sequence tail. The character set revision escape sequence, if present, is indicated by <ESC> 2/6 F, where F specifies the revision number 1 to 63 by taking values 4/0 to 7/14 respectively. The designation sequence tail consists of the character or characters that forms the “tail end” of such designating escape sequences for that character set.

NOTE Information regarding the *designation sequence tail* parameter can be found in the International Register of Coded Character Sets to be Used with Escape Sequences. This register is maintained by the Registration Authority for ISO 2375, which is the European Computer Manufacturers Association (ECMA), Rue du Rhone 114, CH-1204, Geneva, Switzerland.

References:

6.3.4
6.7.3.2

7.3.15 CHARACTER CODING ANNOUNCER

Parameters:

coding technique (E)

Description:

This element informs the metafile interpreter of the code extension capabilities assumed by the metafile generator.

‘Coding technique' identifies the code extension technique and environment assumed by the generator of the metafile. These code extension capabilities apply only to the string parameters of TEXT, APPEND TEXT, RESTRICTED TEXT and possible GENERALIZED DRAWING PRIMITIVE (GDP) elements.

Whether ‘coding technique' applies to string parameters within the data record of a given GDP depends upon the definition of the particular GDP. The defined values and their meanings are

BASIC 7-BIT

Character sets are switched by using CHARACTER SET INDEX, which designates a set into G0.

If ALTERNATE CHARACTER SET INDEX appears in the METAFILE ELEMENT LIST, it signals that the G1 set may be accessed using SI/SO as described in ISO 2022.

BASIC 8-BIT

Character sets are switched by using CHARACTER SET INDEX and ALTERNATE CHARACTER SET INDEX.

The G1 set may be accessed by characters from columns 10 to 15 of an 8-bit code chart. No locking or single shifts are used within the text string. .

EXTENDED 7-BIT

Sets G0, G1, G2, and G3 may be invoked using the 7-bit encoding of any of the locking shifts or single shifts, in conformance with ISO 2022. CHARACTER SET INDEX selects G0 and ALTERNATE CHARACTER SET INDEX selects both G1 and G2. Designation of G2 and G3 may be done within text strings, in conformance with ISO 2022. (Designation of G0 and G1 may not be done in this fashion.)

EXTENDED 8-BIT

Sets G0, G1, G2, and G3 may be invoked using the 8-bit encoding of any of the locking shifts or single shifts, in conformance with ISO 2022. CHARACTER SET INDEX selects G0 and ALTERNATE CHARACTER SET INDEX selects both G1 and G2. Designation of G2 and G3 may be done within text strings, in conformance with ISO 2022. (Designation of G0 and G1 may not be done in this fashion.) .

NOTE This element corresponds to the "announcer" sequences of ISO 2022.

References:

6.7.3.2

7.3.16 NAME PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The precision for operands of data type name (N) is specified for subsequent data of type N. The precision is defined as the field width measured in units applicable to the specific encoding.

References:

6.3

7.3.17 MAXIMUM VDC EXTENT

Parameters:

first corner (P)
second corner (P)

Description:

The two corners define a rectangular extent in VDC space which bounds the values of the VDC EXTENT elements which may be found in the metafile. It may be, but need not be, a closest bound in the sense that it exactly equals the union of the extent rectangles in the metafile.

References:

6.3
6.4.4

7.3.18 SEGMENT PRIORITY EXTENT

Parameters:

minimum priority extent (I)
maximum priority extent (I)

Description:

The parameters represent an extent which bounds the segment display and pick priority values in the metafile. It need not be the minimal bounding extent. Both parameters shall be non-negative, and *minimum display priority* shall be less than *maximum display priority.*

References:

6.3
6.10.4

7.3.19 COLOUR MODEL

Parameters:

colour model indicator (IX)

Description:

The colour model of the metafile is selected. The following values are defined:

1: RGB
2: CIELAB
3: CIELUV
4: CMYK
5: RGB-related

Values greater than 5 are reserved for registration and future standardization.

All occurrences of colour-setting elements, representation setting elements, colour lists, and any other place where a direct colour value may appear shall be in the selected colour model, whether it is explicitly selected by this element or defaulted.

In Version 1 and Version 2 metafiles only one colour model, RGB, shall be used within a metafile.

NOTE Colour models are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a colour model has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the colour model value will be assigned by the Registration Authority.

References:

6.7.6

7.3.20 COLOUR CALIBRATION

Parameters:

calibration selection (IX)
reference white value (X_n,Y_n, Z_n) (3R)
RGB and RGB-related calibration data
  matrix1 (3x3) (9R)
  matrix2 (3x3) (9R)
  n (I)
  array of pairs (R,R') of red components (n(2CCO))
  array of pairs (G,G') of green components (n(2CCO))
  array of pairs (B,B') of blue components (n(2CCO))

CMYK calibration data
  m (I)
  array of grid locations (CMYK) (mCD)
  array of grid values (XYZ) (m(3R))

Description:

Colour calibration supplies the information which defines the transformation from the colour space selected by the COLOUR MODEL element to the CIEXYZ reference colour space.

The value of the *reference white value* parameter specifies the CIEXYZ values (X_n,Y_n,Z_n) of the reference white. The reference white value is the only colour calibration applicable to the CIELAB and CIELUV colour space for conversion to the CIEXYZ reference colour space.

For colour model RGB, the calibration data parameter specifies the values used to position the Red, Green, and Blue colour components in the CIEXYZ reference colour space. The *matrix1* parameter consists of the tristimulus values of the RGB primaries

$$\begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix}$$

The three arrays of n pairs of CCOs define a *colour look-up table* (LUT) in order to transform non-linear RGB to another RGB specification that is linear with respect to luminous intensity and suitable to be transformed with the 3x3 matrix1 into the CIEXYZ reference colour space.

R' = R_LUT[R]
G' = G_LUT[G]
B' = B_LUT[B]

The same entries R, B, G shall be defined in each array. The number of entries in the R, G, and B tables shall be the same and shall be defined in the same order.

When colour model indicator is CMYK, additionally to the white reference value, the calibration data parameter specifies a table of CIEXYZ values for the colours resulting from a grid of specific combinations of C, M, Y, and K colour components. The grid values XYZ in the CIEXYZ reference colour space are specified for each of the m grid locations CMYK.

NOTE 1 It is recommended that the minimum number is m=3.

When the colour model is RGB-related, the *matrix2* parameter transforms an RGB-related colour space with colour components A, B, and C into (possibly non-linear) RGB values, which in turn are transformed into the CIEXYZ reference colour space as described above. Matrix2 consists of transformation matrix elements:

$$\begin{bmatrix} R_a & R_b & R_c \\ G_a & G_b & G_c \\ B_a & B_b & B_c \end{bmatrix}$$

The ISO*calibration selection* parameter indicates which of the calibration data are applied for a selected colour model. Following are the defined values:

1 unspecified;
2 reference white only;
3 reference white, matrix1;
4 reference white, matrix1, look-up table (LUT);
5 reference white, matrix1, look-up table (LUT), matrix2;
6 reference white, matrix1, matrix2;
7 lookup tables, matrix2;
8 matrix2;
9 reference white, CMYK calibration data (grid locations and grid values).

Legal values are positive integers. Values greater than 9 are reserved for registration and future standardization.

Legal values of the ISO*calibration selection* parameter for each colour model are shown in table 12, together with the applicable calibration data parameters for each colour model.

NOTE 2 Some of the calibration selection values merely allow transformation between non-calibrated colour spaces, such as values 7 and 8.

Table 12 — Applicable calibration data and legal calibration selection values

	Colour Models				
	RGB	**CIELAB**	**CIELUV**	**CMYK**	**RGB-related**
reference white	X	X	X	X	X
matrix1	X				X
matrix2					X
LUT	X				X
grid locations &grid values				X	
legal *calibration selection* values	1,2,3,4	1,2	1,2	1,2,9	1,2,5,6, 7,8

An "X" in the table indicates that the particular piece of calibration data is applicable in the given colour model.

NOTE 3 ISO/IEC 8613 mandates value 9 (and only value 9) for legal calibration selection value.

NOTE 4 Calibration selection values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a calibration selection specification has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the calibration selection value will be assigned by the Registration Authority.

References:

6.7.6

7.3.21 FONT PROPERTIES

Parameters:

list of (property indicator, priority, property value record) (n[IX, I, SDR]])

where standardized values of the property indicator (and associated data type) include:

1: fot index (IX)
2: standard version (I)
3: design source (SF)
4: font family (SF)
5: posture (IX)
6: weight (IX)
7: proportionate width (IX)
8: included glyph collections (nIX)
9: included glyphs (mUI32)
10: design size (R)
11: minimum size (R)
12: maximum size (R)
13: design group (3UI8)
14: structure (IX)

Description:

Font properties are described which may be used for substituting another font for the referenced font designated by property 1, *font index*. The value of font index corresponds to a font name which has been defined by the FONT LIST element. The FONT LIST entry corresponding to a font index value shall have been defined before the occurrence of a reference to it by this *font index* property.

The *priority* parameter indicates the relative importance of the property for font substitution. The sum of all priorities is normalized to 1.0 and the relative priorities are computed as a fraction of 1.0. If, for example, no substitution is permissible, then the font index could be given priority 10 and all other properties priority 0. If, on the other hand, all that is needed is a bold serif font, then weight and design group could be given priority 10 and all others priority 0.

The properties which may be referenced by the property indicator are from ISO/IEC 9541. All of the assigned values are from the Minimum Font Description Subset of ISO/IEC 9541-2. Note that the font name itself (referenced by font index), which subsumes all other properties, is one of the properties. Values of property indicator greater than 14 are reserved for registration and future standardization.

The priorities given to the font properties provide guidance to the interpreter to enable rational font matching in the event of the inability to exactly match a font from the font name specified in the FONT LIST element. The priorities do not imply any particular font matching strategy, but do provide the means for generators to indicate relative importance of the various font properties.

The definitions of each of the standardized properties are given below. Following the name of the property, the structure of its SDR in the parameter list is given. The SDR for each of the standardized properties contains only one member (typed sequence), and none of the types is SDR (hence there is no nesting of data records).

NOTE 1 It is possible that additional properties will be standardized or registered in the future which have more complicated, and possibly even nested, SDRs.

Each SDR member definition contains 3 components data type indicator, data count, data list as defined in annex C. The data type indicator is identified in the following by i_XX, where XX is the data type abbreviation for the required data type. This denotes the integer value assigned to this data type in annex C (e.g., i_CI denotes the SDR data type indicator for data type "Colour Index", which is assigned the value 2 in annex C).

Font index. — [i_IX, 1, font index value] — Selects a font resource name which resides in the Metafile FONT LIST.

Standard version. — [i_I, 1, version value] — Gives the version number of ISO/IEC 9541 which is assumed by the writer of the metafile and the formulator of this font reference. Legal values are: 1, corresponding to ISO/IEC 9541 version 1991 (first version).

NOTE 2 ISO/IEC 9541 stipulates that its version number will be the year of publication, e.g., the integer 1991 if the first version is published in the year 1991. Because CGM maps ISO/IEC 9541 font information into CGM syntax, the ability to use ISO/IEC 9541 versions beyond the first (1991) may require amendment to CGM.

Design source. — [i_SF, 1, design source value] — The organizational name of the typeface design source, as specified in ISO/IEC 9541.

Font family. — [i_SF, 1, font family value] — The name of the font family, for example ISOCourier.

Posture. — [i_IX, 1, posture value] — The posture of a font; assigned values are:

0: not applicable;
1: upright;
2: oblique upright design slanted in the direction of the nominal escapement with no design or form change;
3: back slanted oblique upright design slanted in the direction opposite of the nominal escapement with no design or form change;
4: italic slanted in the direction of the nominal escapement with a change in design or form;
5: back slanted italic italic design slanted in the direction opposite of the nominal escapement;
6: other.

Posture values greater than 6 are reserved for registration and future standardization.

Weight. — [i_IX, 1, weight value] — The font weight is a measure of the boldness of the font. Assigned values are:

0: not applicable;
1: ultra light (lowest ratio of glyph stem width to font height);
2: extra light
3: light
4: semi light
5: medium
6: semi bold
7: bold
8: extra bold
9: ultra bold (highest ratio of glyph stem width to font height);

Weights are ordered according to increasing weight. Weight values greater than 9 are reserved for registration and future standardization.

Proportionate width. — [i_IX, 1, proportionate width value] — The proportionate width is an indication of the relative ratio of character height to character width. Assigned values are:

0: not applicable;
1: ultra condensed (lowest ratio of glyph width to font height);
2: extra condensed;
3: condensed;
4: semi condensed;
5: medium;
6: semi expanded;
7: expanded;
8: extra expanded;
9: ultra expanded (highest ratio of glyph width to font height);

Proportionate widths are ordered according to increasing width. Proportionate width values greater than 9 are reserved for registration and future standardization.

Included glyph collections. — [i_IX, n, n glyph collection values] — A list of CHARACTER SET INDEX values. CGM separates character set (glyph collection) from font. A mechanism exists (GLYPH MAPPING) for associating individual glyph identifiers (for glyphs registered under ISO/IEC 10036) with a metafile character set index. This mechanism may be used in conjunction with this element to specify the included glyph collections that the font selected to satisfy this font reference should have (according to the associated priority).

Included glyphs. — [i_UI32, m, m AFII glyph identifiers] — A list of 32-bit AFII glyph identifiers registered under ISO/IEC 10036. The selected font should have these glyph collections available according to the associated priority.

Design size. — [i_R, 1, design size value] — The recommended optimal body size, measured in millimetres, at which the font resource is designed to be used.

Minimum size. — [i_R, 1, minimum size value] — The recommended minimum body size, measured in millimetres, defining the lower limit of the range over which the font resource is designed to be used.

Maximum size. — [i_R, 1, maximum size value] — The recommended maximum body size, measured in millimetres, defining the upper limit of the range over which the font resource is designed to be used.

Design group. — [i_UI8, 3, 3 design group values] — The design grouping of the typeface of the font resource consists of three components: the typeface ISO*class*, the typeface *subclass*, and the typeface ISO*specific group*, as defined in ISO/IEC 9541-1. The typeface general class is the most general grouping of fonts with similar characteristics. Typeface sub-classes are groupings that identify the less general characteristics and start to categorize typefaces into similar designs. Typeface specific groups are typeface groupings with very distinct and unique characteristics. Typefaces categorized to the typeface class level start to show similar characteristics that make them reasonably eligible to be substituted for each other. The assigned design groups, and their properties, are defined by the normative annex A of ISO/IEC 9541-1. The three components are each assigned a value in the range 0..255. In annex A of ISO/IEC 9541-1 a typeface design group specification looks like x.y.z, with each of x, y, and z in the range 0..255.

NOTE 3 The properties weight, proportionate width, posture, structure, specify further typographic variations on the design group.

Structure. — [i_IX, 1, structure value] — Structure indicates the structure of strokes of the glyph shapes of the font resource. Assigned values are:

0: undefined or not applicable;
1: solid the shape contains no voids or patterns within the strokes;
2: outline the shape includes only the outer edges of the strokes.

Structure values greater than 2 are reserved for registration and future standardization.

NOTE 4 Property indicator values, posture values, weight values, proportionate width values, and structure values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a property indicator, posture, weight, proportionate width, or structure has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the appropriate parameter values will be assigned by the Registration Authority.

References:

6.3.4.1

7.3.22 GLYPH MAPPING

Parameters:

character set index (IX)
30 basis character set type (one of : 94-character G-set,
96-character G-set,
94-character multibyte G-set,

96-character multibyte G-set,
complete code) (E),
basis set designation sequence tail (SF)
octets per code (I)
glyph source identifier (IX)
glyph-code association (SDR)

Description:

A character set is defined for use in the metafile. The first four parameters define the defaults for undefined positions in the set, the structure of the character codes (number of bytes), and the index by which it will be referenced. The remaining parameters define the source and association of the glyphs which are being mapped to character codes.

The character set index can be used in the CHARACTER SET INDEX and ALTERNATE CHARACTER SET INDEX elements. An index used in this element cannot also be declared in a CHARACTER SET LIST element. Each code in the defined character set will contain the number of octets indicated in the *octets per code* parameter. The basis set is selected by the *character set type* and *designation sequence tail* parameters, as described under the CHARACTER SET LIST element (see 6.3.4.1). The effect of several GLYPH MAPPING elements with the same character set index is not cumulative. The basis set for the GLYPH MAPPING each time is that specified by the *character set type* and *designation sequence tail* parameters. It is not the result of previous GLYPH MAPPING elements. The basis set provides a default set of glyphs to use with any codes that are not assigned values by this element. The string that specifies the basis set is a designation sequence tail as defined for the CHARACTER SET LIST element (see 6.3.4.1).

The parameter *glyph source* identifies the source of the extended glyph sets, and will determine the nature of the association which is established by the *glyph-code association* parameter. The following source is defined:

1: AFII registry, glyph identifiers are AFII 4-byte identifiers.
Values greater than 1 are reserved for registration and future standardization.

For the value 1 of the *glyph source identifier* parameter, the ISO*glyph-code association* establishes a mapping based on pairs of codes and glyph names. The glyph names are AFII 4-byte identifiers. The structured data record contains 2 members, each consisting of a list: one list of character codes and one list of AFII identifiers:

[(i_UI8, n*m, n(m-byte code)) (i_UI32, n, n(AFII 4-byte identifier))]

where the number of octets that represent each code (mUI8) is equal to the value of the ISOoctets per code parameter.

Each item in the list associates a code with a glyph.

Each glyph name is an integer identifier in the range 1..(2^{32} - 1) which is registered by the ISO Glyph Registration Authority, AFII.

NOTE 1 Each encoding of this part of this International Standard provides a means to more efficiently represent sequences of pairs which have a uniform increment of 1 in the values of both components of successive pairs in the sequence.

NOTE 2 Glyph source identifiers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a glyph source has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the glyph source identifier will be assigned by the Registration Authority.

References:

6.3.4.4

7.3.23 SYMBOL LIBRARY LIST

Parameters:

symbol library names (nSF)

Description:

This element permits selection of named symbol libraries via SYMBOL LIBRARY INDEX. The first symbol library defined in the symbol library list is assigned to index 1, the second to index 2, and so on.

NOTE 1 The strings may contain registered names or private names. Use of the former is recommended for metafile transportability, because registration ensures unique naming of symbol libraries.

NOTE 2 Symbol Libraries are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a symbol library has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the symbol library name will be assigned by the Registration Authority.

References:

6.6.12

7.3.24 PICTURE DIRECTORY

Parameters:

location data type selector (one of: UI8, UI16, UI32) (E)

list of: three-tuples of picture identifier (SF), picture location (ldt), application structure directory location (ldt)

NOTE ldt is the location data type specified by the location data type selector.

Description:

This element provides an optional directory that is intended to assist the metafile interpreter in accessing a picture contained within the metafile. The two parameters for this element comprise a location parameter data type and a list of picture identifier, picture location, application structure directory location three-tuples. The values of the picture identifier parameter shall correspond exactly to the values of the picture identifiers in the BEGIN PICTURE elements of the metafile. The values of the picture location parameter are encoding-dependent representations of the number of increments that separate the start of the metafile from the start of the BEGIN PICTURE element of the associated picture. The values of the application structure directory location parameter are encoding-dependent representations of the number of increments that separate the start of the metafile from the start of the APPLICATION STRUCTURE DIRECTORY element of the associated picture. Even if an application structure directory location parameter is 0, an application structure directory may still be present. The location of the start of the metafile is 0.

References:

4.3.2.7
4.3.2.8
4.3.5

7.4 Picture descriptor elements

7.4.1 SCALING MODE

Parameters:

scaling mode (one of: abstract, metric) (E)
metric scale factor (R)

Description:

The *scaling mode* parameter defines the meaning of the VDC. If set to ‘abstract', the VDC space is dimensionless and the picture is correctly displayed at any size: the metric scale factor parameter is ignored. If set to ‘metric', the VDC space has implied measure: the metric scale factor represents the distance (in millimetres) in the displayed picture corresponding to one VDC unit. One VDC unit represents one millimetre multiplied by the metric scale factor. In this case the picture is correctly displayed at the indicated size only. If both device viewport and scaling mode appear in the same metafile, the last specified shall be used. If neither appears, the default values for device viewport shall take precedence.

NOTE The *metric scale factor* parameter is present in the element even when the scaling mode is ‘abstract'. In this case it is ignored by the interpreter.

References:

6.4.1

7.4.2 COLOUR SELECTION MODE

Parameters:

colour selection mode (one of: indexed, direct) (E)

Description:

Two methods of colour selection are supported: by colour table entries (‘indexed') or by direct colour values (‘direct').

In a Version 1 metafile a single colour selection mode applies to the entire picture body, because this element, if present, shall not appear in the picture body. The mode may be defaulted or explicitly set with the COLOUR SELECTION MODE element. In Version 2, Version 3 and Version 4 metafiles, this element may appear in the picture body as well as the picture descriptor, and the colour selection mode may therefore be changed within the picture.

All occurrences of colour-setting elements (AUXILIARY COLOUR, TRANSPARENT CELL COLOUR, LINE COLOUR, MARKER COLOUR, FILL COLOUR, EDGE COLOUR, TEXT COLOUR, and SYMBOL COLOUR) as well as the colour lists of CELL ARRAY, TILE, and PATTERN TABLE, and the background and foreground colour definitions of BITONAL TILE, shall be in the current mode.

References:

6.4.2
6.7.6

7.4.3 LINE WIDTH SPECIFICATION MODE

Parameters:

line width specification mode (one of: absolute, scaled, fractional, mm) (E)

Description:

One of four methods of specifying geometric aspects associated with lines and the transformation behaviour of those aspects is selected. See clause 6 for a description of the meanings of the four styles. The modes ‘fractional' and ‘mm' are only supported in Version 3 and Version 4 metafiles.

In a Version 1 metafile a single mode applies to the entire picture body, because this element, if present, shall not appear in the picture body. The mode may be defaulted or explicitly set with this element. If used in a Version 1 metafile, this element shall be in the picture descriptor, after BEGIN PICTURE and before BEGIN PICTURE BODY. In Version 2, Version 3 and Version 4 metafiles, this element may appear in the picture body as well as the picture descriptor, and the colour selection mode may therefore be changed within the picture.

References:

6.4.3
6.7.5

7.4.4 MARKER SIZE SPECIFICATION MODE

Parameters:

marker size specification mode (one of: absolute, scaled, fractional, mm) (E)

Description:

One of four methods of specifying geometric aspects associated with markers and the transformation behaviour of those aspects is selected. See clause 6 for a description of the meanings of the four styles. The modes ‘fractional' and ‘mm' are only supported in Version 3 and Version 4 metafiles.

In a Version 1 metafile a single mode applies to the entire picture body, because this element, if present, shall not appear in the picture body. The mode may be defaulted or explicitly set with this element. If used in a Version 1 metafile, this element shall be in the picture descriptor, after BEGIN PICTURE and before BEGIN PICTURE BODY. In Version 2, Version 3 and Version 4 metafiles, this element may appear in the picture body as well as the picture descriptor, and the mode may therefore be changed within the picture.

References:

6.4.3
6.7.5

7.4.5 EDGE WIDTH SPECIFICATION MODE

Parameters:

edge width specification mode (one of: absolute, scaled, fractional, mm) (E)

Description:

One of four methods of specifying geometric aspects associated with edges and the transformation behaviour of those aspects is selected. See clause 6 for a description of the meanings of the four styles. The modes ‘fractional' and ‘mm' are only supported in Version 3 and Version 4 metafiles.

In a Version 1 metafile a single mode applies to the entire picture body, because this element, if present, shall not appear in the picture body. The mode may be defaulted or explicitly set with this element. If used in a Version 1 metafile, this element shall be in the picture descriptor, after BEGIN PICTURE and before BEGIN PICTURE BODY. In Version 2, Version 3 and Version 4 metafiles, this element may appear in the picture body as well as the picture descriptor, and the colour selection mode may therefore be changed within the picture.

References:

6.4.3
6.7.5

7.4.6 VDC EXTENT

Parameters:

first corner (P)

Description:

second corner (P)

The two corners define a rectangular extent in VDC space that is the "region of interest" for the succeeding CGM elements.

The first corner represents the lower-left corner of the picture, and the second corner represents the upper-right corner of the picture as seen by the viewer of the picture. The values of the coordinates for any dimension may be either increasing or decreasing from the first to the second corner. For example, for devices with an upper-left origin, a picture may be described in coordinates that map directly to the device but still may be displayed correctly on a device with a lower-left origin.

The VDC EXTENT thus establishes the sense and orientation of VDC space (that is, the directions of the positive x (+x) and positive y (+y) axes, and whether the +y axis is 90° clockwise or 90° counterclockwise from the +x axis, see 6.4.4 and figure 1).

In particular, VDC EXTENT establishes the direction of positive and negative angles as follows: positive 90° is defined to be the right angle from the positive x-axis to the positive y-axis.

Some attributes such as text attributes (for example, the directions of the up and base component vectors of CHARACTER ORIENTATION and, therefore, the meaning of the enumerative values ‘right', ‘left', ‘up', ‘down') are intimately bound to these definitions.

Specification of values outside VDC EXTENT in parameters of CGM elements is permitted. VDC EXTENT demarcates the region of interest within the picture; the visible portion of an image should be contained within VDC EXTENT.

References:

6.4.4
6.4.5

7.4.7 BACKGROUND COLOUR

Parameters:

colour value (CD)

Description:

The colour value defines the background colour for the image whose definition begins with the next BEGIN PICTURE BODY element.

The single parameter of BACKGROUND COLOUR is always a direct colour value, regardless of the current value of COLOUR SELECTION MODE. If the current COLOUR SELECTION MODE is indexed then: the BACKGROUND COLOUR element defines the initial representation of colour index 0 for the picture as well as the image background colour; and setting a value for colour index 0 using the COLOUR TABLE element also sets the background colour, if it occurs in the Picture Descriptor or METAFILE DEFAULTS REPLACEMENT.

References:

6.4.6

7.4.8 DEVICE VIEWPORT

Parameters:

first corner (VP)

Description:

second corner (VP)

The two parameters define the opposite corners of a rectangular viewport on the device's display surface. These parameters are specified by the unit system selected by DEVICE VIEWPORT SPECIFICATION MODE.

The effective viewport is that area of the display surface onto which the VDC extent rectangle is mapped. If the current DEVICE VIEWPORT MAPPING forces isotropic mapping, and the aspect ratio is not equal to that of the device viewport, the effective viewport will be smaller than the specified viewport on one or the other axis (but not both).

If the current DEVICE VIEWPORT MAPPING does not force isotropic mapping, the effective viewport will be the same as the specified viewport. If the Device Viewport exceeds the available display surface, the Device Viewport is still used to determine the VDC-to-Device mapping.

Mirroring or 180° rotation of the image may be achieved by specifying the corners in some way other than that the first is below and to the left of the second.

If both device viewport and scaling mode appear in the same metafile, the last specified is used. If neither appears, the default values for device viewport take precedence.

References:

6.4.7

7.4.9 DEVICE VIEWPORT SPECIFICATION MODE

Parameters:

VC specifier (one of: fraction of display surface,
millimetres with scalefactor,
physical device coordinates)(E)

metric scale factor (R)

Description:

This element determines how subsequent elements using the data type VC (viewport coordinate) or VP (viewport point) will be defined.

These parameters may be specified in one of three modes: fraction of display surface; millimetres with scale factor; or physical device coordinates.

When the value of the *VC specifier* parameter is ‘fraction of display surface', the value (0.0, 0.0) corresponds to the lower left corner and the value (1.0, 1.0) corresponds to the upper right corner of the default device viewport. (The default device viewport is the largest unrotated rectangular area visible on the display surface). Numbers outside the range [0.0 to 1.0] may be specified (see 7.4.8). When the VC specifier is ‘fraction of display surface' the value of the *metric scale factor* parameter is ignored.

When the VC specifier is ‘millimetres with scalefactor', the metric scale factor represents the distance (in millimetres) on the display surface corresponding to one unit in VC space. One unit in VC space represents one millimetre multiplied by the metric scale factor. The value (0,0) corresponds to the lower left corner and the values increase positively to the right and upwards.

When the VC specifier is 'physical device coordinates', the native units and handedness of the physical device are used. The metric scale factor is ignored.

NOTE Metric scaling with a scale factor provides a device-independent means of generating output at a known size. In metric mode, a scale factor of 1.0 indicates that the VC are in units of millimetres; a scale factor of 0.0254 would imply a VC of one thousand per inch.

References:

6.4.7

7.4.10 DEVICE VIEWPORT MAPPING

Parameters:

isotropy flag (one of: not forced, forced)(E)
horizontal alignment flag (one of: left, centre, right)(E)
vertical alignment flag (one of: bottom, centre, top)(E).

Description:

This element determines how the coordinate mapping is derived from the VDC EXTENT and the specified DEVICE VIEWPORT. The remaining parameters are significant only if isotropy is forced by the first parameter. If so, the effective viewport is generally smaller than the specified viewport, and these parameters determine how it will be positioned within the specified viewport. 'Left' and 'bottom' are interpreted as being towards the "first corner" of the specified DEVICE VIEWPORT, regardless of any mirroring or rotation of the viewport on the physical device.

References:

6.4.7

7.4.11 LINE REPRESENTATION

Parameters:

line bundle index (IX)
line type (IX)
line width specifier (SS)
line colour specifier (CO)

Description:

In the line bundle table, the given line bundle index is associated with the specific parameters.

The *line type* parameter is specified and behaves as indicated in the LINE TYPE attribute element.

The *line width* parameter is defined in the current LINE WIDTH SPECIFICATION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes in the specification mode.

The *line colour* parameter is defined in the current COLOUR SELECTION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes to the selection mode.

Which aspects are used depends on the corresponding ASFs; see the ASPECT SOURCE FLAG element.

References:

6.4.8

7.4.12 MARKER REPRESENTATION

Parameters:

marker bundle index (IX)
marker type (IX)
marker size specifier (SS)
marker colour specifier (CO)

Description:

In the marker bundle table, the given marker bundle index is associated with the specified parameters.

The *marker type* parameter is specified and behaves as indicated in the MARKER TYPE attribute element.

The *marker size* parameter is defined in the current MARKER SIZE SPECIFICATION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes in the specification mode.

The *marker colour* parameter is defined in the current COLOUR SELECTION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes to the selection mode.

Which aspects are used depends on the corresponding ASFs; see the ASPECT SOURCE FLAG element.

References:

6.4.8

7.4.13 TEXT REPRESENTATION

Parameters:

text bundle index (IX)
font index (IX)
text precision (one of: string, character, stroke) (E)
character spacing (R)
character expansion factor (R)
text colour specifier (CO)

Description:

In the text bundle table, the given text bundle index is associated with the specified parameters.

The *font index* parameter is specified and behaves as indicated in the TEXT FONT INDEX attribute element.

The *text precision* parameter is specified and behaves as indicated in the TEXT PRECISION attribute element.

The *character spacing* parameter is specified and behaves as indicated in the CHARACTER SPACING attribute element.

The *character expansion factor* is specified and behaves as indicated in the CHARACTER EXPANSION FACTOR attribute element.

The *text colour* parameter is defined in the current COLOUR SELECTION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes to the selection mode.

Which aspects are used depends on the corresponding ASFs; see the ASPECT SOURCE FLAG element.

References:

6.4.8

7.4.14 FILL REPRESENTATION

Parameters:

fill bundle index (IX)
interior style (one of: hollow, solid, pattern, hatch, empty, geometric pattern, interpolated) (E)
fill colour specifier (CO)
hatch index (IX)
pattern index (IX)

Description:

In the fill bundle table, the given fill bundle index is associated with the specified parameters.

The *interior style* parameter is specified and behaves as indicated in the INTERIOR STYLE attribute element.

The *fill colour* parameter is defined in the current COLOUR SELECTION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes to the selection mode.

The *hatch index* parameter is specified and behaves as indicated in the HATCH INDEX attribute element.

The *pattern index* is specified and behaves as indicated in the PATTERN INDEX attribute element.

Which aspects are used depends on the corresponding ASFs; see the ASPECT SOURCE FLAG element. Interior styles 'geometric pattern' and 'interpolated' shall only be used in Version 3 and Version 4 metafiles.

References:

6.4.8

7.4.15 EDGE REPRESENTATION

Parameters:

edge bundle index (IX)
edge type (IX)
edge width specifier (SS)
edge colour specifier (CO)

Description:

In the edge bundle table, the given edge bundle index is associated with the specified parameters.

The *edge type* parameter is specified and behaves as indicated in the EDGE TYPE attribute element.

The *edge width* parameter is defined in the current EDGE WIDTH SPECIFICATION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes in the specification mode.

The *edge colour* parameter is defined in the current COLOUR SELECTION MODE and is stored in the bundle table along with that mode. Thus, the definition is immune to subsequent changes to the selection mode.

Which aspects are used depends on the corresponding ASFs see the ASPECT SOURCE FLAG element.

References:

6.4.8

7.4.16 INTERIOR STYLE SPECIFICATION MODE

Parameters:

interior style specification mode (one of: absolute, scaled, fractional, mm) (E)

Description:

One of four methods is selected for specifying geometric aspects associated with filled-area primitives and the transformation behaviour of those aspects. See clause 6 for a description of the meanings of the styles.

In Version 1 metafiles and Version 2 metafiles, the style is limited to the default, 'absolute.' This element is permitted only in Version 3 and Version 4 metafiles.

References:

6.6.4.7
6.7.5

7.4.17 LINE AND EDGE TYPE DEFINITION

Parameters:

line type (IX)
dash cycle repeat length (SS)
list of dash elements (nI)

Description:

This element defines a line type or edge type and associates it with an index for future reference. The *line type* is the index of line type being defined. The index shall be negative, to avoid conflict with standardized and registered values. The *list of dash elements* comprises the definition to be associated with the index. The first element is a dash, the second a space, etc. — the defined linetype is solid for l_1 units, gap for l_2 units, solid for l_3 units, and so on. There shall be at least one element in the list of dash elements. If there is only one element in the list, a solid line is drawn. Each dash element shall be non-negative. If an element is 0 for a drawn (rather than gap) element of the dash element list then a dot is drawn.

The *dash cycle repeat length* defines the length of one complete cycle of the dash pattern. The lengths of the dash elements are normalized so that the sum of the specifiers in the *list of dash elements* equals the *dash cycle repeat length*.

The units of the *dash cycle repeat length* are determined by the value of LINE WIDTH SPECIFICATION MODE. The value of 'scaled' indicates that the implementation may normalize and map the sum of the dash pattern elements at its discretion. Otherwise the units to which the dash elements are normalized are as defined by that mode.

NOTE LINE AND EDGE TYPE DEFINITION elements defining different indexes may be interleaved with LINE WIDTH SPECIFICATION MODE elements which specify different modes. Hence it is possible to associate different modes and different repeat length units with different index definitions. This could be desired, for example, if types were being defined for both lines and edges, and these two entities were to have different modes in the picture body.

References:

6.4.9
6.7.1.2
6.7.4.4

7.4.18 HATCH STYLE DEFINITION

Parameters:

hatch index (IX)
style indicator (one of: parallel, crosshatch) (E)
hatch direction vectors specifier (4SS)
duty cycle length (SS)
number of hatch lines (I)
list of gap widths (nI)
list of line types (nIX)

Description:

This element defines a hatch style and associates it with an index for future reference.

The *hatch index* parameter defines the index of the hatch style by which the hatch style is subsequently referenced. The index shall be negative, to avoid conflict with standardized and registered values.

The *number of hatch lines* defines the number of entries in the arrays of gap widths and line types. Valid values are positive integers.

The *list of gap widths* defines the gaps between the centres of the lines comprising the hatch. If *n* is the value of the *number of hatch lines* parameter, then the list of gap widths shall contain exactly *n* widths. Each gap specification defines the gap following the associated line that is, the first gap follows the first drawn hatch line.

The colour of each hatch line is the current fill colour.

The *list of line types* defines the line type of each line comprising the hatch. If *n* is the value of the *number of hatch lines* parameter, then the list of line types shall contain exactly *n* types.

The centre of the first hatch line is aligned with the FILL REFERENCE POINT.

The *duty cycle length* is measured perpendicular to the hatch lines. The hatch line gap widths are normalized so that the sum of the specifiers in the *list of gap widths* equals the *duty cycle length*.

The line width of the hatch lines is defined to be equal to the duty cycle length divided by the sum of the gap widths. That is, it is one gap width unit.

The units of the *duty cycle length* parameter are determined by the value of the INTERIOR STYLE SPECIFICATION MODE. The value of ‘scaled' indicates that the implementation may normalize and map the sum of the gap widths at its discretion. Otherwise the units to which the gap widths are normalized are defined by that mode.

The *hatch direction vectors specifier* parameter consists of two vectors which specify the directions of the hatch lines. The specification units are determined by the current INTERIOR STYLE SPECIFICATION

MODE, as are the transformation properties of the hatch style. See clause 6. If the hatch type is 'parallel', both vectors are present but only the first vector is significant. All hatch lines in the first direction are drawn first, followed by all lines in the second direction (if the style is a crosshatch).

References:

6.7.4.3

7.4.19 GEOMETRIC PATTERN DEFINITION

Parameters:

geometric pattern index (IX)
segment identifier (N)
pattern extent (2P)

Description:

This element defines a geometric pattern and associates it with an index in the geometric pattern table for future reference. The *geometric pattern index* parameter defines the index by which the geometric pattern is subsequently referenced with the PATTERN INDEX element. Legal values for the geometric pattern index are positive integers.

The *segment identifier* parameter identifies an existing, defined segment whose primitives are used to define the geometric pattern.

The *pattern extent* is specified by two points. The first point and second point define two corners of a rectangular extent. The defined pattern extent rectangle is mapped to the pattern box parallelogram as described under the PATTERN SIZE element when a filled area element is displayed with a geometric pattern interior. Valid values for the two points are any two distinct points.

The resulting rectangle is mapped onto the pattern box as described under PATTERN SIZE, and the geometric pattern fills the interior as described under that element.

When interior style is ‘geometric pattern' the interior of a filled-area is filled with the geometric pattern specified by the pattern index and this element. The primitives of the identified segment are clipped to the rectangle defined by the pattern extent. CLIP INHERITANCE and INHERITANCE FILTER have no effect. The attributes which apply to the primitives defining the pattern are those in effect when the segment was defined, as modified by occurrences of attribute elements within the segment definition itself.

All segment attributes are ignored, if any are present in the definition of the segment.

References:

6.7.4.3
D.4.3.2

7.4.20 APPLICATION STRUCTURE DIRECTORY

Parameters:

location data type selector (one of: UI8, UI16, UI32) (E)
list of: pairs of application structure identifier (SF), application structure location (ldt)

NOTE ldt is the location data type specified by the location data type selector.

Description:

This element provides an optional directory that is intended to assist the metafile interpreter in accessing the application structures contained within the pictures of the metafile. The two parameters for this element comprise a location parameter data type and a list of application structure identifier and application structure location pairs. The values of the application structure identifier parameter shall correspond exactly to the values of the application structure identifiers in the BEGIN APPLICATION STRUCTURE elements of the picture. The values of the application structure location parameter are encoding-dependent representations of the number of increments that separate the start of the picture from the start of the BEGIN APPLICATION STRUCTURE elements in the picture. The location of the start of the picture is 0.

References:

6.3.2.7
6.3.5
6.4.10

7.5 Control elements

7.5.1 VDC INTEGER PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The indicated precision for operands of data type point (P) and operands of data type VDC value (VDC) is specified for subsequent data of type P and of type VDC when VDC type is 'integer'. The precision is defined as the field width measured in units applicable to the specific encoding.

NOTE This element enables metafiles to change the form of parameters of types P and VDC in other metafile elements in the middle of a picture so that more efficient storage of data can be used when less precision is needed.

References:

6.5.1

7.5.2 VDC REAL PRECISION

Parameters:

The form of the parameter depends on the specific encoding.

Description:

The indicated precision for operands of data type point (P) and operands of data type VDC value (VDC) is specified for subsequent data of type P and of type VDC. The precision is defined as the field width measured in units applicable to the specific encoding. The precision may consist of parameters that define subfields of data type P and VDC when VDC type is 'real'.

NOTE This element enables metafiles to change the form of parameters of types P and VDC in other metafile elements in the middle of a picture so that more efficient storage of data can be used when less precision is needed.

References:

6.5.1

7.5.3 AUXILIARY COLOUR

Parameters:

auxiliary colour specifier (CO)

Description:

The auxiliary colour index or value is set as specified by the parameter.

The auxiliary colour is applied to drawing of primitives as described under the TRANSPARENCY element, when TRANSPARENCY is ‘off'.

References:

D.4.4

7.5.4 TRANSPARENCY

Parameters:

transparency indicator (one of: off, on) (E)

Description:

The value of the *transparency indicator* parameter controls the application of AUXILIARY COLOUR to the drawing of subsequent primitives.

When the value is ‘off', the following primitives are affected as described:

a) line elements: When LINE TYPE is non-solid, the dashes and dots are drawn in the current LINE COLOUR as usual, and the spaces between are drawn in the AUXILIARY COLOUR.

b) POLYMARKER: For devices that display markers within raster cells, pixels that are not part of the marker definition are displayed in the AUXILIARY COLOUR.

c) text elements: for devices that display TEXT within raster cells, pixels within the character cell that are not part of the character definition are displayed in the AUXILIARY COLOUR.

d) filled-area elements: when INTERIOR STYLE is ‘hatch' or ‘geometric pattern', pixels in the interior of the filled-area element that are either not on a hatch line or not covered by a drawn part of the geometric pattern are displayed in the AUXILIARY COLOUR; when EDGE TYPE is non-solid, the dashes and dots are drawn in the current EDGE COLOUR as usual, and the spaces between are drawn in the AUXILIARY COLOUR.

When the value is ‘on', the portions of the above primitives that would be drawn in AUXILIARY COLOUR when the value is ‘off' are rendered transparently, i.e., nothing is drawn in that portion of the primitive when the primitive is drawn.

Interpretation of this element is implementation dependent. Some recommendations are provided in annex D.

References:

None

7.5.5 CLIP RECTANGLE

Parameters:

first corner (P)
second corner (P)

Description:

The two corner points define the clip rectangle in VDC space.

When CLIP INDICATOR is ‘on', only the portions of graphics elements inside or on the boundary of the clip rectangle are drawn.

References:

6.5.2
D.4.4

7.5.6 CLIP INDICATOR

Parameters:

clip indicator (one of: off, on) (E)

Description:

When the value of the *clip indicator* parameter is ‘off', clipping of graphical primitive elements is not required.

When the value is ‘on', only the portions of graphics elements inside or on the boundary of the clip rectangle are drawn.

NOTE It is interpreter dependent whether or not clipping is done to some limit such as VDC EXTENT or display surface boundaries even when the clip indicator is ‘off'. Such action is not precluded by ISO/IEC 8632, and may be handled by the interpreter in accord with the particular needs of the implementation and driven device(s).

References:

6.5.2

7.5.7 LINE CLIPPING MODE

Parameters:

mode (one of: locus, shape, locus then shape) (E)

Description:

The Line Clipping Mode is set to the value specified.

References:

6.5.2
6.6.1.6
D.4.4

7.5.8 MARKER CLIPPING MODE

Parameters:

mode (one of: locus, shape, locus then shape) (E)

Description:

The Marker Clipping Mode is set to the value specified.

References:

6.5.2
6.6.2.3
D.4.4

7.5.9 EDGE CLIPPING MODE

Parameters:

mode (one of: locus, shape, locus then shape) (E)

Description:

The Edge Clipping Mode is set to the value specified.

References:

6.5.2
6.6.4.5
D.4.4

7.5.10 NEW REGION

Parameters:

none

Description:

This element is used for control of subregion construction within closed figures.

If the current region has not yet been closed by a preceding NEW REGION element and if the last point of the last line element is not coincident with the current closure point, then the current subregion is closed by a line segment connecting the last point of the preceding line element to the current closure point. This line becomes a part of the implicit boundary specification. If the NEW REGION was preceded by a CONNECTING EDGE element, which was itself preceded by a line primitive, then this line also becomes part of the edge specification. If the region which has been previously closed is empty, or if the last point of the last line element is coincident with the current closure point, or if the last element was a filled-area primitive then no line segment is generated by this element.

The first point of the next line element following a NEW REGION element becomes the new closure point, starting a new subregion.

References:

6.6.11

7.5.11 SAVE PRIMITIVE CONTEXT

Parameters:

context name (N)

Description:

This element allows for the grouping and identification of the set of current values of the attribute and control elements listed below as a single named entity.

Groups of elements may be saved in a picture or segment (local or global) using the *context name* parameter.

The attribute and control elements which are saved by SAVE PRIMITIVE CONTEXT and restored by RESTORE PRIMITIVE CONTEXT are:

LINE BUNDLE INDEX	INTERIOR STYLE
LINE TYPE	FILL COLOUR(1)
LINE WIDTH(1)	HATCH INDEX
LINE COLOUR(1)	PATTERN INDEX
LINE CLIPPING MODE	INTERPOLATED INTERIOR
LINE CAP	EDGE BUNDLE INDEX
LINE JOIN	EDGE TYPE
LINE TYPE CONTINUATION	EDGE WIDTH(1)
LINE TYPE INITIAL OFFSET	EDGE COLOUR(1)
MARKER BUNDLE INDEX	EDGE VISIBILITY
MARKER TYPE	EDGE CLIPPING MODE
MARKER SIZE(1)	EDGE CAP
MARKER COLOUR(1)	EDGE JOIN
MARKER CLIPPING MODE	EDGE TYPE CONTINUATION
TEXT BUNDLE INDEX	EDGE TYPE INITIAL OFFSET
TEXT FONT INDEX	FILL REFERENCE POINT
TEXT PRECISION	PATTERN SIZE
CHARACTER EXPANSION FACTOR	PICK IDENTIFIER
CHARACTER SPACING	SYMBOL LIBRARY INDEX
TEXT COLOUR(1)	SYMBOL COLOUR
CHARACTER HEIGHT	SYMBOL SIZE
CHARACTER ORIENTATION	SYMBOL ORIENTATION
TEXT PATH	CLIP INDICATOR
TEXT ALIGNMENT	CLIP RECTANGLE
CHARACTER SET INDEX	PROTECTION REGION INDICATOR(2)
ALTERNATE CHARACTER SET INDEX	AUXILIARY COLOUR(1)
TEXT SCORE TYPE	TRANSPARENCY
RESTRICTED TEXT TYPE	MITRE LIMIT
FILL BUNDLE INDEX	ASPECT SOURCE FLAGS

NOTE 1 The corresponding specification mode or selection mode in which this value was last set is also recorded. This will not cause an implicit change of mode on interpretation of RESTORE PRIMITIVE CONTEXT (see 6.5.3). Although after a RESTORE PRIMITIVE CONTEXT element primitives are drawn with the restored attribute values and specification modes or selection modes, subsequent attribute elements should be specified with the specification mode or selection mode current prior to the RESTORE PRIMITIVE CONTEXT element

NOTE 2 The currently defined regions are saved/restored as well.

References:

6.5.3

7.5.12 RESTORE PRIMITIVE CONTEXT

Parameters:

context name (N)

Description:

The attribute and control set recorded in the set specified by the *context name* parameter is recalled on interpretation. The context name shall correspond to a context name saved within the same picture by a SAVE PRIMITIVE CONTEXT element.

NOTE A primitive context can only be restored within the picture or segment within which it was saved.

References:

6.5.3

7.5.13 PROTECTION REGION INDICATOR

Parameters:

region index (IX)
region indicator (IX)

Description:

The *region indicator* parameter determines how the protection region associated with the given index is used. The legal values and their meanings are:

1: off, the region is not used;
2: clip, the region is used for clipping;
3: shield, the region is used for shielding.

It is independent of CLIP INDICATOR, which affects only the use of CLIP RECTANGLE.

References:

6.5.4

7.5.14 GENERALIZED TEXT PATH MODE

Parameters:

mode (one of: off, non-tangential, axis-tangential) (E)

Description:

The value of the *mode* parameter specifies which path the text string is to follow. If the value is ‘off' then the path specified by the TEXT PATH element (‘right', ‘left', ‘up', or ‘down') is used. If the value is ‘non-tangential' the characters are positioned along the path and oriented as specified by the character orientation vectors but the character orientation axes are not rotated — each character has the same orientation regardless of the path direction. If the value is ‘axis-tangential' the x-axis of the character orientation axes is tangent to the path at the character position — the orientation of each character depends upon the path direction at the character’s placement point. In particular, the character orientation vectors are rotated together through the angle between the tangent to the path at the placement point and the positive x direction.

This element affects the TEXT, RESTRICTED TEXT and APPEND TEXT primitives.

References:

6.5.5

7.5.15 MITRE LIMIT

Parameters:

mitre limit (R)

Description:

The *mitre limit* is defined for subsequent line elements and edges of filled areas. Mitre limit is measured as a scale factor applied to the current line or edge width. See clause 6 for a description of the effect of mitre limit.

Valid values of the *mitre limit* parameter are non-negative reals.

References:

6.5.6

7.5.16 TRANSPARENT CELL COLOUR

Parameters:

transparency indicator (one of: off, on) (E)
transparent cell colour specifier (CO)

Description:

When the value of the *transparency indicator* parameter is ‘off', then cells of CELL ARRAY, TILE, or BITONAL TILE elements are drawn normally for all values of all cell colour specifiers, and cells of PATTERN TABLE are treated normally when drawing filled-area primitives for all values of all cell colour specifiers.

When the value is ‘on', then CELL ARRAY, TILE, BITONAL TILE or PATTERN TABLE elements which contain cell colour specifiers whose value equals the *transparent cell colour specifier* are affected as described in clause 6. See clause 6 also for additional restrictions on transparent cells capability defined by this element.

References:

6.5.7

7.6 Graphical primitive elements

7.6.1 POLYLINE

Parameters:

point list (nP)

Description:

A line is drawn from the first point in the parameter list to the second point, from the second point to the next point, ..., and from the next-to-last point to the last point.

The appearance of POLYLINE is controlled by the line element attributes.

References:

6.6
6.6.1
6.7.1

7.6.2 DISJOINT POLYLINE

Parameters:

point list (nP)

Description:

A line is drawn from the starting point to the second point, from the third point to the fourth point, from the fifth point to the sixth point, ... forming a series of disjoint single line segments.

The appearance of DISJOINT POLYLINE is controlled by the line element attributes.

NOTE This element allows significant data compression for applications wishing to perform line pattern generation or vector polygon fill prior to metafile generation in a graphics system, and for other applications such as drawing grids.

References:

6.6
6.6.1
6.7.1

7.6.3 POLYMARKER

Parameters:

point list (nP)

Description:

The marker corresponding to the currently selected marker type is drawn at each of the points in the point list. If the marker type is one of the five predefined markers, it is drawn centred at each of the points. Other implementation-dependent markers may have other alignments.

If the resulting marker is completely within the clipping area, the entire marker is drawn. For Version 1 metafiles, if the marker position is within the clipping rectangle but any part of the marker is outside the clipping area, then the portion of the marker within the clipping rectangle is displayed and the display of the portion outside the rectangle is device or interpreter dependent.

For Version 2, Version 3 and Version 4 metafiles, the clipping of markers which are partially within the clipping area and partially outside of it is controlled by MARKER CLIPPING MODE as described in 6.5.2 and 6.6.2.3.

References:

6.6
6.6.2
6.7.2

7.6.4 TEXT

Parameters:

point (P)
flag (one of: not final, final) (E)
string (S)

Description:

The character codes specified in the *string* parameter are interpreted to obtain the associated glyphs from the currently selected character set. The glyphs are displayed on the view surface as specified by the text attributes.

See 6.3.4.3 regarding the legal character codes and the character set repertoire.

The characters are dimensioned according to the CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR and are oriented according to CHARACTER ORIENTATION. The direction of the character placement in the string relative to CHARACTER ORIENTATION is according to TEXT PATH.

The *flag* parameter is used to permit changing the following text attributes and control elements within a string which will be aligned as a single block:

TEXT FONT INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOUR, CHARACTER HEIGHT, CHARACTER SET INDEX, ALTERNATE CHARACTER SET INDEX, TEXT SCORE TYPE, TEXT BUNDLE INDEX, AUXILIARY COLOUR, and TRANSPARENCY.

When the value of the *flag* parameter is ‘not final', the character codes in the string parameter are accumulated, along with the current attribute settings. In this case, only the attribute setting elements listed above are allowed between this element and the APPEND TEXT element. With the exception of the ESCAPE element, no other metafile elements of any type are allowed. ESCAPE is permitted but has no standardized effect.

When the value is ‘final', the string parameter constitutes the entire string to be displayed. The position of the string relative to the text point parameter is according to TEXT ALIGNMENT.

Text elements with a null string parameter are legal and may be followed by the allowed text attributes and APPEND TEXT as described above.

References:

6.3.4.3
6.6
6.6.3
6.7.3

7.6.5 RESTRICTED TEXT

Parameters:

extent: delta width, delta height (2VDC)
point (P)
flag (one of: not final, final) (E)
string (S)

Description:

RESTRICTED TEXT behaves as does TEXT, with the exception that the text is constrained to a parallelogram determined by the *extent* parameter, the position, and the text attributes. The RESTRICTED TEXT TYPE specifies how the string is positioned within the parallelogram.

The character codes specified in the *string* parameter are interpreted to obtain the associated glyphs from the currently selected character set. The glyphs are displayed on the view surface as specified by the text attributes.

See 6.3.4.3 regarding the legal character codes and the character set repertoire.

The characters are dimensioned according to the CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR and are oriented according to CHARACTER ORIENTATION. The direction of the character placement in the string relative to CHARACTER ORIENTATION is according to TEXT PATH. If GENERALIZED TEXT PATH MODE is ‘off', then the text is positioned relative to the position point of the TEXT element as described in clause 6. If GENERALIZED TEXT PATH MODE is ‘non-tangential' or ‘axis-tangential', elements between the BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements specify the path the text string is to follow, and the method of orienting characters along the path is defined by the mode.

The first component of the *extent* parameter is measured parallel to the base vector of CHARACTER ORIENTATION, and the second component is measured parallel to the up vector. A parallelogram, the text restriction box, is formed whose sides are parallel to the vectors, and which have lengths as in the extent parameter. The box is placed at the position point and aligned as per the current TEXT ALIGNMENT.

The string is displayed within the resulting positioned box. If necessary, the values of the text attributes CHARACTER HEIGHT, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT PRECISION, and TEXT FONT INDEX which are used to display this string are varied to achieve the required restriction. It is only the realized values of these attributes, used to display this single string, which are varied.

For Version 3 and Version 4 metafiles, the way in which the text string is to fit the box may be controlled by the RESTRICTED TEXT TYPE element. Several particular fitting styles are standardized. In some cases this places particular constraints and requirement on how some of the individual text attributes listed above shall be varied. Any of these styles is valid for Version 1 and Version 2 metafiles, but there is no metafile element to select amongst them.

The *flag* parameter is used to define a single display string in a sequence of pieces, and to permit changing the following text attributes and control elements between the pieces:

TEXT FONT INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOUR, CHARACTER HEIGHT, CHARACTER SET INDEX, ALTERNATE CHARACTER SET INDEX, TEXT SCORE TYPE, TEXT BUNDLE INDEX, AUXILIARY COLOUR, and TRANSPARENCY.

When the value of the *flag* parameter is ‘not final', the character codes in the string parameter are accumulated, along with the current attribute settings. In this case, only the attribute setting elements listed above are allowed between this element and the next occurrence of the APPEND TEXT element. With the exception of the ESCAPE element, no other metafile elements of any type are allowed. ESCAPE is permitted but has no standardized effect.

When the value is ‘final', the string parameter constitutes the entire string to be displayed. It is this complete string to which the text restriction box applies. The position of the string relative to the text point parameter is according to TEXT ALIGNMENT. Text elements with a null string parameter are legal and may be followed by the allowed text attributes and APPEND TEXT as described above.

Valid values of the width and height components of the extent are non-negative VDC.

NOTE TEXT PRECISION is included in the attributes which may be changed to achieve the text restriction because TEXT PRECISION controls the relationship between currently set values of text attributes and the values actually used for display of a string (the "realized" values). The realization of the text restriction required by the RESTRICTED TEXT element may mandate another mapping from requested to realized attribute values than would be allowable under the current TEXT PRECISION. Hence the requirements of the current TEXT PRECISION may have to be ignored to achieve proper display of the RESTRICTED TEXT element.

References:

6.3.4.3
6.6
6.6.3
6.7.3
D.4.5.2

7.6.6 APPEND TEXT

Parameters:

flag (one of not final, final) (E)

string (S)

Description:

The character codes specified in the *string* parameter are appended to the string defined by preceding nonfinal TEXT, RESTRICTED TEXT, and APPEND TEXT elements. The codes are interpreted to obtain the associated glyphs from the current character set. The glyphs are displayed on the view surface as specified by the text attributes.

See 6.3.4.3 regarding the legal character codes and the character set repertoire.

The characters are dimensioned according to the CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR and are oriented according to CHARACTER ORIENTATION. The direction of the character placement in the string relative to CHARACTER ORIENTATION is according to TEXT PATH. If GENERALIZED TEXT PATH MODE is ‘off', then text is positioned relative to the position point of the TEXT element as described in clause 4. If GENERALIZED TEXT PATH MODE is ‘non-tangential' or ‘axis-tangential'elements between the BEGIN COMPOUND TEXT PATH and END COMPOUND TEXT PATH elements specify the path the text string is to follow, and the method of orienting characters along the path is defined by the mode.

The *flag* parameter is used to permit changing the following text attributes and control elements within a string which will be aligned as a single block: TEXT FONT INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOUR, CHARACTER HEIGHT, CHARACTER SET INDEX, ALTERNATE CHARACTER SET INDEX, TEXT SCORE TYPE, TEXT BUNDLE INDEX, AUXILIARY COLOUR, and TRANSPARENCY.

If the value of the *flag* is ‘not final', the character codes in the string parameter are accumulated, along with the current attribute settings. In this case, only the attribute setting elements listed above are allowed between this element and the APPEND TEXT element. With the exception of the external element ESCAPE, no other metafile elements of any type are allowed.

If the value is ‘final', the accumulated string parameter constitutes the entire string to be displayed. APPEND TEXT elements with a null string parameter are legal and may be followed by the allowed text attributes and further APPEND TEXT elements as described above.

References:

6.3.4.3
6.6
6.6.3
6.7.3
D.4.5.1

7.6.7 POLYGON

Parameters:

point list (nP)

Description:

A boundary of a polygonal region is defined by connecting each vertex to its successor in the ordered point list with straight edges and connecting the last vertex to the first. The polygonal region may be nonsimple. For example, edges are allowed to cross. In this way, subareas can be created. The interior of the polygon is as defined in 6.6.4.4.

A non-degenerate polygon (one with three or more vertices, not all of which are collinear) is displayed with interior as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, interior style attributes, AUXILIARY COLOUR and TRANSPARENCY. The appearance of the edge is controlled by the edge attributes and by AUXILIARY COLOUR and TRANSPARENCY.

References:

6.6
6.6.4
6.7.4

7.6.8 POLYGON SET

Parameters:

List of:

point (P)
edge out flag (one of: invisible, visible, close invisible, close visible) (E)

Description:

A set of closed polygons is drawn (according to the edge visibility flags and the current edge attributes) and filled (according to the current filled-area attributes).

The list of points and flags is processed sequentially. The first point starts the first polygon of the set; the point that starts each polygon is recorded as the "current closure point". Each point in the list is connected either to its successor or to the current closure point (but not both) by a straight edge.

The *edge out flag* parameter associated with each point in the list defines how the edge coming from that point is generated. The enumerations of the flag mean:

invisible: the edge from point n to point n+1 defines a fill boundary, and is not drawn.

visible: the edge from point n to point n+1 defines a fill boundary, and is drawn.

close invisible: the edge from point n to the current closure point defines a polygon boundary, but is not drawn. The next point in the list (if any) will define the first point of another polygon; it will not be connected by any edge to any point of the polygon being closed.

close visible: as close invisible, but the closing edge added is drawn.

If the edge out flag of the last point in the list is ‘visible', it is treated as ‘close visible'; if the flag is ‘invisible', it is treated as ‘close invisible'.

The interior of the polygon set (see 6.6.4.4) is filled according to the current filled-area attributes. The set of polygons is filled according to the parity (odd or even) algorithm described under the POLYGON element, with the exception that the transition from a vertex marked ‘close visible'or ‘close invisible' to the next point in the point list does not constitute a boundary to the fill algorithm.

The individual polygons of the set are not filled individually. The polygons in the set may be disjoint (as in the ‘dot' and the body of the letter ‘i'), may create ‘holes' (as in a torus shape), or may overlap.

The visible edges are drawn using the current edge attributes. An edge will be drawn only if it was generated with either a ‘visible' or ‘close visible' flag and EDGE VISIBILITY is set to ON. EDGE VISIBILITY thus acts as an override on the visibility of the edges specified in the polygon set, in that it can turn off edges which were specified as ‘visible', but cannot turn on edges which were specified as ‘invisible'.

See figure 30 for an example of POLYGON SET.

NOTE The ability to intermix visible and invisible edges can be used for example to accomodate clipping of polygons before they are placed in the CGM.

References:

6.6.4
6.7.4
D.4.5

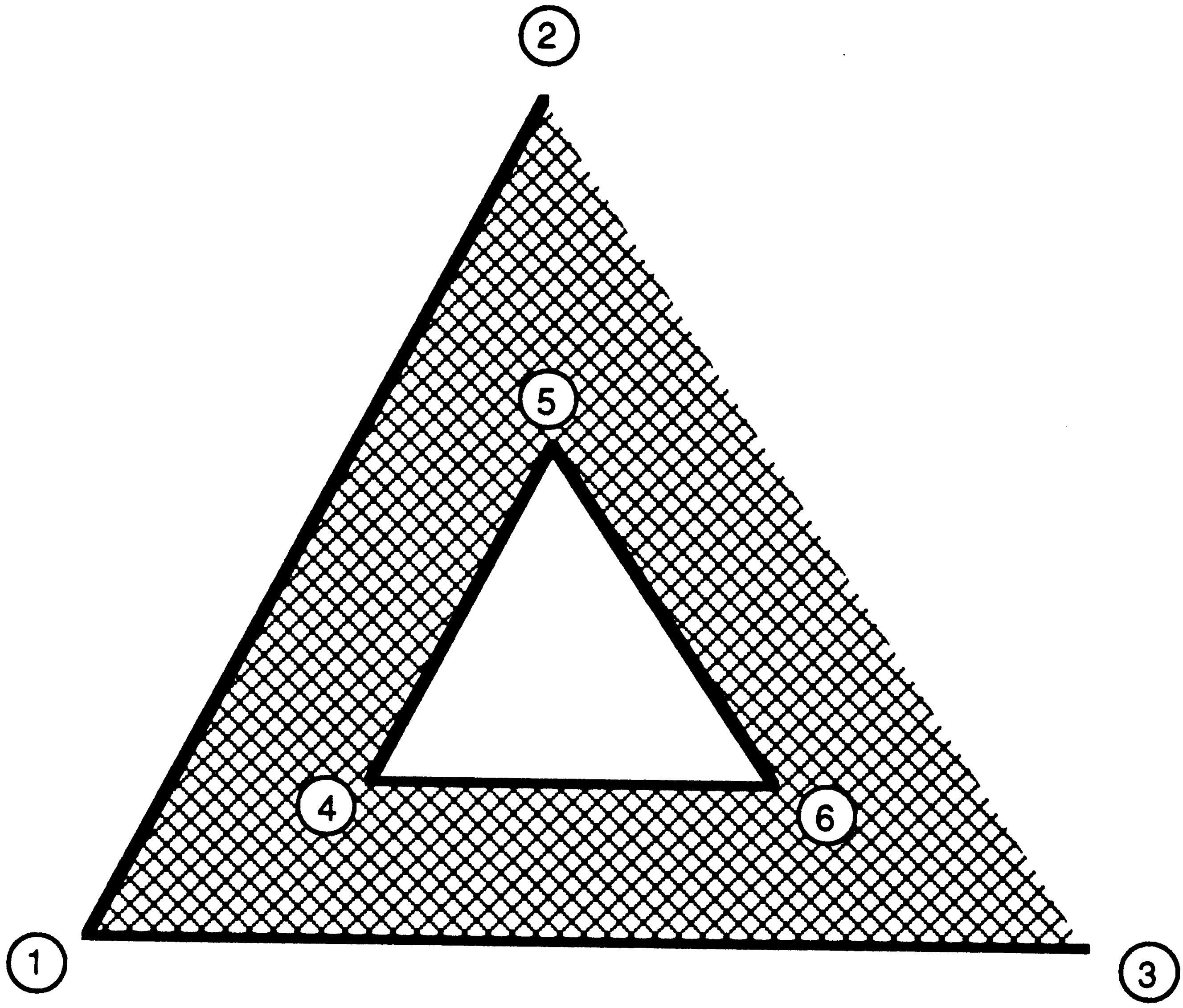


Figure 30 — Example for POLYGON SET

This figure would be generated by the following sequence, where (n) is the notation for the coordinate corresponding to the circled point n on the picture.

(1) visible
(2) invisible
(3) close visible
(4) visible
(5) visible
(6) close visible

7.6.9 CELL ARRAY

Parameters:

3 corner points P, Q, and R (3P)
nx,ny (2I)
local colour precision (form depends upon specific encoding)
cell colour specifiers $(nx * nyCO)$

Description:

In the general case, P,Q,R can delimit an arbitrary parallelogram. P and Q delimit the end points of a diagonal of the parallelogram, and R defines a third corner.

In the simplest case, the three corner points, P,Q,R, define a rectangular area in VDC space. This area is subdivided into $nx \cdot ny$ contiguous rectangles as follows. The edge from P to R is subdivided into nx equal intervals, and the edge from R to Q is subdivided into ny equal intervals. The grid implied consists of $nx \cdot ny$ identical cells. The colour list consists of $nx \cdot ny$ colour specifications, conceptually an array of dimensions nx and ny representing respectively the column and row dimensions. Array element (1,1) is mapped to the cell at corner P, and array element (nx,1) is mapped to the cell at corner R. Array element (nx,ny) is mapped to the cell at corner Q. Hence, the colour elements are mapped within rows running from P to R, and with the rows incrementing in order from R to Q. Array element (1,1) corresponds to the first colour index or colour value stored in the *cell colour specifiers* parameter and array element (nx,1) corresponds to the *nxth* colour index or colour value stored in the *cell colour specifiers* parameter.

The *local colour precision* parameter declares the precision of the *cell colour specifiers.* The precision specification is represented in either ‘indexed’ or 'direct' colour mode, according to the current value of the COLOUR SELECTION MODE element. As with the COLOUR INDEX PRECISION and COLOUR PRECISION elements, the form of the parameter is encoding dependent. If the picture uses indexed colour selection, then the form of the parameter is the same as that of COLOUR INDEX PRECISION. If the picture uses direct colour selection, then the form of the parameter is the same as that of COLOUR PRECISION.

Legal values of local colour precision include the legal values of COLOUR (INDEX) PRECISION. In addition, each encoding defines a special value, the ‘default colour precision indicator', as an indicator that the colour specifiers of the element are to be encoded in the COLOUR (INDEX) PRECISION of the metafile, i.e., to indicate that the local colour precision defaults to COLOUR (INDEX) PRECISION.

Recommendations for the interpretation of cell array for devices that cannot display a cell array are given in annex D.

NOTE Figure 31 illustrates a cell array where the order of mapping the cells to a display surface corresponds to the common left-to-right, top-to-bottom pixel scan order of many devices. In the figure lines indicate cell locations and dots indicate pixel centres.

References:

6.6
6.6.5.1
D.4.5

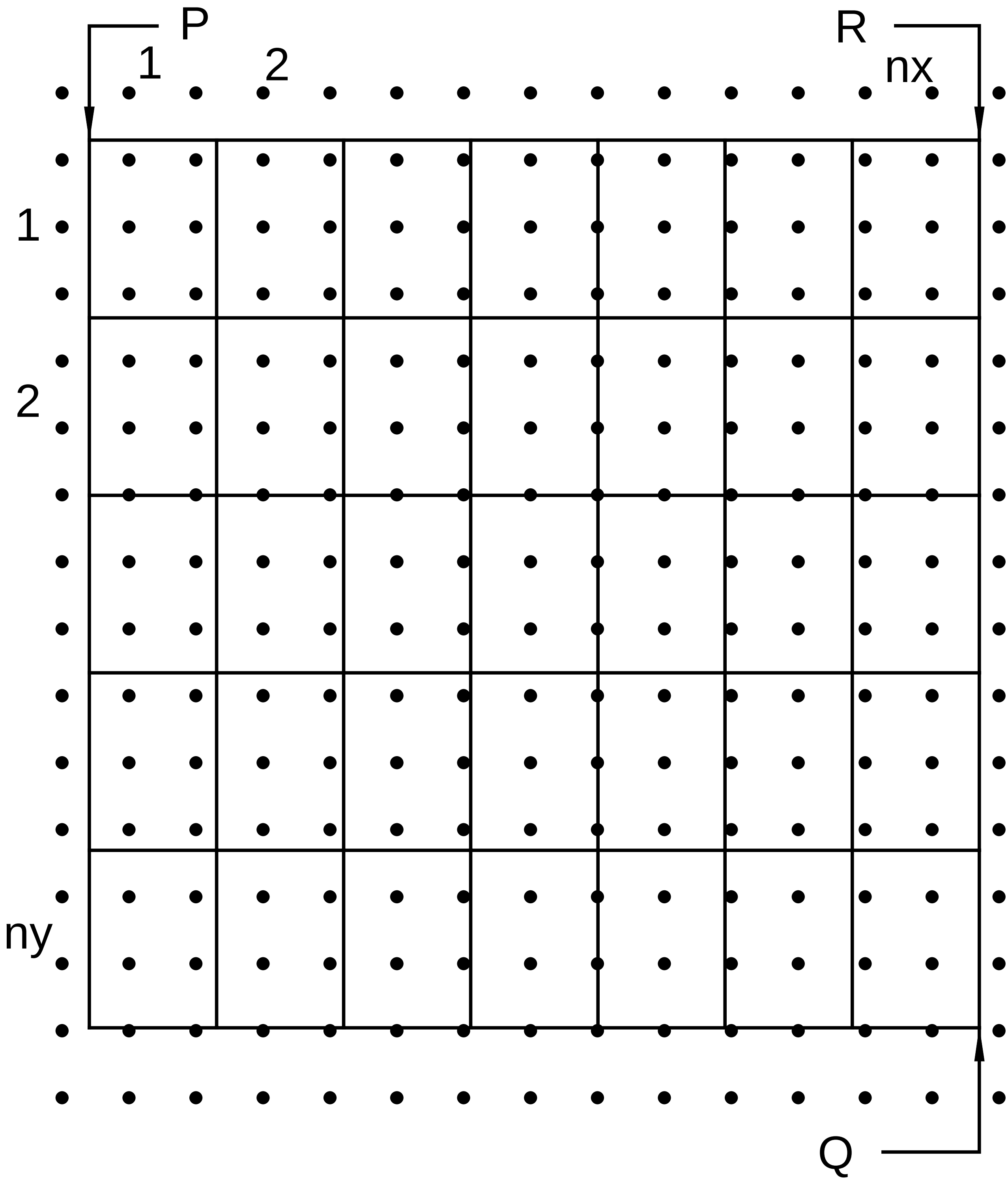


Figure 31 — Nx-by-ny CELL ARRAY rectangle mapped onto display surface

7.6.10 GENERALIZED DRAWING PRIMITIVE (GDP)

Parameters:

identifier (I)
point list (nP)
data record (D)

Description:

A Generalized Drawing Primitive (GDP) of the type specified by the *identifier* parameter is generated on the basis of the points in the *point list* parameter and other information in the *data record* parameter.

Non-negative values of the identifier are reserved for registration and future standardization, and negative values are available for private use.

The appearance of the GDP is determined by zero or more of the attribute sets of the standardized graphical primitive elements, depending on the particular GDP. The parameters of the GDP are interpreted and utilized in an interpreter dependent manner.

See D.4.7 regarding the format of the *data record* parameter.

NOTE 1 GDP provides convenient access to non-standardized graphical primitives that a device may support. GDP is similar to ESCAPE in this sense, but GDP provides a mechanism for handling of coordinate data whereas ESCAPE does not. GDP is thus designed for generating graphical output, and ESCAPE is designed for such applications as non-standardized control functions.

NOTE 2 GDP identifiers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a GDP identifier has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the GDP identifier value will be assigned by the Registration Authority.

References:

6.6
D.4.7

7.6.11 RECTANGLE

Parameters:

two points (2P)

Description:

The *two points* parameter specifies two diagonally opposite corners of a rectangle oriented parallel to the VDC axes. The rectangle so defined is displayed with interior (see 6.6.4.4) as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, AUXILIARY COLOUR, TRANSPARENCY, and interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

References:

6.6
6.6.4
6.7.4

7.6.12 CIRCLE

Parameters:

centrepoint (P)
radius (VDC)

Description:

The *radius* and *centrepoint* parameters specify respectively the radius and centre of a circle, to be displayed with interior (see 6.6.4.4) as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, AUXILIARY COLOUR, TRANSPARENCY, and interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

Valid values of the radius are non-negative VDC.

References:

6.6
6.6.4
6.7.4

7.6.13 CIRCULAR ARC 3 POINT

Parameters:

starting point, intermediate point, ending point (3P)

Description:

A circular arc is displayed from the *starting point*, through the *intermediate point*, to the *ending point*.

A non-degenerate specification is one in which the three specified coordinates are non-collinear.

If the three specified coordinates are collinear the specification is mathematically degenerate, and the interpretation of this element is implementation dependent (see also annex D).

References:

6.6
6.6.1
6.6.6
6.7.1
D.4.5

7.6.14 CIRCULAR ARC 3 POINT CLOSE

Parameters:

starting point, intermediate point, ending point (3P)
close type (one of: pie, chord) (E)

Description:

A filled circular arc is displayed from the specified starting point through the specified intermediate point, to the specified ending point. The close types are illustrated in figure 32.

If close type is ‘chord', the segment defined by the arc and the chord from the starting point to the ending point is displayed with interior (see 6.6.4.4) as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, AUXILIARY COLOUR, TRANSPARENCY, and interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

If close type is ‘pie', the pie sector defined by the computed arc centre, the specified starting point, and the ending point is displayed with interior as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, AUXILIARY COLOUR, TRANSPARENCY, and interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

A non-degenerate specification is one in which the three specified coordinates are non-collinear.

If the three specified coordinates are collinear the specification is mathematically degenerate and ambiguous, and the interpretation of this element is implementation dependent (see also annex D).

References:

6.6
6.6.4
6.6.6
6.7.4
D.4.5

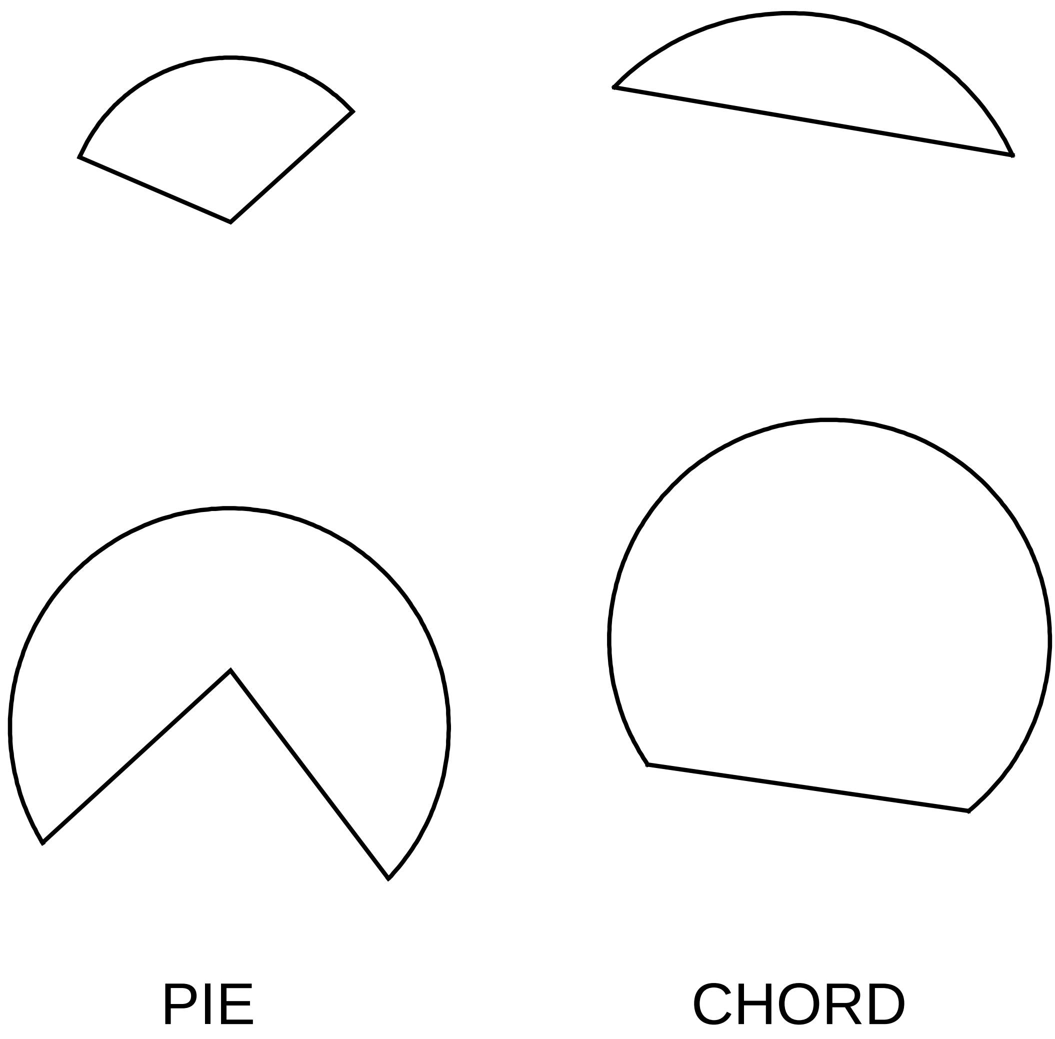


Figure 32 — CIRCULAR ARC 3 POINT CLOSE specifications with ‘pie' and ‘chord’.

7.6.15 CIRCULAR ARC CENTRE

Parameters:

centrepoint (P)
DX_start, DY_start, DX_end, DY_end (4VDC)
radius (VDC)

Description:

A circular arc is drawn which is defined as follows:

The *DX_start* and *DY_start* parameters define a start vector, and the *DX_end* and *DY_end* parameters define an end vector. The tails of these vectors are placed at the position specified by the *centrepoint* parameter. A start ray and end ray are derived from the start and end vectors. The start and end rays are the semi-infinite lines from the centrepoint in the directions of the start and end vectors respectively.

The specified radius and centrepoint define a circle. The arc is drawn in the positive angular direction (as defined by VDC EXTENT) from the intersection of the circle and the start ray (as obtained by measuring a distance *radius* along the start ray from the centrepoint) to the intersection of the circle and the end ray.

The arc is displayed with current line element attributes.

Valid values of the vector components are those which produce vectors of non-zero length.

Valid values of the radius are non-negative VDC.

If the start ray and end ray are coincident, it is ambiguous whether the defined arc subtends 0° or 360° of central angle (see annex D for recommended interpretation).

References:

6.6
6.6.1
6.6.6
6.7.1
D.4.5

7.6.16 CIRCULAR ARC CENTRE CLOSE

Parameters:

centrepoint (P)
DX_start,DY_start,DX_end,DY_end (4VDC)
radius (VDC)
close type (one of: pie, chord) (E)

Description:

A circular arc is drawn and filled which is defined as follows:

The *DX_start* and *DY_start* parameters define a start vector, and the *DX_end* and *DY_end* parameters define an end vector. The tails of these vectors are placed at the position specified by the *centrepoint* parameter. A start ray and end ray are derived from the start and end vectors. The start and end rays are the semi-infinite lines from the centrepoint in the directions of the start and end vectors respectively.

The specified radius and centrepoint define a circle. The arc is drawn in the positive angular direction (as defined by VDC EXTENT) from the intersection of the circle and the start ray (as obtained by measuring a distance *radius* along the start ray from the centrepoint) to the intersection of the circle and the end ray.

If the value of the *close type* parameter is ‘chord', the circular segment defined by the arcand the chord from the starting point to the ending point of the arc is displayed.

If the value is ‘pie', the circular sector defined by the arc and the specified centrepoint is displayed.

The primitive is displayed with interior (see 6.6.4.4) as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, and interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

Valid values of the vector components are those which produce vectors of non-zero length.

Valid values of the radius are non-negative VDC.

If the start ray and end ray are coincident, it is ambiguous whether the defined arc subtends 0° or 360° of central angle (see annex D for recommended interpretation).

References:

6.6
6.6.4
6.6.6
6.7.4
D.4.5

7.6.17 ELLIPSE

Parameters:

centrepoint (P)
first CDP endpoint (P)
second CDP endpoint (P)

Description:

The *centrepoint* parameter specifies the centre of an ellipse. The CDP endpoints include one endpoint from each conjugate diameter; together with the centrepoint they define the two conjugate diameters of the ellipse.

The ellipse so specified is displayed with interior (see 6.6.4.4) as defined by the FILL BUNDLE INDEX, ASPECT SOURCE FLAGS, AUXILIARY COLOUR, TRANSPARENCY, and interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

Valid values of the three specifying points of the ellipse are those which yield three distinct points. The specified ellipse is non-degenerate if and only if the three points are non-collinear.

References:

6.6
6.6.4
6.6.7
6.7.4
D.4.5

7.6.18 ELLIPTICAL ARC

Parameters:

centrepoint (P)
first CDP endpoint (P)

second CDP endpoint (P)
DX_start,DY_start,DX_end,DY_end (4VDC)

Description:

An elliptical arc is drawn which is defined as follows:

The *centrepoint* parameter specifies the center of an ellipse. The CDP endpoints include one endpoint from each conjugate diameter; together with the centrepoint they define the two conjugate diameters of the ellipse.

The *DX_start* and *DY_start* parameters define a start vector, and the *DX_end* and *DY_end* parameters define an end vector. The tails of these vectors are placed at the position specified by the *centrepoint* parameter. A start ray and end ray are derived from the start and end vectors. The start and end rays are the semi-infinite lines from the centrepoint in the directions of the start and end vectors respectively.

The defined arc begins at the intersection of the ellipse and the start ray and follows the ellipse to the intersection of the ellipse and the end ray in the direction defined as follows. A "conjugate radius" is defined to be half of a conjugate diameter. Letting the centrepoint be labelled M, the first CDP endpoint P1, and the second CDP endpoint P2, then the line segments M-P1 and M-P2 define two conjugate radii, referred to in what follows as the first conjugate radius and the second conjugate radius respectively. The conjugate radii meet at M and define two angles: the sum of the two angles is 360°, one angle is less than 180° and the other is greater than 180°. The drawing direction of the elliptical arc is the direction from the first conjugate radius to the second conjugate radius through the smaller of these two angles.

The elliptical arc is displayed with the current line attributes.

Valid values of the three specifying points of the ellipse are those which yield three distinct points. The specified ellipse is non-degenerate if and only if the three points are non-collinear.

Valid values of the vector components are those which produce vectors of non-zero length. If the start ray and end ray are coincident, it is ambiguous whether the defined arc is null (zero arc length) or the entire ellipse (see annex D for recommended interpretation).

References:

6.6
6.6.1
6.6.7
6.7.1
D.4.5

7.6.19 ELLIPTICAL ARC CLOSE

Parameters:

centrepoint (P)
first CDP endpoint (P)
second CDP endpoint (P)
DX_start,DY_start,DX_end,DY_end (4VDC)
close type (one of: pie, chord) (E)

Description:

An elliptical arc is drawn and filled which is defined as follows:

The *centrepoint* parameter specifies the center of an ellipse. The CDP endpoints include one endpoint from each conjugate diameter; together with the centrepoint they define the two conjugate diameters of the ellipse.

The *DX_start* and *DY_start* parameters define a start vector, and the *DX_end* and *DY_end* parameters define an end vector. The tails of these vectors are placed at the position specified by the *centrepoint* parameter. A start ray and end ray are derived from the start and end vectors. The start and end rays are the semi-infinite lines from the centrepoint in the directions of the start and end vectors respectively.

The defined arc begins at the intersection of the ellipse and the start ray (the "starting point") and follows the ellipse to the intersection of the ellipse and the end ray (the "ending point") in the direction defined as follows. A "conjugate radius" is defined to be half of a conjugate diameter. Letting the centrepoint be labelled M, the first CDP endpoint P1, and the second CDP endpoint P2, then the line segments M-P1 and M-P2 define two conjugate radii, referred to in what follows as the first conjugate radius and the second conjugate radius respectively. The conjugate radii meet at M and define two angles: the sum of the two angles is 360°, one angle is less than 180° and the other is greater than 180°. The drawing direction of the elliptical arc is the direction from the first conjugate radius to the second conjugate radius through the smaller of these two angles.

If the value of the *close type* parameter is ‘chord', the segment defined by the elliptical arc and the chord from the starting point to the ending point is displayed with interior (see 6.6.4.4) as defined by the FILL BUNDLE INDEX, FILL ASF, and interior style attributes. The appearance of the edge is controlled by edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

If the value is ‘pie', the elliptical pie sector defined by the elliptical arc centrepoint, the starting point, and the ending point is displayed with interior as defined by the FILL BUNDLE INDEX, FILL ASF, and the interior style attributes. The appearance of the edge is controlled by the edge attributes, and by AUXILIARY COLOUR and TRANSPARENCY.

Valid values of the three specifying points of the ellipse are those which yield three distinct points. The specified ellipse is non-degenerate if and only if the three points are non-collinear.

Valid values of the vector components are those which produce vectors of non-zero length.

If the start ray and end ray are coincident, it is ambiguous whether the defined arc is null (zero arc length) or the entire ellipse (see annex D for recommended interpretation).

References:

6.6
6.6.4
6.6.7
6.7.4
D.4.5

7.6.20 CIRCULAR ARC CENTRE REVERSED

Parameters:

centrepoint (P)
DX_start, DY_start, DX_end, DY_end (4VDC)
radius (VDC)

Description:

A circular arc is drawn which is defined as follows:

The *DX_start* and *DY_start* parameters define a start vector, and the *DX_end* and *DY_end* parameters define an end vector. The tails of these vectors are placed at the position specified by the *centrepoint* parameter. A start ray and end ray are derived from the start and end vectors. The start and end rays are semi-infinite lines from the centrepoint in the directions of the start and end vectors respectively.

The values of the *radius* and *centrepoint* parameters define a circle. The arc is drawn in the negative angular direction (as defined by VDC EXTENT) from the intersection of the circle and the start ray (as obtained by measuring a distance *radius* along the start ray from the centrepoint) to the intersection of the circle and the end ray.

The arc is displayed with current line element attributes.

Valid values of the vector components are those which produce vectors of non-zero length.

Valid values of the radius are non-negative VDC.

If the start ray and end ray are coincident, it is ambiguous whether the defined arc subtends 0° or 360° of central angle (see annex D for recommended interpretation).

References:

6.6.
6.6.1
6.6.6
6.7.1
D.4.5

7.6.21 CONNECTING EDGE

Parameters:

none

Description:

During the construction of a closed figure a line segment connecting the last point of the preceding line element and the next point is added to the boundary and edge definitions. The next point may be either:

1. the first point of the next line element, or
2. the current closure point (in cases where CONNECTING EDGE is followed by either NEW REGION or END FIGURE).

The appearance of the connecting edge is fully determined by the edge attributes including EDGE VISIBILITY.

References:

6.6.11

7.6.22 HYPERBOLIC ARC

Parameters:

centre point (P)
transverse radius endpoint (P)
conjugate radius endpoint (P)
start vector (2VDC)
end vector (2VDC)

Description:

A hyperbolic arc is defined. The asymptotes of the full hyperbola pass through the *centre point* and are parallel to two vectors defined by the sum and difference of the vectors from the centre to the points defined by the *transverse radius end point* and *conjugate radius endpoint* parameters, respectively. The complete hyperbola passes through the transverse radius endpoint and its tangent there is parallel to the vector from the centre point to the conjugate radius endpoint. The defined arc is a finite arc starting and ending at the points where the rays from the centre in the directions of the start and end vectors intersect the complete hyperbola. See clause 6 for further discussion of the geometric significance of the parameterization and details of rendering of hyperbolic arcs.

References:

6.6
6.6.1
6.6.8
6.7.1

7.6.23 PARABOLIC ARC

Parameters:

centre point (P)
start point (P)
end point (P)

Description:

A parabolic arc is defined. A parabolic arc is drawn from the point defined by the *start point* parameter to the point defined by the *end point* parameter. The point defined by the *centre point* parameter is the intersection of the tangents to the parabola at the start point and end point. See clause 6 for further discussion of the geometric significance of the parameterization and details of rendering of parabolic arcs.

References:

6.6
6.6.1
6.6.9
6.7.1

7.6.24 NON-UNIFORM B-SPLINE

Parameters:

spline order (I)
number of control points (I)
control points (nP)
list of knots ((k+n)R)
parameter start value (R)
parameter end value (R)

Description:

The value of the *spline order* parameter shall be positive. The *list of knots* parameter shall form a non-decreasing sequence of numbers (see clause 6). The *number of control points* parameter value shall be at least as large as the spline order. The sum of the number of control points and the spline order shall equal the number of knots. If the spline order is k and the number of control points is n then the number of knots is $(n+k)$. N shall be positive.

The values defined by the *parameter start value* and *parameter end value* specify over what range of the parameter the B-spline curve is evaluated. The start value shall be less than or equal to the end value. The start value shall be greater than or equal to the value of the *k-th* knot in this sequence, where k is the spline order. The end value shall be less than the *n-th* knot value (where n is the number of control points).

When an element of this type is interpreted, a non-uniform B-spline curve is generated for parameter values between the parameter start value and parameter end value.

References:

6.6
6.6.10
6.6.10.1

7.6.25 NON-UNIFORM RATIONAL B-SPLINE

Parameters:

spline order (I)
number of control points (I)
control points (nP)
list of knots ((k+n)R)
parameter start value (R)
parameter end value (R)
weights (nR)

Description:

The value of the *spline order* parameter shall be positive. The *list of knots* parameter shall form a non-decreasing sequence of numbers (see clause 6). The *number of control points* parameter value shall be at least as large as the spline order. The sum of the number of control points and the spline order shall equal the number of knots. If the spline order is *k* and the number of control points is *n* then the number of knots is *(n+k)*. *N* shall be positive.

The array of numbers defined by the *weights* parameter contains one real number for each control point in the array of numbers defined by the *control points* parameter.

The values defined by the *parameter start value* and *parameter end value* specify over what range of the parameter the B-spline curve is evaluated. The start value shall be less than or equal to the end value. The start value shall be greater than or equal to the value of the *k-th* knot in this sequence, where *k* is the spline order. The end value shall be less than or equal to the *n-th* knot value (where *n* is the number of control points).

When an element of this type is interpreted, a non-uniform rational B-spline curve is generated for parameter values between the parameter start value and parameter end value.

References:

6.6
6.6.10
6.6.10.1

7.6.26 POLYBEZIER

Parameters:

continuity indicator (IX)
point list

if the *continuity indicator* is 1 (discontinuous) (4nP)

if the *continuity indicator* is 2 (continuous) ((3n+1)P)

Description:

This element defines one or more cubic Bezier curves. The association of points in the parameter list with control points is dependent upon the value of the *continuity indicator* parameter and is defined in clause 4. The cubic parametric equations defining the specified Bezier curves are given in clause 4.

The relationship of the Nth Bezier curve to the (N-1)th, if there is more than one curve, is specified by the continuity indicator. Valid values are:

1: discontinuous — successive curves may be disjoint;

2: continuous — successive curves are connected, final point of Nth curve matches initial point of (N+1)th.

References:

6.6
6.6.10
6.6.10.2

7.6.27 POLYSYMBOL

Parameters:

symbol index (IX)
position point list (nP)

Description:

The symbol corresponding to the symbol index parameter in the symbol library specified by the current SYMBOL LIBRARY INDEX is dimensioned according to SYMBOL SIZE, oriented according to SYMBOL ORIENTATION, and drawn at each point in the *position point list*. The symbol is displayed according to the current SYMBOL COLOUR.

References:

6.6
6.6.12

7.6.28 BITONAL TILE

Parameters:

compression type (IX)
row padding indicator (I)
cell background colour (CO)
cell foreground colour (CO)
method-specific parameters (SDR)
compressed colour specifiers (BS)

Description:

The *compression type* parameter specifies the compression type used. The following methods are defined:

0: null background
1: null foreground
2: T6
3: T4 1-dimensional
4: T4 2-dimensional
5: bitmap (uncompressed)
6: run length

Compression types greater than 6 are reserved for registration and future standardization.

The compression types ‘null background' and ‘null foreground' indicate that all cells in the tile are known to be background or foreground respectively. In this case the tile has no encoded content. The bitstream parameter is null.

If the method is ‘T4', the image is encoded according to the one or two dimensional scheme defined by CCITT Recommendation T4 (Group 3 facsimile). If the method is ‘T6', the image is encoded according to the two dimensional scheme defined in CCITT Recommendation T6 (Group 4 facsimile).

If the compression type is ‘run length', then sequences of colour specifiers with identical values are represented by a pair of values: an integer count followed by a colour specifier. The integer is an unsigned binary integer at a fixed precision specified by an integer value, *n*, in the parameter *method-specific parameters*. The bitstream consists of consecutive packets of *n+1* bits, where the first *n* bits specify the run length and the remaining bit specifies the cell colour.

The colour specifiers of the *compressed colour specifiers* parameter is compressed according to the value of the *compression type* parameter and stored in the metafile as a compressed binary data object.

The value of *row padding indicator* parameter shall be a non-negative integer. The row padding indicator indicates whether there is any padding of rows prior to compression. If the value is 0, then there is no padding and the compressed row contains exactly as many colour specifiers as indicated in the parameter *number of cells/tile in path direction* of the BEGIN TILE ARRAY element. If the value is *n*, greater than 0, then the parameter *number of cells/tile in path direction* of the BEGIN TILE ARRAY element defines the number of actual cells with image content in the row. Prior to compression the row is padded with colour specifiers of value 0, if necessary, so that the number of specifiers in the row prior to compression is a multiple of *n*. Any colour specifiers added to the row to satisfy this requirement are not included in the value of the *the number of cells/tile in path direction* parameter of the BEGIN TILE ARRAY element.

The order of the data in the BS parameter is that order sometimes referred to as "down". The first bit of data resulting from the compression process is the first or most signficant bit of the first octet of the BS parameter.

The cell colour specifiers have only two values, the indexes 0 and 1. Index 0 designates the cell background colour. Index 1 designates the cell foreground colour. The precompressed or uncompressed colour specifiers considered as a binary data stream are represented at 1 bit per cell. The parameter *method-specific parameters* contains parameters that are specific to particular compression types. The SDR for each of the defined compression types contains:

Compression type	**SDR contents**
0: null background	null
1: null foreground	null
2: T6	null
3: T4 1-dimensional	null
4: T4 2-dimensional	null
5: bitmap (uncompressed)	null
6: run length	run-count precision (I)

NOTE 1 The encoding of SDR in each of the standard encodings of this International Standard supports the concept of a "null" record, which is to be distinguished from an omitted record.

NOTE 2 Compression method values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a compression method has been approved by ISO/IEC Subcommittee for *Computer Graphics*, the compression method value will be assigned by the Registration Authority.

References:

6.6
6.6.5
6.6.5.2
D.4.6

7.6.29 TILE

Parameters:

compression type (IX)
row padding indicator (I)
cell colour precision (form depends upon specific encoding)
method-specific parameters (SDR)
compressed colour specifiers (BS)

Description:

The *compression type* parameter specifies the method by which the *compressed colour specifiers* are compressed. The following methods are defined:

0: null background
1: null foreground
2: T6
3: T4 1-dimensional
4: T4 2-dimensional
5: bitmap (uncompressed)
6: run length

Compression types greater than 6 are reserved for registration and future standardization.

Compression types ‘null background' and ‘null foreground' indicate that all cells in the tile are known to be background or foreground respectively. In this case the tile has no encoded content. The bitstream parameter is null. For indexed colour selection mode, background corresponds to index 0 and foreground corresponds to index 1. For direct colour selection mode, background corresponds to the value defined by the BACKGROUND COLOUR element (or its default) and foreground corresponds to an implementation dependent foreground colour.

If the method is T4, the image is encoded according to the one or two dimensional scheme defined by CCITT Recommendation T4 (Group 3 facsimile). If the method is T6, the image is encoded according to the two dimensional scheme defined in CCITT Recommendation T6 (Group 4 facsimile).

If the compression type is ‘run length', then sequences of colour specifiers with identical values are represented by a pair of values: an integer count followed by a colour specifier. The integer is an unsigned binary integer at a fixed precision specified by an integer value, *n*, in the parameter *method-specificparameters*. The bitstream consists of consecutive packets of n+m bits, where the first n bits specify the run length and the remaining m bits specify the cell colour. The value of m depends upon whether the Colour Selection Mode is indexed or direct, and depends as well on the colour model in the latter case. If k is the effective cell colour precision (see the description of the cell colour precision parameter), expressed in bits, then m=k for indexed colour selection, and m=3k or m=4k for direct colour selection.

The sequence of compressed colour specifiers is compressed according to the value of the *compression type* parameter and stored in the metafile as a compressed binary data object.

The value of *row padding indicator* parameter shall be a non-negative integer. The row padding indicator indicates whether there is any padding of rows prior to compression. If the value is 0, then there is no padding and the compressed row contains exactly as many colour specifiers as indicated in the parameter *number of cells/tile in path direction* of the BEGIN TILE ARRAY element. If the value is *n*, greater than 0, then the parameter *number of cells/tile in path direction* of the BEGIN TILE ARRAY element defines the number of actual cells with image content in the row. Prior to compression the row is padded with the least number of colour specifiers of value 0, if necessary, so that the number of specifiers in the row prior to compression is a multiple of *n*. Any colour specifiers added to the row to satisfy this requirement are not included in the value of the *the number of cells/tile in path direction* parameter of the BEGIN TILE ARRAY element.

The *cell colour precision* parameter declares the precision of the cells in the *compressed colour specifiers* parameter. This is the precision of the uncompressed colour specifiers. The precision specification is represented in either 'indexed' or 'direct' colour mode, according to the current value of the COLOUR SELECTION MODE element. As with the COLOUR INDEX PRECISION and COLOUR PRECISION elements, the form of the parameter is encoding dependent. If the picture uses indexed colour selection, then the form of the parameter is the same as that of COLOUR INDEX PRECISION. If the picture uses direct colour selection, then the form of the parameter is the same as that of COLOUR PRECISION.

Legal values of local colour precision include the legal values of COLOUR (INDEX) PRECISION. In addition, each encoding defines a special value, the 'default colour precision indicator', as an indicator that the colour specifiers of the element are to be encoded in the COLOUR (INDEX) PRECISION of the metafile, i.e., to indicate that the cell colour precision defaults to COLOUR (INDEX) PRECISION.

The order of the data in the *compressed colour specifiers* parameter is that order sometimes referred to as "down". The first bit of data resulting from the compression process is the first or most signficant bit of the first octet of the BS parameter.

The *cell colour precision* parameter defines the colour precision of the colour specifiers in the pre-compression (uncompressed) data stream. When decompressing the Bitstream operand, these are the precisions of the binary data comprising the individual colour specifiers.

The parameter *method-specific parameters* contains parameters that are specific to particular compression types. The SDR for each of the defined compression types contains:

Compression type	**SDR contents**
0: null background	null
1: null foreground	null
2: T6	null
3: T4 1-dimensional	null
4: T4 2-dimensional	null
5: bitmap (uncompressed)	null
6: run length	run-count precision (I)

NOTE 1 The encoding of SDR in each of the standard encodings of this International Standard supports the concept of a "null" record, which is to be distinguished from an omitted record.

NOTE 2 T4 and T6 compression methods are not likely to give useful results if the cell colour precision is other than 1 and the colour selection mode is direct.

NOTE 3 Compression method values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a compression method has been approved by ISO/IEC Subcommittee for *Computer Graphics*, the compression method value will be assigned by the Registration Authority.

References:

6.6
6.6.5
6.6.5.2
D.4.6

7.7 Attribute elements

7.7.1 LINE BUNDLE INDEX

Parameters:

line bundle index (IX)

Description:

The line bundle index is set to the value specified by the parameter. When subsequent line elements occur, the values for LINE TYPE, LINE WIDTH, and LINE COLOUR are taken from the corresponding components of the indexed bundle if the ASFs for those attributes are set to ‘bundled'. See 6.6 for a list of line elements.

If the ASF for a given attribute is ‘individual', this element does not affect the value used for that attribute until the ASF returns to ‘bundled'.

Legal values are positive integers.

References:

6.6.1
6.7.1.1
D.4.6

7.7.2 LINE TYPE

Parameters:

line type indicator (IX)

Description:

The line type indicator is set to the value specified by the parameter.

When the LINE TYPE ASF is ‘individual', subsequent line elements are displayed with this line type. See 6.6 for a list of line elements.

When the LINE TYPE ASF is ‘bundled', this element does not affect the display of subsequent line elements until the ASF returns to ‘individual'.

The following line types are assigned:

1: solid
2: dash
3: dot
4: dash-dot
5: dash-dot-dot

Values above 5 are reserved for registration and future standardization, and negative values are available for implementation-dependent use.

NOTE 1 Line type continuity is addressed in Version 3 and Version 4 metafiles with the LINE TYPE CONTINUATION element. In Version 1 and Version 2 metafiles, ideally the line type is maintained continuously between adjacent spans of a single line element (see annex D for further discussion.) ISO/IEC 8632 does not specify continuity between separate, but graphically connected, line elements; nor does it specify continuity across sections of a single line element that may have been clipped away.

NOTE 2 Whether or not line type is maintained continuously across the segments of DISJOINT POLYLINE is not addressed by ISO/IEC 8632.

NOTE 3 Line type indicator values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a line type indicator has been approved by ISO/IEC Subcommittee for *Computer Graphics*, the line type indicator value will be assigned by the Registration Authority.

References:

6.6.1
6.7.1.1
D.4.6

7.7.3 LINE WIDTH

Parameters:

line width specifier (SS)

Description:

The absolute line width or line width scale factor is set as specified by the parameter.

When the LINE WIDTH ASF is 'individual', subsequent line elements are displayed according to the size specification of this element. See 6.6 for a list of line elements.

When the LINE WIDTH ASF is 'bundled', this element does not affect the display of subsequent line elements until the ASF returns to 'individual'.

Valid values of the 'line width specifier' parameter are non-negative VDC or R (according to the value of the LINE WIDTH SPECIFICATION MODE element).

NOTE The line width is measured perpendicular to the defining line (that is, it is independent of the orientation of the defining line). A wide line is aligned with its ideal zero-width defining line such that the distance between the defining line and either edge is half the line width.

References:

6.4.3
6.6.1
6.7.1.1
6.7.5
D.4.6

7.7.4 LINE COLOUR

Parameters:

line colour specifier (CO)

Description:

The line colour index or line colour value is set as specified by the parameter.

When the LINE COLOUR ASF is 'individual', subsequent line elements are drawn in this line colour. See 6.6 for a list of line elements.

When the LINE COLOUR ASF is 'bundled', this element does not affect the interpretation of subsequent line elements until the ASF returns to 'individual'.

References:

6.6.1
6.7.1.1

6.7.6
D.3.2

7.7.5 MARKER BUNDLE INDEX

Parameters:

marker bundle index (IX)

Description:

The marker bundle index is set to the value specified by the parameter. When subsequent marker elements occur, the values for MARKER TYPE, MARKER SIZE, and MARKER COLOUR are taken from the corresponding components of the indexed bundle if the ASFs for those attributes are ‘bundled'.

If the ASF for a given attribute is ‘individual', this element does not affect the value used for that attribute until its ASF returns to ‘bundled'.

Legal values of MARKER BUNDLE INDEX are positive integers.

References:

6.6.2
6.7.2.1
D.4.6

7.7.6 MARKER TYPE

Parameters:

marker type (IX)

Description:

The marker type is set to the value specified by the parameter.

When the MARKER TYPE ASF is ‘individual', subsequent marker elements are displayed with this marker type.

When the MARKER TYPE ASF is ‘bundled', this element does not affect the display of subsequent marker elements until the ASF returns to ‘individual'.

The following marker types are assigned:

1: dot (.)
2: plus (+)
3: asterisk (*)
4: circle (o)
5: cross (x)

The marker type ‘dot' is intended always to be displayed as the smallest visible point on the display surface at metafile interpretation time. It is thus intended to behave as a "polypoint" element.

Values above 5 are reserved for registration and future standardization, and negative values are available for implementation-dependent use.

NOTE Marker type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a marker type has been approved by ISO/IEC Subcommittee for *Computer Graphics*, the marker type value will be assigned by the Registration Authority.

References:

6.6.2
6.7.2.1
D.4.6

7.7.7 MARKER SIZE

Parameters:

marker size specifier (SS)

Description:

The absolute marker size or marker size scale factor is set as specified by the parameter. If absolute, the specified size is the maximum extent of the marker.

When the MARKER SIZE ASF is 'individual', subsequent marker elements are displayed according to the size specification of this element.

When the MARKER SIZE ASF is 'bundled', this element does not affect the display of subsequent marker elements until the ASF returns to 'individual'.

Valid values of the 'marker size specifier' parameter are non-negative VDC or R (according to the value of the MARKER SIZE SPECIFICATION MODE element).

References:

6.4.3
6.6.2
6.7.2.1
6.7.5
D.4.6

7.7.8 MARKER COLOUR

Parameters:

marker colour specifier (CO)

Description:

The marker colour index or marker colour value is set as specified by the parameter.

When the MARKER COLOUR ASF is 'individual', subsequent marker elements are displayed with this marker colour.

When the MARKER COLOUR ASF is 'bundled', this element does not affect the display of subsequent marker elements until the ASF returns to 'individual'.

References:

6.6.2
6.7.2.1
6.7.6
D.3.2

7.7.9 TEXT BUNDLE INDEX

Parameters:

text bundle index (IX)

Description:

The text bundle index is set to the value specified by the parameter. When subsequent text elements occur, the values for TEXT FONT INDEX, TEXT PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, and TEXT COLOUR are taken from the corresponding components of the indexed bundle if the ASFs for those attributes are set to 'bundled'. See 6.6 for a list of text elements.

If the ASF for a given attribute is 'individual', this element does not affect the value used for that attribute until the ASF returns to 'bundled'.

Legal values of the *text bundle index* parameter are positive integers.

References:

6.6.3
6.7.3
D.4.6

7.7.10 TEXT FONT INDEX

Parameters:

font index (IX)

Description:

The font index is set to the value specified by the parameter. The font index is used to select a font from the font list table defined in the Metafile Descriptor.

When the TEXT FONT INDEX ASF is 'individual', subsequent text elements are displayed with this font index. See 6.6 for a list of text elements.

When the TEXT FONT INDEX ASF is 'bundled', this element does not affect the display of subsequent text elements until the ASF returns to 'individual'.

Legal values of the *font index* parameter are positive integers.

NOTE 1 Metafile generators should ensure that the selected character set and text font are compatible.

NOTE 2 Annex D gives recommendations for interpreters to follow in the case that the currently selected character set cannot be rendered in the specified text font.

References:

6.6.3
6.7.3
D.4.6

7.7.11 TEXT PRECISION

Parameters:

text precision (one of: string, character, stroke) (E)

Description:

The text precision is set to the value specified by the parameter.

When the TEXT PRECISION ASF is ‘individual', subsequent text elements are displayed with this text precision. See 6.6 for a list of text elements.

When the TEXT PRECISION ASF is ‘bundled', this element does not affect the display of subsequent text elements until the ASF returns to ‘individual'.

The accuracy of execution of TEXT attributes can be controlled by one of three values.

If ‘string' precision is specified, only the text position of subsequent text strings need be guaranteed, and the manner in which the string is clipped is implementation dependent.

If ‘character' precision is specified, the metafile interpreter guarantees that the starting position of each character satisfy the relevant text attributes, thus guaranteeing orientation and placement of the string; however, skew, orientation, and size of each character are not guaranteed. All characters of the string which lie completely inside or outside the clipping region are clipped as appropriate, but the effect of clipping on a character whose character box is intersected by the clipping boundary is implementation dependent.

If ‘stroke' precision is specified, the metafile interpreter guarantees that the placement, skew, orientation, and size of all characters satisfy all standardized text attributes. Characters are clipped to the geometric accuracy of the device.

References:

6.6.3
6.7.3
D.4.6

7.7.12 CHARACTER EXPANSION FACTOR

Parameters:

character expansion factor (R)

Description:

The character expansion factor is set to the value specified by the parameter.

When the CHARACTER EXPANSION FACTOR ASF is ‘individual', subsequent text elements are displayed with this character expansion factor. See 6.6 for a list of text elements.

When the CHARACTER EXPANSION FACTOR ASF is ‘bundled', this element does not affect the display of subsequent text elements until the ASF returns to ‘individual'.

The character expansion factor specifies the deviation of the width-to-height ratio of the character from the ratio indicated by the font designer.

Legal values of the *character expansion factor* are non-negative reals.

NOTE The character expansion factor is a scalar. The resulting character width is the product of the CHARACTER HEIGHT multiplied by the width/height ratio (a characteristic of each font and a quantity that can vary on a character-by-character basis) for the character multiplied by the CHARACTER EXPANSION FACTOR. The character width so derived is further scaled by multiplying it by the ratio of the length of the character base vector to the length of the character up vector.

References:

6.6.3
6.7.3
D.4.6

7.7.13 CHARACTER SPACING

Parameters:

character spacing (R)

Description:

The character spacing is set to the value specified by the parameter.

When the CHARACTER SPACING ASF is ‘individual', subsequent text elements are displayed with this character spacing. See 6.6 for a list of text elements.

When the CHARACTER SPACING ASF is ‘bundled', this element does not affect the display of subsequent text elements until the ASF returns to ‘individual'.

The parameter represents the desired space to be added between character bodies of a text string, which is in addition to any intercharacter spacing provided by the font within the character's body. It is specified as a fraction of the current CHARACTER HEIGHT attribute. The space is added along the text path. A negative value implies that characters may overlap.

When TEXT PATH is 'right' or 'left', the character spacing is scaled by the ratio of the length of the character base vector to the length of the character up vector.

References:

6.6.3
6.7.3
D.4.6

7.7.14 TEXT COLOUR

Parameters:

text colour specifier (CO)

Description:

The text colour index or text colour value is set as specified by the parameter.

When the TEXT COLOUR ASF is ‘individual', subsequent text elements are displayed with this text colour. See 6.6 for a list of text elements.

When the TEXT COLOUR ASF is ‘bundled', this element does not affect the display of subsequent text elements until the ASF returns to ‘individual'.

References:

6.6.3
6.7.3
6.7.6
D.3.2

7.7.15 CHARACTER HEIGHT

Parameters:

character height (VDC)

Description:

The character height is set to the value specified by the parameter. Subsequent text elements are displayed with this character height. See 6.6 for a list of text elements.

The parameter represents the desired height of the character body, from baseline to capline, in VDC units; it is a positive number. It is measured along the character up vector. If the character orientation vectors are not orthogonal, this will not be the perpendicular distance between baseline and capline.

Valid values of ‘character height' are non-negative VDC.

References:

6.6.3
6.7.3.2
D.4.6

7.7.16 CHARACTER ORIENTATION

Parameters:

x character up component (VDC)
y character up component (VDC)
x character base component (VDC)
y character base component (VDC)

Description:

The two vectors define the orientation and skew of the character body in subsequent text elements. See 6.6 for a list of text elements. For purposes of alignment and path, ‘up' is in the direction of the character up vector and ‘right' is in the direction of the character base vector. The ratio of the length of the base vector to the length of the up vector is used as a scaling factor for the CHARACTER EXPANSION FACTOR and CHARACTER SPACING elements.

The two vectors shall be non-collinear and shall have positive length.

NOTE The way in which software above the metafile generator and/or the metafile generator itself may use this element is as follows. A vector whose length is the character height (baseline-to-capline) and whose direction is the desired character up vector is created. A second vector is also created with the same length, whose direction is negative 90° from the up vector. This pair of vectors may be transformed before being given to the metafile generator as the parameters to CHARACTER ORIENTATION. If the resultant vectors are not orthogonal, the text extent rectangle becomes a parallelogram, and the characters are skewed. If the vectors have different lengths, the aspect ratio derived from the font design and the character expansion attribute will be altered. If the positive angle from the up vector to the base vector is less than 180°, the following effects occur: characters are mirror imaged; and the "intuitive" notions of right and left (as applied to TEXT PATH and TEXT ALIGNMENT) are reversed, as described in 6.7.3.2.

References:

6.6.3
6.7.3.2
D.4.6

7.7.17 TEXT PATH

Parameters:

text path (one of: right, left, up, down) (E)

Description:

The text path is set to the value specified by the parameter. If the GENERALIZED TEXT PATH MODE is ‘off', subsequent text elements are displayed with this text path. See 6.6 for a list of text elements.

This function sets the value of the text path attribute, specifying the writing direction of a text string relative to the character up vector and character base vector. ‘Right' means in the direction of the character base vector. ‘Left' means 180° from the character base vector. ‘Up' means in the direction of the character up vector. ‘Down' means 180° from the character up vector.

References:

6.6.3
6.7.3.2
D.4.6

7.7.18 TEXT ALIGNMENT

Parameters:

horizontal alignment (one of: normal horizontal, left, centre, right, continuous horizontal) (E)
vertical alignment (one of: normal vertical, top, cap, half, base, bottom, continuous vertical) (E)
continuous horizontal alignment (R)
continuous vertical alignment (R)

Description:

The text alignment is set to the value specified by the parameters. Subsequent text strings are displayed with this text alignment.

The horizontal alignment type parameter is an enumerated data type with the possible values shown above. If its value is ‘continuous horizontal', the continuous horizontal alignment parameter (which is a fraction of the side of the text extent rectangle perpendicular to character up vector) becomes significant.

The vertical alignment type parameter is an enumerated data type with the possible values shown above. If the value is ‘continuous vertical', the continuous vertical alignment parameter (which is a fraction of the side of the text extent rectangle parallel to character up vector) becomes significant.

The "normal" parameters are dependent on the text path at the time of the elaboration of the text elements. See 6.6 for a list of text elements.

PATH	NORMAL HORIZONTAL	NORMAL VERTICAL
RIGHT	LEFT	BASELINE
LEFT	RIGHT	BASELINE
UP	CENTRE	BASELINE
DOWN	CENTRE	TOP

The continuous horizontal and vertical parameters may exceed the range of 0.0 to 1.0 in order to align a string with a coordinate outside its text extent rectangle.

References:

6.6.3
6.7.3.2
D.4.6

7.7.19 CHARACTER SET INDEX

Parameters:

character set index (IX)

Description:

The specified character set from the table specified in the CHARACTER SET LIST or GLYPH MAPPING Metafile Descriptor element becomes the currently designated G0 set. Since BEGIN PICTURE invokes the G0 set into positions 2/1 to 7/14 of the 7-bit or 8-bit code chart, the character set designated by CHARACTER SET INDEX is used to display the text in the text elements. See 6.6 for a list of text elements. The character set is used for the subsequent mapping of character codes to character glyphs.

Legal values of the *character set index* parameter are positive integers.

NOTE One use of this element is to switch among character sets for different languages.

References:

6.6.3
6.7.3.2
D.4.6

7.7.20 ALTERNATE CHARACTER SET INDEX

Parameters:

alternate character set index (IX)

Description:

The specified character set from the table specified in the CHARACTER SET LIST or GLYPH MAPPING Metafile Descriptor element becomes the currently designated G1 set and also the currently designated G2 set. Since BEGIN PICTURE invokes the G1 set into positions 10/1 to 15/14 (or 10/0 to 15/15 if the G1 set is a 96-character set), the character set designated by ALTERNATE CHARACTER SET INDEX is used to display 8-bit bytes whose most significant bit is set when those bytes occur within the string parameters of the text elements. This character set is used for the subsequent mapping of character codes to character glyphs.

NOTE SI and SO are only defined and usable with 7-bit coding.

Legal values of the *alternate character set index* parameter are positive integers.

If the appropriate CHARACTER CODING ANNOUNCER is selected, the SO and SI controls and ISO 2022 escape sequences may be embedded within the string parameters of text elements. If they are, the characters occurring after SO (SHIFT OUT) and before the SI (SHIFT IN) are displayed using the G1 character set: the same character set which is designated by ALTERNATE CHARACTER SET INDEX.

References:

6.6.3
6.7.3.2

7.7.21 FILL BUNDLE INDEX

Parameters:

fill bundle index (IX)

Description:

The fill bundle index is set to the value specified by the parameter. When subsequent filled-area elements occur, values for INTERIOR STYLE, FILL COLOUR, PATTERN INDEX, and HATCH INDEX are taken from the corresponding components of the indexed bundle, if the ASFs for these attributes are set to ‘bundled'. See 6.6 for a list of filled-area elements. If the ASF for a given attribute is ‘individual', this element does not affect the value used for that attribute until its ASF returns to ‘bundled'.

Legal values of the *fill bundle index* parameter are positive integers.

References:

6.6.4
6.7.4.1
6.7.4.3
D.4.6

7.7.22 INTERIOR STYLE

Parameters:

interior style (one of: hollow, solid, pattern, hatch, empty, geometric pattern, interpolated) (E)

Description:

The interior style of the filled-area elements is set to the value specified by the parameter. See 6.6 for a list of filled-area elements.

When the INTERIOR STYLE ASF is ‘bundled', this element does not affect the display of filled-area elements until the ASF returns to ‘individual'.

The interior fill style is used to determine in what style the area is to be filled. (See 6.7.4.3 for discussion of interior styles, and of extent of the interior and relationship of the interior to the edge.) Interior styles ‘geometric pattern' and ‘interpolated' are only supported by Version 3 metafiles.

References:

6.6.4
6.7.4.1
6.7.4.3
D.4.6

7.7.23 FILL COLOUR

Parameters:

fill colour specifier (CO)

Description:

The fill colour index or fill colour value is set as specified by the parameter.

When the FILL COLOUR ASF is ‘individual', subsequent filled-area elements are filled with this colour. See 6.6 for a list of filled-area elements.

When the FILL COLOUR ASF is ‘bundled', this element does not affect the display of these subsequent filled-area elements until the ASF returns to ‘individual'.

The fill colour attribute is significant only if INTERIOR STYLE is ‘hollow', ‘solid', ‘hatch', or 'geometric pattern'.

References:

6.6.4
6.7.4.1
6.7.4.3
6.7.6
D.3.2

7.7.24 HATCH INDEX

Parameters:

hatch index (IX)

Description:

The hatch index is set to the value specified by the parameter.

The following hatch indexes are assigned:

1: horizontal equally spaced parallel lines
2: vertical equally spaced parallel lines
3: positive slope equally spaced parallel lines
4: negative slope equally spaced parallel lines
5: horizontal/vertical crosshatch
6: positive slope/negative slope crosshatch

The ideal angle for the positive slope hatch patterns is +45°, and the ideal angle for the negative slope hatch patterns is +135° (see also annex D).

When the HATCH INDEX ASF is ‘individual' and the interior style is ‘hatch', subsequent filled-area elements are displayed using this hatch index. See 6.6 for a list of filled-area elements.

When the HATCH INDEX ASF is ‘bundled', this element does not affect the display of subsequent filled-area elements until the ASF returns to ‘individual'.

The fill colour attribute determines the colour of the hatch lines.

Values above 6 are reserved for registration and future standardization, and negative values are available for implementation-dependent use.

NOTE Hatch index values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a hatch index has been approved by ISO/IEC Subcommittee for *Computer Graphics*, the hatch index value will be assigned by the Registration Authority.

References:

6.6.4
6.7.4.1
6.7.4.3
D.4.6

7.7.25 PATTERN INDEX

Parameters:

pattern index (IX)

Description:

The pattern index is set to the value specified by the parameter.

When the PATTERN INDEX ASF is ‘individual' and the interior style is 'pattern', subsequent filled-area elements are displayed using this pattern index. See 6.6 for a list of filled-area elements. The pattern index is a pointer into the pattern table.

When the PATTERN INDEX ASF is ‘individual' and the interior style is ‘geometric pattern' subsequent filled-area elements are displayed using this pattern index. The pattern index is a pointer into the table of geometric patterns defined by GEOMETRIC PATTERN DEFINITION.

Legal values of the *pattern index* parameter are positive integers.

References:

6.6.4
6.7.4.1
6.7.4.3

7.7.26 EDGE BUNDLE INDEX

Parameters:

edge bundle index (IX)

Description:

The edge bundle index is set to the value specified by the parameter. For subsequent filled-area elements, values for EDGE TYPE, EDGE WIDTH, and EDGE COLOUR are taken from the corresponding components of the indexed bundle, if the ASFs for these attributes are set to ‘bundled'. See 6.6 for a list of filled-area elements. If the ASF for a given attribute is ‘individual', this element does not affect the value used for that attribute until its ASF returns to ‘bundled'.

Legal values of the *edge bundle index* parameter are positive integers.

References:

6.6.4
6.7.4.1
6.7.4.3
D.4.6

7.7.27 EDGE TYPE

Parameters:

edge type indicator (IX)

Description:

The edge type indicator is set to the value specified by the parameter.

When the EDGE TYPE ASF is ‘individual' and EDGE VISIBILITY is ‘on' the edges of filled-area elements are displayed with this edge line type. See 6.6 for a list of filled-area elements.

When the EDGE TYPE ASF is ‘bundled', this element does not affect the display of subsequent filled-area elements until the ASF returns to ‘individual'.

Edge type indicator has the same correspondence between type (for example, 4) and representation (dash-dot) as line type indicator. The following edge types are assigned:

1: solid
2: dash
3: dot
4: dash-dot
5: dash-dot-dot

Non-negative values of the index are reserved for standardized edge types, and negative values are available for implementation dependent use.

NOTE 1 Edge type continuity is addressed in Version 3 and Version 4 metafiles with the EDGE TYPE CONTINUATION element. In Version 1 and Version 2 metafiles, ideally the edge type is maintained continuously between adjacent spans of a single filled-area element (see annex D for further discussion). Continuity across edge sections that may have been clipped away or that have been declared invisible (in the case of POLYGON SET) is not addressed by this International Standard.

NOTE 2 Edge type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a edge type has been approved by ISO/IEC Subcommittee for *Computer Graphics*, the edge type value will be assigned by the Registration Authority.

References:

6.6.4
6.7.4.1
6.7.4.3
D.4.6

7.7.28 EDGE WIDTH

Parameters:

edge width specifier (SS)

Description:

The absolute edge width or edge width scale factor is set as specified by the parameter.

When the EDGE WIDTH ASF is ‘individual' and EDGE VISIBILITY is ‘on', the edge of filled-area elements are displayed with this width. See 6.6 for a list of filled-area elements.

When the EDGE WIDTH ASF is ‘bundled', this element does not affect the display of subsequent filled-area elements until the ASF returns to ‘individual'.

Valid values of the ‘edge width specifier’ parameter are non-negative VDC or R (according to the value of the EDGE WIDTH SPECIFICATION MODE element).

NOTE When a edge line is displayed, the edge width is measured perpendicular to the defining line (that is, it is independent of the orientation of the defining line). See annex D regarding alighment of the finite-width displayed edge with the zero-width defining line of the ideal edge.

References:

6.4.3
6.6.4
6.7.4.1
6.7.5
6.7.4.3
D.4.6

7.7.29 EDGE COLOUR

Parameters:

edge colour specifier (CO)

Description:

The edge colour index or edge colour value is set as specified by the parameter.

When the EDGE COLOUR ASF is ‘individual' and EDGE VISIBILITY is ‘on', the edge of filled-area elements are displayed with this colour. See 6.6 for a list of filled-area elements.

When the EDGE COLOUR ASF is ‘bundled', this element does not affect the interpretation of subsequent filled-area elements until the ASF returns to ‘individual'.

References:

6.6.4
6.7.4.1
6.7.6
6.7.4.3
D.3.2

7.7.30 EDGE VISIBILITY

Parameters:

edge visibility (one of: off,on) (E)

Description:

EDGE VISIBILITY specifies whether the edge of a filled-area element is displayed. This is independent of the display of the boundary, which is rendered when INTERIOR STYLE is ‘hollow'. See 6.6 for the distinction between the edge and the boundary of a filled-area element. See 6.6 for a list of filled-area elements.

The edge is never displayed if the current value is ‘off'. If the current value is ‘on' it is displayed for all primitives except POLYGON SET. For polygon set, individual edges are displayed if and only if the current value of EDGE VISIBILITY is on and the edge flag indicates a visible edge.

References:

6.6.4
6.7.4.3

7.7.31 FILL REFERENCE POINT

Parameters:

reference point (P)

Description:

The fill reference point is set to the value specified by the parameter.

When the currently selected interior style is ‘pattern' or ‘geometric pattern', this value is used in conjunction with pattern size for displaying filled-area primitives.

When the currently selected interior style is ‘hatch', the FILL REFERENCE POINT provides a common origin for the hatch patterns in all subsequent hatched filled areas.

When the currently selected interior style is ‘interpolated' the FILL REFERENCE POINT provides one of the reference points in the definition of the interior style.

The common origin for the interiors of filled areas means that separate filled areas, which have the same hatch index and which abut, have a visually continuous interior rendering across all of the filled areas.

References:

6.6.4
6.7.4.3

7.7.32 PATTERN TABLE

Parameters:

pattern table index (IX)
nx,ny (2I)
local colour precision (form depends upon specific encoding)
pattern colour specifiers (*nx · ny*CO)

Description:

The representation of the specified *pattern table index* is defined. The representation consists of an nx-by-ny array of colours. When INTERIOR STYLE is 'pattern', the pattern is mapped onto the interior of a filled-area element as described under the PATTERN SIZE element.

Legal values of the *pattern table index* parameter are positive integers.

The *local colour precision* parameter declares the precision of the *pattern colour specifiers*. The precision specification is represented in either 'indexed' or 'direct' colour mode, according to the current value of the COLOUR SELECTION MODE element. As with the COLOUR INDEX PRECISION and COLOUR PRECISION elements, the form of the parameter is encoding dependent. If the picture uses indexed colour selection, then the form of the parameter is the same as that of COLOUR INDEX PRECISION. If the picture uses direct colour selection, then the form of the parameter is the same as that of COLOUR PRECISION.

Legal values of local colour precision include the legal values of COLOUR (INDEX) PRECISION. In addition, each encoding defines a special value, the 'default colour precision indicator', as an indicator that the colour specifiers of the element are to be encoded in the COLOUR (INDEX) PRECISION of the metafile, i.e., to indicate that the local colour precision defaults to COLOUR (INDEX) PRECISION.

Changes to the representations of pattern table indexes have no effect on any previous graphical primitive elements that may have used the affected indexes.

References:

6.6.4
6.7.6
6.7.4.3

7.7.33 PATTERN SIZE

Parameters:

pattern size specifier (4SS)

Description:

The pattern size is set to the values specified by the parameter.

When the INTERIOR STYLE is set to 'pattern' or 'geometric pattern', subsequent filled-area elements are displayed using this pattern size. See 6.6 for a list of filled-area elements.

Pattern size comprises two vectors, a height vector and a width vector. The two vectors shall be non-collinear and shall have positive length.

In the general case the pattern size vectors and the FILL REFERENCE POINT define a parallelogram, the pattern box. When the interior style is 'pattern' this pattern box is divided into cells, nx in the width vector direction and ny in the height vector direction, where nx and ny are the colour array dimensions of the pattern table entry selected by the current pattern index. When the interior style is 'geometric pattern' the associated pattern extent rectangle is mapped onto the pattern box parallelogram.

The units in which the pattern size vectors are specified, as well as their behaviour and the behaviour of the rendered interior under transformations, is determined by the current value of the INTERIOR STYLE SPECIFICATION MODE.

When the interior style is 'pattern' the array of colours of the current pattern is mapped onto the array of cells as follows. The colour array element (1,ny) is mapped to the pattern box cell which is located at the FILL REFERENCE POINT. Colour array elements with increasing first dimension are associated with successive cells in the width direction, and colour array elements with decreasing second dimension are associated with successive cell in the direction of the height vector. In this way, each of the $nx \cdot ny$ colour array elements is associated with one of the $nx \cdot ny$ cells of the pattern box.Array element (1,1) corresponds to the first colour index or colour value stored in the *pattern colour specifiers* parameter (of the PATTERN TABLE element) and array element (nx,1) corresponds to the *nxth colour index* or colour value stored in the *pattern colour specifiers* parameter.

Conceptually, the pattern box so defined is replicated in directions parallel to the vectors of the PATTERN SIZE element until the interior of a filled-area element to which the pattern is to be applied is completely covered. The coincidence of this imposed pattern and the interior to which it is to be applied defines the interior style for the filled-area element being displayed.

References:

6.6.4
6.7.4.3
D.4.6

7.7.34 COLOUR TABLE

Parameters:

starting index (CI)
colour list (nCD)

Description:

The colour list elements are loaded, in the order specified, into the consecutive locations in the colour table beginning at the starting index. Only the specified colour table entries are changed. Changes in the colour table have no effect on any previous graphical primitive elements that use the affected indices. Setting a value for colour index zero using the COLOUR TABLE element sets the background colour if it occurs in the Picture Descriptor or METAFILE DEFAULTS REPLACEMENT. Definition of colour index 0 in the picture body has no effect on the background colour, only on subsequent primitives displayed with colour index 0 -- the background colour is determined and bound at the occurrence of BEGIN PICTURE BODY, if it occurs in the Picture Descriptor or METAFILE DEFAULTS REPLACEMENT. Definition of colour index 0 in the picture body has no effect on the background colour, only on subsequent primitives displayed with colour index 0 — the background colour is determined and bound at the occurrence of BEGIN PICTURE BODY.

Legal values of the colour index are non-negative integers.

Changes to the representations of colour table indexes have no effect on any previous graphical primitive elements that may have used the affected indexes.

References:

6.7.6
6.7.4.3

7.7.35 ASPECT SOURCE FLAGS

Parameters:

list of: pairs of
ASF type, ASF value (one of: individual, bundled) n[E,E]

Description:

The designated Aspect Source Flags (ASFs) are set to the values indicated by the parameter. The following ASF types are assigned:

line type ASF
line width ASF
line colour ASF
marker type ASF
marker size ASF
marker colour ASF
text font index ASF
text precision ASF
character expansion factor ASF
character spacing ASF
text colour ASF
interior style ASF
fill colour ASF
hatch index ASF
pattern index ASF
edge type ASF
edge width ASF
edge colour ASF

The Aspect Source Flags determine the attribute values that will be bound to a primitive. If the ASF for a particular aspect of a primitive is set to 'individual', the value used is the value of the corresponding individually specified attribute of the primitive. If the ASF is set to 'bundled', the value used is the value of the corresponding aspect of the bundle pointed to by the current bundle index for the primitive.

ASFs are modally bound to primitives just as are other primitive attributes. Thus changing the value of an ASF within a picture will have no retroactive effect on any previous graphical primitive element.

References:

6.7
D.4.6

7.7.36 PICK IDENTIFIER

Parameters:

pick identifier (N)

Description:

The pick identifier value is associated with all of the graphical primitive elements of a segment until the next PICK IDENTIFIER element. Usage of the PICK IDENTIFIER on interpretation is dependent upon the application.

References:

6.7.7

7.7.37 LINE CAP

Parameters:

line cap indicator (IX)
dash cap indicator (IX)

Description:

The line cap and dash cap styles are defined for subsequent line elements. The *line cap indicator* determines the appearance of open endpoints (as opposed to interior vertices) of line elements. The following values are defined:

1: unspecified — no specific treatment is required.
2: butt — the line is squared off at the endpoint, there is no projection beyond the endpoint.
3: round — a semicircular arc with diameter equal to the line width is drawn around the endpoint and filled in. The drawn line thus projects beyond the endpoint.
4: projecting square — the line is squared off at a distance equal to half the line width beyond the endpoint.
5: triangle — a cap is added to the line which is an equilateral triangle whose side equals the line width.

Legal values of the *line cap indicator* parameter are positive integers. Values greater than 5 are reserved for future standardization and registration.

The dash cap indicator determines the appearance of the endpoints of individual dashs for subsequent dashed lines. The dash cap indicator applies to all endpoints of dashes within the lines except the open endpoints which have their style defined by the line cap indicator. The following values are allowed:

1: unspecified — no specific treatment is required.
2: butt — the dash is squared off at the endpoint, there is no projection beyond the endpoint.
3: match — the endpoints of the dashes have the style defined by the line cap indicator.

Legal values of the *dash cap indicator* parameter are positive integers. Values greater than 3 are reserved for future standardization and registration.

This element is not permitted in Version 1 and Version 2 metafiles; Version 1 and Version 2 metafiles support only ‘unspecified' for both line cap and dash cap indicators.

NOTE Line cap and dash cap values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a line cap or dash cap has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the line cap value or dash cap value will be assigned by the Registration Authority.

References:

6.7.1.2

7.7.38 LINE JOIN

Parameters:

line join indicator (IX)

Description:

The line join style is defined for subsequent line elements. The line join style defines the appearance of interior vertices of individual line elements as well as the junctions between successive individual line elements in compound line elements. The defined values are:

1: unspecified — no specific treatment is required.
2: mitre — the outer edges of the two adjoining line segments are extended until they meet at a point.
3: round — a circular arc with diameter equal to the line width is drawn around the vertex between the adjoining segments and is filled in, producing a rounded corner.
4: bevel — the adjoining line segments are terminated with a butt cap, and the resulting triangular notch is filled in.

Legal values of the *line join indicator* parameter are positive integers. Values greater than 4 are reserved for future standardization and registration.

This element is not permitted in Version 1 and Version 2 metafiles; Version 1 and Version 2 metafiles support only ‘unspecified'.

NOTE Line join indicator values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a line join has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the line join indicator value will be assigned by the Registration Authority.

References:

6.7.1.2

7.7.39 LINE TYPE CONTINUATION

Parameters:

continuation mode (IX)

Description:

The behaviour of dashed line patterns at the vertices of individual line elements and the junctions between successive individual line elements in compound line elements is determined. The following values are defined:

1: unspecified — no specific treatment is required.
2: continue — the style is continued without interruption across vertices.
3: restart — the style is restarted at each vertex.
4: adaptive continue — the style is continued, but each vertex shall be "inked".

Legal values of the *continuation mode* parameter are positive integers. Values greater than 4 are reserved for future standardization and registration.

This element is not permitted in Version 1 and 2 metafiles; Version 1 and Version 2 metafiles support only ‘unspecified'.

The value ‘adaptive continue' requires that each vertex contains a drawn portion of the pattern. This may require the pattern to be stretched or compressed. For this style the initial and final points of the line are included in those which shall be "inked".

NOTE 1 The styles 'restart' and 'adaptive continue' are likely to yield poor results if the drawn lines consist of many short segments.

NOTE 2 Line type continuation values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a line type continuation has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the line type continuation value will be assigned by the Registration Authority.

References:

6.7.1.2

7.7.40 LINE TYPE INITIAL OFFSET

Parameters:

line pattern offset (R)

Description:

The value of the *line pattern offset* indicates how far into the current line pattern definition to start when drawing of a dashed line is begun.

Valid values are real numbers between zero and 1.

References:

6.7.1.2

7.7.41 TEXT SCORE TYPE

Parameters:

list of pairs (score type, score indicator) (n[IX,E])

Description:

The following values are defined for score type:

1: right score (equivalent to underscore in left-to-right writing mode);
2: left score (equivalent to overscore in left-to-right writing mode);
3: through score (equivalent to strikeout in left-to-right writing mode);
4: kendot (emphasis similar to underscore for Kanji)

Legal values of the *score type* parameter are positive integers. Values greater than 4 are reserved for registration and future standardization

The score indicator may be either ‘off' or ‘on'. The value ‘off' indicates that the corresponding score type is not used. The value ‘on' indicates that the corresponding score is used.

Any combination of score types may be active simultaneously.

NOTE Score type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a score type has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the score type value will be assigned by the Registration Authority.

References:

6.7.3.2

7.7.42 RESTRICTED TEXT TYPE

Parameters:

restriction type (IX)

Description:

RESTRICTED TEXT constrains text strings to be within a parallelogram. This attribute selects one of a number of ways of applying the restriction to the text string. The defined values of the *restriction type* parameter are:

1: basic;
2: boxed-cap;
3: boxed-all;
4: isotropic-cap;
5: isotropic-all;
6: justified.

Legal values of the *restriction type* parameter are positive integers. Values greater than 6 are reserved for future standardization and registration.

The effects of these values are described in clause 6.

This element is not permitted in Version 1 and Version 2 metafiles; Version 1 and Version 2 metafiles support only ‘basic'.

NOTE Restriction type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a restriction type has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the restriction type value will be assigned by the Registration Authority.

References:

6.6.3.2
6.7.3.2

7.7.43 INTERPOLATED INTERIOR

Parameters:

style (IX)
reference geometry (2nSS)
number of stages (I)
list of stage designators (mR)

if style is 'parallel' or 'elliptical',
list of reference colour specifiers ((m+1)CO)

if style is 'triangular',
list of reference colour specifiers (3CO)

Description:

The *style* parameter selects the way of defining the coloured plane. The following values are defined:

1: parallel;
2: elliptical;
3: triangular.

Legal values of the *style* parameter are positive integers. Values greater than 3 are reserved for future standardization and registration.

The geometry of the shaded plane is defined relative to the FILL REFERENCE POINT. The scalars of *reference geometry* are applied as follows:

parallel: the number of scalars shall be 2. The FILL REFERENCE POINT is one defining point of a reference line. A second defining point of the reference line is defined by the 2 scalars, which are respectively the x and y offset of the second point from the FILL REFERENCE POINT.

elliptical: the number of scalars shall be 4. The FILL REFERENCE POINT is the centre of a reference ellipse. The first pair of scalars are respectively the x and y offset from the FILL REFERENCE POINT to the first CDP of ellipse and the second pair are respectively the x and y offset from the FILL REFERENCE POINT to the second CDP of ellipse.

triangular: the number of scalars shall be 4. The first pair of scalars are respectively the x and y offset from the FILL REFERENCE POINT to the second corner of a reference triangle and the second pair are respectively the x and y offset from the FILL

REFERENCE POINT to the third corner of the reference triangle. The *number of stages* shall be 0 and the list of stage designators shall be empty.

Valid values of the scalars of the *reference geometry* are those which produce distinct (non-coincident) geometry reference points when applied as described above.

When the value of the *style* parameter is ‘parallel' or ‘elliptical', one or more bands of parallel or concentric interpolated colours are defined as follows.

Delimiting points dividing adjacent interpolation stages are defined along the line through the FILL REFERENCE POINT and the first reference geometry point. One delimiter of the first stage (there shall be at least one stage) is the FILL REFERENCE POINT. If the line through the FILL REFERENCE POINT and the first reference geometry point is designated **L**, the distance from the FILL REFERENCE POINT to the first reference geometry point is designated *d*, and the *i*-th stage designator is denoted S_i, then additional stage delimiters are defined as follows. The *i*-th stage delimiter is located on the line **L** at a distance $d{\cdot}S_i$ from the FILL REFERENCE POINT. The S_i shall be positive and in increasing order.

The colours are assigned to the stage delimiters in order, and linearly interpolated across bands that are defined by adjacent stage delimiters. In the case of the parallel style, the colours in the plane are constant on lines perpendicular to **L**, and on such lines have colour equal to the colour at the intersection with **L**. In the case of the elliptical style, the colours in the plane are constant on ellipses concentric with the reference ellipse and on such ellipses have colour equal to the colour at the intersection with **L**.

For the triangular style, the first reference colour is applied at the FILL REFERENCE POINT, which is the first corner of the interpolated triangle. The second reference colour is applied to the second corner of the triangle and the third reference colour is applied to the third corner of the triangle. The interpolation is defined on these three colour-tagged points as defined in clause 6.

NOTE Style values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a style has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the style value will be assigned by the Registration Authority.

References:

6.7.4.3

7.7.44 EDGE CAP

Parameters:

edge cap indicator (IX)
dash cap indicator (IX)

Description:

The edge cap and dash cap styles are defined for subsequent line elements. The value of the *edge cap indicator* parameter determines the appearance of open endpoints (as opposed to interior vertices) of line elements. The following values are defined:

1: unspecified — no specific treatment is required.
2: butt — the edge is squared off at the endpoint, there is no projection beyond the endpoint.
3: round — a semicircular arc with diameter equal to the edge width is drawn around the endpoint and filled in. The drawn edge thus projects beyond the endpoint.
4: projecting square — the edge is squared off at a distance equal to half the edge width beyond the endpoint.
5: triangle — a cap is added to the edge which is an equilateral triangle whose side equals the edge width.

Legal values of 1,…,5 are positive integers. Values greater than 5 are reserved for future standardization and registration.

The value of the *dash cap indicator* parameter determines the appearance of the endpoints of individual dashs for subsequent dashed edges. The dash cap indicator applies to all endpoints of dashes within the edges except the open endpoints which have their style defined by the edge cap indicator. The following values are defined:

1: unspecified — no specific treatment is required.
2: butt — the dash is squared off at the endpoint, there is no projection beyond the endpoint.
3: match — the endpoints of the dashes have the style defined by the edge cap indicator.

Legal values of the *dash cap* parameter are positive integers. Values greater than 3 are reserved for future standardization and registration.

This element is not permitted in Version 1 and Version 2 metafiles; Version 1 and Version 2 metafiles support only ‘unspecified' for both edge cap and dash cap indicators.

NOTE Edge cap and dash cap values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When an edge cap or dash cap has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the edge cap value or dash cap value will be assigned by the Registration Authority.

References:

6.7.4.4

7.7.45 EDGE JOIN

Parameters:

edge join indicator (IX)

Description:

The edge join style is defined for subsequent filled-area elements. The edge join style defines the appearance of interior vertices between individual edge segments of filled-area elements, as well as the appearance of edges at junctions between successive individual line and filled-area elements in compound filled-area elements. The defined values are:

1: unspecified — no specific treatment is required.
2: mitre — the outer edges of the two adjoining edge segments are extended until they meet at a point.
3: round — a circular arc with diameter equal to the edge width is drawn around the vertex between the adjoining segments and is filled in, producing a rounded corner.
4: bevel — the adjoining edge segments are terminated with a butt cap, and the resulting triangular notch is filled in.

Legal values of the *edge join indicator* parameter are positive integers. Values greater than 4 are reserved for future standardization and registration.

This element is not permitted in Version 1 and Version 2 metafiles; Version 1 and Version 2 metafiles support only ‘unspecified'.

NOTE Edge join values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a edge join has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the edge join value will be assigned by the Registration Authority.

References:

6.7.4.4

7.7.46 EDGE TYPE CONTINUATION

Parameters:

continuation mode (IX)

Description:

The behaviour of dashed edge patterns at the vertices of individual edges of filled-area elements and the junctions between successive individual line and filled-area elements in compound filled-area elements is determined. The following standardized values are defined:

1: unspecified — no specific treatment is required.
2: continue — the style is continued without interruption across vertices.
3: restart — the style is restarted at each vertex.
4: adaptive continue — the style is continued, but each vertex shall be "inked".

Legal values of the *continuation mode* parameter are positive integers. Values greater than 4 are reserved for future standardization and registration.

The value ‘adaptive continue' requires that each vertex contains a drawn portion of the pattern. This may require the pattern to be stretched or compressed. For this style the initial and final points of the line are included in those which shall be "inked".

This element is not permitted in Version 1 and Version 2 metafiles; Version 1 and Version 2 metafiles support only ‘unspecified'.

NOTE 1 The styles ‘restart’ and ‘adaptive continue' are likely to yield poor results if the drawn lines consist of many short segments.

NOTE 2 Edge type continuation values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When an edge type continuation has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the edge type continuation value will be assigned by the Registration Authority.

References:

6.7.4.4

7.7.47 EDGE TYPE INITIAL OFFSET

Parameters:

edge pattern offset (R)

Description:

The value of the *edge pattern offset* indicates how far into the current edge pattern definition to start when drawing of a dashed edge is begun.

Valid values are real numbers between zero and 1.

References:

6.7.4.4

7.7.48 SYMBOL LIBRARY INDEX

Parameters:

symbol library index (IX)

Description:

The value of the *symbol library index* parameter selects the symbol library to be used for subsequent POLYSYMBOL elements. The symbol library index selects a symbol library from the symbol library list defined in the Metafile Descriptor.

Legal values of the symbol library index parameter are positive integers.

References:

6.6.12

7.7.49 SYMBOL COLOUR

Parameters:

symbol colour specifier (CO)

Description:

The symbol colour index or symbol colour value is set as specified by the parameter.

NOTE Colour may be an aspect of a symbol's definition in the symbol library. Annex D gives recommendations on how to handle SYMBOL COLOUR when the symbol itself contains colour.

References:

6.6.12
D.4.6

7.7.50 SYMBOL SIZE

Parameters:

scale indicator (one of: height, width, both) (E)
symbol height (VDC)
symbol width (VDC)

Description:

The value of the *scale indicator* parameter determines whether the symbol design height is scaled while preserving aspect ratio, or the symbol design width is scaled while preserving aspect ratio, or both are scaled to the values of the *symbol height* and *symbol width* parameters simultaneously.

The *symbol height* and *symbol width* parameters define the value of symbol height and width for subsequently occurring symbols. See 6.6.12 for a list of symbol elements, as well as a description of the use of the symbol attributes by generators and interpreters.

Valid values of symbol height and symbol width are positive VDC.

References:

6.6.12

7.7.51 SYMBOL ORIENTATION

Parameters:

symbol up, x component (VDC)
symbol up, y component (VDC)
symbol base, x component (VDC)
symbol base, y component (VDC)

Description:

Two vectors are defined which determine the orientation and skew of symbols in subsequent symbol elements. See 6.6.12 for a list of symbol elements, as well as a description of how symbols are sized and oriented for display.

The two vectors shall be non-collinear and shall have positive length.

References:

6.6.12

7.8 Escape elements

7.8.1 ESCAPE

Parameters:

function identifier (I)
data record (D)

Description:

ESCAPE provides access to device capabilities not specified by ISO/IEC 8632. The function identifier parameter specifies the particular escape function. Non-negative values are reserved for registration and future standardization, and negative values are available for implementation dependent use.

See D.4.7 regarding the format of the *data record* parameter.

NOTE 1 ESCAPE is designed for access to non-standardized attribute and control functions, whereas the GENERALIZED DRAWING PRIMITIVE is designed for access to non-standardized geometric primitives.

NOTE 2 Function identifiers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a function identifier has been approved by ISO/IEC Sub-committee for *Computer Graphics*, the function identifier value will be assigned by the Registration Authority.

References:

6.8
D.4.7

7.9 External elements

7.9.1 MESSAGE

Parameters:

action required flag (one of: no action, action) (E)
text (SF)

Description:

The MESSAGE element specifies a string of characters used to communicate information to operators at Metafile interpretation time through a path separate from normal graphical output.

If the value of the *action required flag* parameter is ‘action', the metafile interpreter may need to pause to wait for an operator response. Because the message and an associated pause may be directed at a particular device, only the interpreter may determine if a pause is appropriate. Character set selection for the text parameter is independent of any character set selection specified by this standard.

References:

6.9

7.9.2 APPLICATION DATA

Parameters:

identifier (I)
data record (D)

Description:

This element supplements the information in the metafile in an application-dependent way. It has no effect on the picture generated by interpreting the metafile, or on the states of the metafile generator or interpreter.

The content of the *identifier* and *data record* parameters is not standardized.

See D.4.8 regarding the format of the *data record* parameter.

The contents of the data record may include such information as history data associated with pictures, description of algorithms used, etc. The element is non-graphical in the sense that a fully capable interpreter which did not understand the meaning of APPLICATION DATA elements in a metafile should be able to produce the same picture as an interpreter which did understand the elements.

References:

6.9
D.4.8

7.10 Segment elements

7.10.1 COPY SEGMENT

Parameters:

segment identifier (N)
copy transformation matrix:
    scaling and rotation portion (2 x 2) (R)
    translation portion (2 x 1) (VDC)
segment transformation application (one of: no, yes) (E)

Description:

The segment which is indicated by the segment identifier is referenced at this point in the metafile for copying into the picture, or into a segment when referenced from a segment, on interpretation. The identified segment is referred to as the copied segment. With the possible exception of the segment transformation associated with the copied segment, the segment attributes of the copied segment are ignored. The segment attributes of a segment in which the COPY SEGMENT may occur are unchanged by this element.

The copy transformation is applied to all graphic objects of the copied segment before they are copied into the picture or into the segment. The copy transformation is also applied to clipping rectangles under some circumstances.

The INHERITANCE FILTER element allows for control of the control and attribute values which are used when copying segments. This filter controls whether values of individual attribute and control elements are reapplied to the graphic objects. The effects of INHERITANCE FILTER are described in 6.10.5. The way in which clipping is applied to primitives within a copied segment is controlled by CLIP INHERITANCE (see 6.10.5).

The 'segment transformation application' parameter controls whether or not the segment transformation associated with the copied segment will be applied as an effect of the copy process. In no case is the segment transformation applied to a clip rectangle associated with a copied graphic object. In case the *segment transformation application* is 'yes', the segment transformation is applied prior to the copy transformation.

References:

6.10.1
6.10.5

7.10.2 INHERITANCE FILTER

Parameters:

filter selection list (list of elements or groups from:

LINE BUNDLE INDEX
LINE TYPE
LINE WIDTH
LINE COLOUR
LINE CLIPPING MODE
MARKER BUNDLE INDEX
MARKER TYPE
MARKER SIZE
MARKER COLOUR
MARKER CLIPPING MODE
TEXT BUNDLE INDEX
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOUR
CHARACTER HEIGHT
CHARACTER ORIENTATION
TEXT PATH
TEXT ALIGNMENT
FILL BUNDLE INDEX
INTERIOR STYLE
FILL COLOUR
HATCH INDEX
PATTERN INDEX
EDGE BUNDLE INDEX
EDGE TYPE
EDGE WIDTH
EDGE COLOUR
EDGE VISIBILITY
EDGE CLIPPING MODE
FILL REFERENCE POINT
PATTERN SIZE
AUXILIARY COLOUR
TRANSPARENCY
LINE ATTRIBUTES
MARKER ATTRIBUTES
TEXT REPRESENTATION AND PLACEMENT ATTRIBUTES
TEXT PLACEMENT AND ORIENTATION ATTRIBUTES
FILL ATTRIBUTES
EDGE ATTRIBUTES

ALL
LINE TYPE ASF
LINE WIDTH ASF
LINE COLOUR ASF
MARKER TYPE ASF
MARKER SIZE ASF
MARKER COLOUR ASF
TEXT FONT INDEX ASF
TEXT PRECISION ASF
CHARACTER EXPANSION FACTOR ASF
CHARACTER SPACING ASF
TEXT COLOUR ASF
INTERIOR STYLE ASF
FILL COLOUR ASF
HATCH INDEX ASF
PATTERN INDEX ASF
EDGE TYPE ASF
EDGE WIDTH ASF
EDGE COLOUR ASF
LINE ASF
MARKER ASF
TEXT ASF
FILL ASF
EDGE ASF
ALL ASFS
MITRE LIMIT
LINE CAP
LINE JOIN
LINE TYPE CONTINUATION
LINE TYPE INITIAL OFFSET
TEXT SCORE TYPE
RESTRICTED TEXT TYPE
INTERPOLATED INTERIOR
EDGE CAP
EDGE JOIN
EDGE TYPE CONTINUATION
EDGE TYPE INITIAL OFFSET

SYMBOL LIBRARY INDEX

PATTERN ATTRIBUTES
OUTPUT CONTROL
PICK IDENTIFIER
ALL ATTRIBUTES AND CONTROL
SYMBOL COLOUR
SYMBOL SIZE
SYMBOL ORIENTATION
SYMBOL ATTRIBUTES (nE)

selection setting (one of: state list, segment) (E)

Description:

The setting of the inheritance filter is modified for those attributes in the filter selection list. Attributes may be inherited from the modal state lists or from the copied segment depending on the selection setting.

References:

6.10.5

7.10.3 CLIP INHERITANCE

Parameters:

clip inheritance (one of: state list, intersection) (E)

Description:

The behaviour of clipping as applied to graphic objects in copied segments is defined. Simple clipping against the current rectangle in the modal state list is selected by the value ‘state list'. The value ‘intersection' not only selects the clip rectangle to come from the segment but also enables an "object clipping" feature. The transformation of clip rectangles and accumulation or composition of multiple transformed rectangles is enabled, depending upon the settings of CLIP INDICATOR (see 6.10.5).

The description of clipping in this subclause also applies to the protection region used for clipping and shielding of arbitrary areas.

References:

6.10.5

7.10.4 SEGMENT TRANSFORMATION

Parameters:

segment identifier (N)
transformation matrix:

scaling and rotation portion (2 x 2) (R)
translation portion (2 x 1) (VDC)

Description:

The transformation matrix for the identified segment is set to the value specified by the *transformation matrix* parameter.

SEGMENT TRANSFORMATION, if used, shall appear after BEGIN SEGMENT, and before the first element of any type other than another Segment Attribute element. The value of the *segment identifier* parameter shall be identical to the identifier of the segment in which the element occurs except when the element appears in the Metafile Defaults Replacement element in which case the segment identifier is ignored.

References:

6.10.4.2

7.10.5 SEGMENT HIGHLIGHTING

Parameters:

segment identifier (N)
highlighting (one of: normal, highlighted) (E)

Description:

The value of the *segment highlighting* parameter defines the highlighting attribute for the identified segment. When the highlighting attribute is set to 'highlighted', the visual appearance of the segment is interpretation dependent. When the highlighting attribute is set to 'normal', the segment is displayed according to the segment and primitive attributes.

SEGMENT HIGHLIGHTING, if used, shall appear after BEGIN SEGMENT, and before the first element of any type other than another Segment Attribute element. The value of the *segment identifier* parameter shall be identical to the identifier of the segment in which the element occurs except when the element appears in the Metafile Defaults Replacement element in which case the segment identifier is ignored.

References:

6.10.4.3

7.10.6 SEGMENT DISPLAY PRIORITY

Parameters:

segment identifier (N)
segment display priority (I)

Description:

The display priority for the identified segment is defined by the value of the *segment display priority* parameter.

Segments with higher segment display priority appear to be in front of segments with lower segment display priorities when displayed following interpretation. When the segment display priorities of two overlapping segments are the same, then the segment definition or copy which occurs later in the file has higher priority than the one earlier in the file.

SEGMENT DISPLAY PRIORITY, if used, shall appear after BEGIN SEGMENT, and before the first element of any type other than another Segment Attribute element. The value of the *segment identifier* parameter shall be identical to the identifier of the segment in which the element occurs except when the element appears in the Metafile Defaults Replacement element in which case the segment identifier is ignored.

References:

6.10.4.4

7.10.7 SEGMENT PICK PRIORITY

Parameters:

a segment identifier (N)
segment pick priority (I)

Description:

The pick priority for the identified segment is defined by the value of the *segment pick priority* parameter. The pick priority does not affect the display of segments.

SEGMENT PICK PRIORITY, if used, shall appear after BEGIN SEGMENT, and before the first element of any type other than another Segment Attribute element. The value of the *segment identifier* parameter shall be identical to the identifier of the segment in which the element occurs except when the element appears in the Metafile Defaults Replacement element in which case the segment identifier is ignored.

References:

6.10.4.5

7.11 Application structure descriptor elements

7.11.1 APPLICATION STRUCTURE ATTRIBUTE

Parameters:

application structure attribute type (SF)
data record (SDR)

Description:

This element supplements the information in the metafile in an application-specific way. The Application Structure Descriptor section of an application structure comprises zero or more instances of only this element. It is intended that this element provide a mechanism to attach properties to an application structure definition. The application structure attribute type parameter may be used to classify attributes into groups meaningful to the application. The data record values contain application-specific data.

References:

6.13

8 Metafile defaults

This clause contains the Metafile default values that are used for those default values not explicitly set in the METAFILE DEFAULTS REPLACEMENTS element. The default values of some elements are dependent upon the values of other elements (for example, default CHARACTER HEIGHT is dependent upon VDC EXTENT). In these cases, the default of the dependent element is tied to the default of the other element, whether the latter is as defined in the table below or is defined with the METAFILE DEFAULTS REPLACEMENT elements. The value of the dependent element does not, however, change when the value of the element upon which it depends is changed explicitly by a metafile element. Rather, the value of the dependent element remains unchanged, in its default state, until explicitly changed by the occurrence of the element. See 7.3.12 for further discussion of element defaults and the action of METAFILE DEFAULTS REPLACEMENT .

VDC TYPE:	integer
INTEGER PRECISION:	encoding dependent
REAL PRECISION:	encoding dependent
INDEX PRECISION:	encoding dependent
COLOUR PRECISION:	encoding dependent
COLOUR INDEX PRECISION:	encoding dependent
MAXIMUM COLOUR INDEX:	63
COLOUR VALUE EXTENT:	encoding dependent
METAFILE ELEMENT LIST:	n/a
METAFILE DEFAULTS REPLACEMENT:	n/a
FONT LIST:	for index 1, any font that can represent the nationality-independent subset of ISO/IEC 646 which is the default for CHARACTER SET LIST described below
CHARACTER SET LIST:	for index 1, any character set which includes the nationality-independent subset of ISO/IEC 646 in the positions specified in ISO/IEC 646
CHARACTER CODING ANNOUNCER:	basic 7-bit
NAME PRECISION:	encoding dependent
MAXIMUM VDC EXTENT:	default VDC EXTENT
SEGMENT PRIORITY EXTENT:	0..255
COLOUR MODEL:	1 (RGB)
COLOUR CALIBRATION:	1 (unspecified); n/a for all other parameters
FONT PROPERTIES:	n/a
GLYPH MAPPING:	n/a
SYMBOL LIBRARY LIST:	n/a
SCALING MODE:	abstract; metric scale factor n/a
COLOUR SELECTION MODE:	indexed
LINE WIDTH SPECIFICATION MODE:	scaled
MARKER SIZE SPECIFICATION MODE:	scaled
EDGE WIDTH SPECIFICATION MODE:	scaled
VDC EXTENT:	if VDC TYPE is integer, lower left (0,0), upper right (32767,32767); if VDC TYPE is real, lower left (0.0,0.0), upper right (1.0,1.0)

BACKGROUND COLOUR:	device-dependent background colour
VDC INTEGER PRECISION:	encoding dependent
VDC REAL PRECISION:	encoding dependent
AUXILIARY COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 0; if COLOUR SELECTION MODE is ‘direct', device-dependent background colour
TRANSPARENCY:	on
DEVICE VIEWPORT:	0.0,0.0,1.0,1.0
DEVICE VIEWPORT SPECIFICATION MODE:	fraction of display surface
DEVICE VIEWPORT MAPPING:	forced,left,bottom
LINE REPRESENTATION:	interpreter dependent
MARKER REPRESENTATION:	interpreter dependent
TEXT REPRESENTATION:	interpreter dependent
FILL REPRESENTATION:	interpreter dependent
EDGE REPRESENTATION:	interpreter dependent
INTERIOR STYLE SPECIFICATION MODE:	absolute
GEOMETRIC PATTERN DEFINITION:	n/a
CLIP RECTANGLE:	default VDC EXTENT
CLIP INDICATOR:	on
LINE CLIPPING MODE:	interpreter dependent
MARKER CLIPPING MODE:	interpreter dependent
EDGE CLIPPING MODE:	interpreter dependent
PROTECTION REGION INDICATOR:	1; off (also, whenever a region is defined, its associated protection region indicator assumes an initial default value of ‘off')
Protection region:	1 region identical to default CLIP RECTANGLE
GENERALIZED TEXT PATH MODE:	off Compound text path: the line from lower-left to lower-right corner of the default VDC EXTENT
MITRE LIMIT:	32767.0
TRANSPARENT CELL COLOUR :	‘off'; n/a
LINE BUNDLE INDEX:	1
LINE TYPE:	1 (solid)
LINE WIDTH:	if LINE WIDTH SPECIFICATION MODE is ‘absolute', 1/1000 of the longest side of the rectangle defined by default VDC EXTENT; if LINE WIDTH SPECIFICATION MODE is ‘scaled', 1.0; if LINE WIDTH SPECIFICATION MODE is ‘fractional', 0.001; if LINE WIDTH SPECIFICATION MODE is ‘mm', 0.35
LINE COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 1; if COLOUR SELECTION MODE is ‘direct', device-dependent foreground colour
MARKER BUNDLE INDEX:	1
MARKER TYPE:	3 (asterisk)

MARKER SIZE:	if MARKER SIZE SPECIFICATION MODE is ‘absolute', 1/100 of the longest side of the rectangle defined by default VDC EXTENT; if MARKER SIZE SPECIFICATION MODE is ‘scaled', 1.0; if MARKER SIZE SPECIFICATION MODE is ‘fractional', 0.01; if MARKER SIZE SPECIFICATION MODE is ‘mm', 2.50
MARKER COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 1; if COLOUR SELECTION MODE is ‘direct', device-dependent foreground colour
TEXT BUNDLE INDEX:	1
TEXT FONT INDEX:	1
TEXT PRECISION:	string
CHARACTER EXPANSION FACTOR:	1.0
CHARACTER SPACING:	0.0
TEXT COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 1; if COLOUR SELECTION MODE is ‘direct', device-dependent foreground colour
CHARACTER HEIGHT:	1/100 of the length of the longest side of the rectangle defined by default VDC extent
CHARACTER ORIENTATION:	0,dy,dy,0, where dy is the height of the rectangle defined by the default VDC extent
TEXT PATH:	right
TEXT ALIGNMENT:	normal horizontal, normal vertical
CHARACTER SET INDEX:	1
ALTERNATE CHARACTER SET INDEX:	1
FILL BUNDLE INDEX:	1
INTERIOR STYLE:	hollow
FILL COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 1; if COLOUR SELECTION MODE is ‘direct', device-dependent foreground colour
HATCH INDEX:	1
PATTERN INDEX:	1
EDGE BUNDLE INDEX:	1
EDGE TYPE:	1 (solid)
EDGE WIDTH:	if EDGE WIDTH SPECIFICATION MODE is ‘absolute', 1/1000 of the longest side of the rectangle defined by default VDC EXTENT; if EDGE WIDTH SPECIFICATION MODE is ‘scaled', 1.0; if EDGE WIDTH SPECIFICATION MODE is ‘fractional', 0.001; if EDGE WIDTH SPECIFICATION MODE is ‘mm', 0.35
EDGE COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 1; if COLOUR SELECTION MODE is ‘direct', device-dependent foreground colour
EDGE VISIBILITY:	off
FILL REFERENCE POINT:	lower-left corner point of default VDC extent
PATTERN TABLE:	1;

	nx=ny=1; local colour precision is the (encoding-dependent) ‘default colour precision indicator'; if COLOUR SELECTION MODE is ‘indexed', the default colour specification is 1; if COLOUR SELECTION MODE is ‘direct', the default colour specification is device-dependent foreground colour
PATTERN SIZE:	0,dy,dx,0, where depending upon the value of INTERIOR STYLE SPECIFICATION MODE dy and dx are respectively: — height and width of default VDC extent if ‘absolute'; — height and width of some device-dependent "nominal" if ‘scaled'; — 0.0,1.0,1.0,0.0 if ‘fractional'; — 0.0,1.0,1.0,0.0 if ‘mm';
COLOUR TABLE:	device-dependent background colour for index = 0; device-dependent foreground colours for indexes greater than 0
ASPECT SOURCE FLAGS:	all individual
PICK IDENTIFIER:	0
LINE CAP:	1 (unspecified); 1 (unspecified)
LINE JOIN:	1 (unspecified)
LINE TYPE CONTINUATION:	1 (unspecified)
LINE TYPE INITIAL OFFSET:	0.0
TEXT SCORE TYPE:	all text scores are ‘off'
RESTRICTED TEXT TYPE:	1 (basic)
INTERPOLATED INTERIOR:	style ‘parallel'; reference points default VDC extent; number of stages 1; list of stage designators 1.0; reference colours device-dependent background colour if COLOUR SELECTION MODE is ‘direct', 0 if COLOUR SELECTION MODE is ‘indexed'
EDGE CAP:	1 (unspecified); 1 (unspecified)
EDGE JOIN:	1 (unspecified)
EDGE TYPE CONTINUATION:	1 (unspecified)
EDGE TYPE INITIAL OFFSET:	0.0
SYMBOL LIBRARY INDEX:	n/a
SYMBOL COLOUR:	if COLOUR SELECTION MODE is ‘indexed', 1; if COLOUR SELECTION MODE is ‘direct', device-dependent foreground colour
SYMBOL SIZE:	scaling indicator, ‘height'; height and width, 0.01 of longest side of default VDC EXTENT.
SYMBOL ORIENTATION:	as default CHARACTER ORIENTATION
INHERITANCE FILTER:	segment
CLIP INHERITANCE:	state list

SEGMENT TRANSFORMATION:	1.0,0.0, 0.0,1.0, 0.0,0.0
SEGMENT HIGHLIGHTING:	normal
SEGMENT DISPLAY PRIORITY:	0
SEGMENT PICK PRIORITY:	0

9 Profiles and conformance

9.1 Introduction

9.1.1 Objectives

This clause provides rules for defining valid profiles of ISO/IEC 8632. Profiles are used as a method for defining subsets of ISO/IEC 8632 by identifying the CGM elements, parameters, options, and implementation requirements necessary for meeting a particular set of requirements.

The primary objectives of the profile rules are:

a) to promote interoperability by minimizing arbitrary subsets of ISO/IEC 8632;

b) to provide the framework for developing profiles;

c) to promote uniformity in the development of conformance tests;

d) to supplement ISO/IEC TR10000 for International Standardized Profiles (ISPs) for the CGM standard;

e) to provide a basis for evaluating profiles as potential ISPs.

9.1.2 Scope

This clause:

a) defines the concept of profiles of ISO/IEC 8632;

b) provides rules for defining profiles of ISO/IEC 8632;

c) provides conformance criteria for profiles of ISO/IEC 8632;

d) provides conformance criteria for metafiles, metafile generators, and metafile interpreters;

e) defines criteria on which to evaluate profiles of ISO/IEC 8632;

f) provides a Profile Proforma (PPF) and Model Profile (see Annex I).

This clause addresses the CGM data stream and implementation requirements. Implementation requirements address the latitude allowed by CGM generators and interpreters. This clause does not directly address the environmental, performance, or resource requirements of the generator or interpreter.

This clause does not define the application requirements or dictate application functional content of a profile — the latter is the purview of application constituencies.

The scope of this clause is limited to rules for valid profiles for open interchange of graphical picture metafiles.

9.1.3 Concept and purpose of profiles for CGM

A major goal of ISO/IEC 8632 is to facilitate the transfer of picture information between computers, sites, and applications.

Profiles provide a means to:

a) improve interoperability between implementations by inhibiting the proliferation of private subsets of ISO/IEC 8632;

b) provide a foundation for testing and promote uniformity of conformance tests;

c) enhance the availability of consistent implementations of profiles.

A profile of ISO/IEC 8632 defines the options, elements, and parameters of ISO/IEC 8632 necessary to accomplish a particular function and maximize the probability of interchange between systems implementing the profile. Profiles are defined by application constituencies who agree to adhere to the same subset of CGM for the purpose of graphical data interchange using ISO/IEC 8632. Alternatively, profiles of ISO/IEC 8632 may be part of a set of interrelated standards and profiles assembled for the purpose of accomplishing a larger functional purpose.

A profile may:

d) give the meaning of implementation dependent semantics of some elements;

e) enforce common resolution of ambiguous semantics of ISO/IEC 8632;

f) ensure that identical use of identical elements and parameter values has the same meaning;

g) specify subsets or groupings of registered items from the appropriate ISO/IEC registers;

h) prohibit undefined or ill-defined elements or parameter values.

A profile of ISO/IEC 8632, according to the taxonomy of ISO/IEC TR10000-2, is an FCG Profile, that is, an interchange format and representation profile of CGM.

A profile of ISO/IEC 8632 shall not specify any requirement that would contradict or cause non-conformance to ISO/IEC 8632. Any metafile conforming to a profile of ISO/IEC 8632 conforms to ISO/IEC 8632.

Profiles address metafile requirements, as well as implementation requirements for metafile generators and metafile interpreters. Profiles define maximum requirements for generators and minimum requirements for interpreters.

9.1.4 Purpose of the Model Profile

The Model Profile serves two purposes:

1) It is a usable, implementable instance of a profile of ISO/IEC 8632. It is the only instance contained in ISO/IEC 8632. While it is designed to be implementable on a range of systems, it is also designed with modest limits that will not preclude its implementation in limited environments (low to mid-range computing systems). The Model Profile may not be suitable for application communities with more advanced and demanding requirements.

2) It is a guide to writing profiles. As an instance of a profile, the Model Profile is a starting point from which an application-specific profile should be defined, for those application communities for which the Model Profile itself does not suffice. Writers of profiles should consider each of the specifications of the Model Profile and either accept the specifications where they are adequate, or modify them when not.

9.2 Conformance

9.2.1 Conformance of profiles

A profile of ISO/IEC 8632

a) shall meet all requirements specified in ISO/IEC 8632;

b) shall be structured in accordance with the structural components and presentation rules defined in 9.4;

c) shall not specify any requirements that would contradict or cause non-conformance to ISO/IEC 8632;

d) may contain a conformance clause that adds requirements that are more specific and limited in scope than ISO/IEC 8632;

e) shall meet the conformance requirements for a FCG Profile as defined in ISO/IEC TR10000-1;

f) shall meet all the specific rules in this clause.

9.2.2 Conformance of metafiles

Conformance of metafiles to ISO/IEC 8632 is defined in terms of conformance to profiles. A metafile conforms to ISO/IEC 8632 if it conforms to a profile.

In order to conform to a profile of ISO/IEC 8632, a metafile

a) shall be a syntactically correct metafile for a specific version;

b) shall conform to all profile requirements defined for that version.

A metafile is a syntactically correct metafile for a specific version of ISO/IEC 8632 if the following conditions are met.

c) The metafile contains exactly one correct METAFILE VERSION element.

d) All graphical elements contained therein match the functional specification of the corresponding elements of ISO/IEC 8632-1 for that version. The metafile shall obey the relationships defined in the formal grammar for that version, the state tables, and all other syntactic requirements for that version.

e) The sequence of elements in the metafile obeys the relationships specified in ISO/IEC 8632-1 for that version, producing the structure specified in ISO/IEC 8632-1. For example, the metafile must begin with BEGIN METAFILE and end with END METAFILE, and include exactly one metafile descriptor at the beginning which contains at least all the required elements.

f) No elements appear in the metafile other than those specified in ISO/IEC 8632-1 for that version, unless required for the encoding technique. All non-standardized elements are encoded using the ESCAPE or GDP elements or the external elements APPLICATION DATA and MESSAGE.

g) The metafile is encoded according to the rules in one of the standardized encodings specified in ISO/IEC 8632-3 or ISO/IEC 8632-4.

9.2.3 Conformance of metafile generators

Conformance of metafile generators is defined in terms of conformance to a particular profile of CGM.

If ***P*** is a profile of CGM which conforms to the rules of this clause, then a metafile generator is a conforming ***P*** generator if it:

a) generates only metafiles which conform to the requirements of profile ***P*** or is directed to operate in a mode where only such metafiles can be generated;

b) maps the graphical characteristics of the pictures onto a set of CGM elements which define those pictures within the accuracy and latitude defined by the Generator Implementation Requirements in the profile ***P***.

A metafile generator which conforms to the Model Profile for a specific version, shall:

c) generate no syntax in violation of that version of ISO/IEC 8632;

d) generate metafiles which conform to that version of the Model Profile;

e) map the graphical characteristics of application pictures onto a set of CGM elements which define those pictures within the latitude allowed by the Generator Implementation Requirements of the Model Profile.

9.2.4 Conformance of metafile interpreters

Conformance of metafile interpreters is defined in terms of conformance to a particular profile of CGM.

If ***P*** is a profile of CGM which conforms to the rules of this clause, then a metafile interpreter is a conforming ***P*** interpreter if it:

a) is able to read any metafile which conforms to the requirements of profile ***P***;

b) renders the graphical characteristics of the CGM elements in any such metafile into a graphical image or picture within the accuracy and latitude defined by the Interpreter Implementation Requirements in the profile ***P***.

A metafile interpreter which conforms to the Model Profile for a specific version, shall:

c) be able to read any metafile which conforms to that version of the Model Profile of ISO/IEC 8632;

d) render the graphical characteristics of the CGM elements in any such metafile into a graphical image or picture within the latitude defined by the Interpreter Implementation Requirements of the Model Profile.

9.3 Criteria for designing profiles

The following criteria provide the means for determining the appropriateness and correctness of proposed profiles. The objective is to limit the proliferation of profiles and ensure the quality of those profiles.

9.3.1 Criteria on the profile in its entirety

The following criteria shall be applied to a proposed profile.

a) The application constituency and functional purpose of a proposed profile shall be well defined.

b) The functional purpose of a proposed profile shall not be satisfied by an existing profile. If the functional purpose of a proposed profile can be satisfied by a derivative of an existing profile, it shall be so defined — significant subsets shall not be replicated.

c) The proposed profile shall meet the identified functional requirements.

9.3.2 Criteria for the technical content of the profile

The following criteria shall be applied to the technical content of a proposed profile.

a) A proposed profile shall not specify requirements that violate ISO/IEC 8632.

b) A proposed profile shall place requirements on the CGM and not on the internal behaviour, structure, or performance of implementations (e.g., generators and interpreters).

c) A proposed profile may contain requirements on the functional and graphical characteristics of implementations claiming conformance to the profile.

d) A proposed profile shall be consistent in its requirements regarding CGM elements and parameters. For example, if a profile places no restrictions on the number of indexes defined by the CHARACTER SET LIST element, then it is inconsistent to place a restriction on CHARACTER SET INDEX.

e) A proposed profile shall not specify requirements which are conflicting, unnecessary, or redundant.

9.4 Form and format of a profile

A profile of ISO/IEC 8632 shall contain the following components:

a) a concise definition of the scope and purpose of the profile;

b) a scenario illustrating the profile's use and applicability;

c) all references to ISO/IEC 8632, i.e., approved amendments, errata, and registers.

d) references to any other relevant source documents;

e) specification of the set of elements, parameters, implementation requirements, and features of ISO/IEC 8632, presented in the format and according to the rules of 9.5.

f) a definintion of conformance of metafiles and implementations to the profile.

The content and layout of the profile shall conform to the *Rules for Drafting and Presentation of International Standardized Profiles* in annex A of ISO/IEC TR10000-1.

9.5 Profile rules, proforma, and model profile

9.5.1 Overview

This clause presents:

a) rules for defining CGM profiles;

b) a Profile Proforma (PPF);

c) a definition of the Model Profile.

The PPF is a set of tables which are a template for writing profiles. Most of the profile rules are inherent in the structure of the PPF. For example, the PPF requires certain information to be completed — each such case is a statement, equivalent to the rule, "Profiles shall specify ...".

All CGM profiles shall include a completed PPF.

Annex I contains the completed PPF of the Model Profile, together with the empty PPF template.

The following subcluases address

General Principles which apply to all profiles.

Metafile Rules, which apply to the general characteristics of a conforming metafile.

Multi-element Rules, which apply to several elements.

Individual Element Rules, which apply to the elements one by one. A rule is described for each element defined in clause 7. Each rule in this subclause specifies whether a profile must address the rule, whether a profile may optionally address the rule, or whether the profile shall not restrict the use of the element in any manner.

Generator Implementation Requirements, which apply to the behaviour of CGM generator implementations.

Interpreter Implementation Requirements, which apply to the behaviour of CGM interpreter implementations.

Rules which address encoding issues are described in parts 3 and 4 of ISO/IEC 8632.

The PPF is presented in tabular form supplemented by descriptive material. The PPF template and the Model Profile are presented together in Annex I.

9.5.2 General principles

9.5.2.1 Self-identification of profiles

It is required that the METAFILE DESCRIPTION element identify the profile, and its edition, to which the metafiles conform. The edition indicates the version or release date of the profile.

9.5.2.2 Source identification of profiles

The optional information of the PPF, for the METAFILE DESCRIPTION element, allows profiles to require that metafiles identify their source (e.g., vendor, product, product version).

9.5.2.3 Private encodings

The "encodings" item of the PPF effectively prohibits profiles from specifying private encodings.

9.5.2.4 Restrictions on grammar and state tables

In completing the PPF, profile writers may restrict the use of some elements by restricting the formal grammar or the state tables.

EXAMPLE — A profile may specify that segments are not allowed in Picture Body by either:

1) restricting the formal grammar:
   Replace
   <picture content> ::= <picture element> **|** <segment>
   with
   <picture content> ::= <picture element>.

2) modifying the state table:
   Remove the "X" in the POS column of table 8a in "BEGIN SEGMENT".

3) making the statement:
   Segments are not permitted to appear in the Picture Body.

9.5.2.5 Defining subsets

It is a principal role of profiles to define subsets of the options of ISO/IEC 8632. However, defining subsets of ISO/IEC 8632 shall not be arbitrary and shall have a clear connection to the achievement of one or more of the defined goals of the profile.

9.5.2.6 Metafile defaults

Clause 8 addresses all elements which have default values. While no profile can change these values, an equivalent effect may be achieved by use of the METAFILE DEFAULTS REPLACEMENT element. Profiles may require that a metafile contain a METAFILE DEFAULTS REPLACEMENT element with well-defined content.

For default values in clause 8 which are listed as "device dependent" or "interpreter dependent", if there is a element for setting the value of such an element, then profiles shall require the use of such elements. The appropriate element shall be included in the METAFILE DEFAULTS REPLACEMENT element or in the metafile body. For such elements, profiles shall not assign implicit defaults. These elements include:

<u>For profiles of Version 2 (and above)</u>

LINE REPRESENTATION

MARKER REPRESENTATION

TEXT REPRESENTATION

FILL REPRESENTATION

EDGE REPRESENTATION

Specifying the default value for elements which specify colour values shall be consistent with the rules for colour (see 9.5.4.1). Specifically, if all colours used within the metafile shall be defined, then the element for setting the colour value shall be used; otherwise, the element shall not be used. The elements affected by this rule are:

BACKGROUND COLOUR

COLOUR TABLE

When specified as a direct colour:

LINE COLOUR

MARKER COLOUR

TEXT COLOUR

FILL COLOUR

EDGE COLOUR

If no element exists to set the value, then the profile shall define the default values to be used. The elements affected by this rule, because they do not exist in Version 1 include:

For profiles of Version 1 (only)

LINE REPRESENTATION

MARKER REPRESENTATION

TEXT REPRESENTATION

FILL REPRESENTATION

EDGE REPRESENTATION

9.5.2.7 Restricting element values

In those cases where it is necessary to restrict an element to its default value in order to meet the goals of a profile, the restriction shall be achieved by allowing the element to appear in conforming files and restricting its value to the default rather than prohibiting the element.

NOTE The implementation burden necessary to implement this guideline is small compared to the interoperability gain. Many implementations, even if only interested in the default value, consider such a "defensive" strategy to be good insurance against mistakes of other implementations in realizing the defaults of ISO/IEC 8632.

9.5.2.8 Classification of elements and parameter values

Elements and parameter values can be classified as either standard, registered, profile-defined, or private.

— Standard refers to elements and parameter values which have been defined in ISO/IEC 8632.

— Registered refers to elements and parameter values which have been entered into a registry and thus, have an internationally recognized definition and have undergone a standardization process.

— Profile-defined refers to elements such as ESCAPEs and GDPs, whose syntax, semantics, and identifier are defined within the profile.

— Private refers to truly private elements and parameter values, that is, known only by prior agreement between generators and interpreters.

Profiles shall limit metafiles to standard, registered, and profile-defined metafile elements and/or parameter values.

Profiles shall limit the use of ESCAPE and GDP elements to those which are registered or profile-defined.

Profiles shall prohibit private elements and parameter values which produce graphical effect. Profiles may allow private elements and parameter values which produce no graphical effect, (e.g., APPLICATION DATA).

9.5.2.9 Registered elements and values

Profiles shall specify the set of registered elements and parameter values which are allowed. Registered elements and parameter values are those which have been entered into any registry which is referenced in a normative manner by ISO/IEC 8632 (see 6.12). Profiles shall refer to these registered items using their registered identifier and definition.

If a CGM profile exists which contains profile-defined, non-registered elements with values which are affected by the ISO International Register of Graphical Items, then the elements shall be registered first before such a profile is approved as an ISP.

9.5.2.10 Generator and interpreter behaviour

Profiles of CGM shall address implementation conformance requirements. Profiles shall not address the internal structure, performance or other internal behavioural characteristics of implementations of generators or interpreters. These issues may be addressed by the application community in supplemental documentation.

9.5.2.11 Physical media

Physical file format and other issues of media, delivery, or networking are beyond the scope of ISO/IEC 8632 and shall not be specified by profiles of ISO/IEC 8632. These issues may however be important for successful interoperability and if addressed by the application commmunity, shall be addressed in specifications other than the profiles.

9.5.3 Metafile rules

The metafile rules are completely contained in the PPF table 13. These rules apply to the entire metafile.

9.5.4 Multi-element rules

The information in this subclause explains and supplements the rules in the PPF tables 14 through 24.

9.5.4.1 Colour

The PPF requires that profile writers choose one of the following rules.

a) For each metafile, either all colours used within the metafile, including background and foreground colours, shall be defined or none shall be defined.

b) For all metafiles, all colours used within the metafile, including background and foreground colours, shall be defined.

c) For all metafiles, no colours shall be defined.

NOTE This rule requires that, if any direct colours are to be explicitly specified in the metafile, or if any indexed colours are to have their representations defined, then these should be done for the colours associated with each primitive type (e.g., line, marker, text, fill area) before the first occurrence of a primitive of that type.

Colour is "used" at the point that the associated graphical primitive is encountered in the metafile, except for primitives in Picture Descriptor or Metafile Descriptor segments. In the latter cases, colour is used when the associated primitive is encountered during execution of a COPY SEGMENT. The background colour is used at BEGIN PICTURE BODY.

Profiles shall not define implicit defaults for colour attributes or for colour representations.

Profiles may define conformance categories for both metafiles and implementations based on colour content and capability: colour, greyscale, and monochrome.

The elements affected by these colour rules are:

BACKGROUND COLOUR	LINE COLOUR
LINE REPRESENTATION	MARKER COLOUR
MARKER REPRESENTATION	TEXT COLOUR
TEXT REPRESENTATION	FILL COLOUR
FILL REPRESENTATION	EDGE COLOUR
EDGE REPRESENTATION	CELL ARRAY
AUXILIARY COLOUR	PATTERN TABLE
BITONAL TILE	COLOUR TABLE
TILE	SYMBOL COLOUR

9.5.4.2 Indexes (except colour)

The purpose of this rule is that every referenced index value correspond to a well-defined representation of that value. Specific rules are given in the PPF tables for the individual elements listed below.

An index value is "referenced" if it appears explicitly in a referencing element. An index value is also referenced if it is the default value of that referencing element, and is used in the display of a primitive.

The elements affected by this rule are:

Referencing element	Corresponding representation element
LINE BUNDLE INDEX	LINE REPRESENTATION
LINE TYPE	LINE AND EDGE TYPE DEFINITION
MARKER BUNDLE INDEX	MARKER REPRESENTATION
TEXT BUNDLE INDEX	TEXT REPRESENTATION
TEXT FONT INDEX	FONT LIST
FONT PROPERTIES	FONT LIST

CHARACTER SET INDEX	CHARACTER SET LIST, GLYPH MAPPING
ALTERNATE CHARACTER SET INDEX	CHARACTER SET LIST, GLYPH MAPPING
FILL BUNDLE INDEX	FILL REPRESENTATION
HATCH INDEX	HATCH STYLE DEFINITION
PATTERN INDEX	PATTERN TABLE and GEOMETRIC PATTERN DEFINITION
EDGE BUNDLE INDEX	EDGE REPRESENTATION
EDGE TYPE	LINE AND EDGE TYPE DEFINITION
SYMBOL LIBRARY INDEX	SYMBOL LIBRARY

NOTE LINE TYPE, HATCH INDEX, and EDGE TYPE values which are either registered or clause 7 values are already well-defined.

9.5.4.3 Line primitives - geometric degeneracies

The PPF requires that profiles specify whether geometric degeneracies are permitted or prohibited.

If geometric degeneracies are permitted, the PPF requires that profiles specify the graphical meaning of the degeneracy.

The elements affected by this rule are:

POLYLINE	HYPERBOLIC ARC
DISJOINT POLYLINE	PARABOLIC ARC
GENERALIZED DRAWING PRIMITIVE	NON-UNIFORM B-SPLINE
CIRCULAR ARC 3 POINT	NON-UNIFORM RATIONAL B-SPLINE
CIRCULAR ARC CENTRE	POLYBEZIER
ELLIPTICAL ARC	Compound Line
CIRCULAR ARC CENTRE REVERSED	

9.5.4.4 Area primitives - geometric degeneracies

The PPF requires that profiles specify whether geometric degeneracies are permitted or prohibited.

If geometric degneracies are permitted, the PPF requires that profiles specify the graphical meaning of the degeneracy.

The elements affected by this rule are:

POLYGON	CIRCULAR ARC CENTRE CLOSE

POLYGON SET
RECTANGLE
CIRCLE
CIRCULAR ARC 3 POINT CLOSE
GENERALIZED DRAWING PRIMITIVE
ELLIPSE
ELLIPTICAL ARC CLOSE
CONNECTING EDGE
Closed Figure

9.5.4.5 Graphical text strings

Length

The PPF requires that profiles specify a finite limit for the maximum string length (bytes) in graphical text strings. This limit applies to the total length of the text string including all appended text elements.

NOTE The string length in bytes equals the number of character codes for single-byte character sets, but not for multi-byte character sets.

Content

All profiles prohibit the non-printing codes, except for NUL (code value 0) and the codes required to effect ISO/IEC 2022 character set switching consistent with the value of the CHARACTER CODING ANNOUNCER. Profiles may further restrict the use of ISO/IEC 2022 switching controls.

The elements affected by this rule are:

TEXT

RESTRICTED TEXT

APPEND TEXT

GENERALIZED DRAWING PRIMITIVE

9.5.4.6 Non-graphical text strings

Length

The PPF requires that profiles specify a finite limit for the maximum string length (bytes) in non-graphical text strings (e.g., elements with data type SF).

Profiles may specify two length numbers:

one for the group of elements with data type SF parameters;

one for the group of elements with data type D parameters which have type SF data within the D parameters, in the case that the D parameter uses SDR formatting and contain type SF data.

NOTE The string length in bytes equals the number of character codes for single-byte character sets, but not for multi-byte character sets.

Content

The PPF requires that all profiles:

prohibit all C0 control except for NUL (code value 0) which has no effect and ISO/IEC 2022 character set switching. Profiles may further restrict the use of ISO/IEC 2022 switching controls.

The elements affected by this rule are:

Type SF Parameters	SF data within D parameters
BEGIN METAFILE	GENERALIZED DRAWING PRIMITIVE
BEGIN PICTURE	ESCAPE
METAFILE DESCRIPTION	APPLICATION DATA
FONT LIST	
CHARACTER SET LIST	SF data within SDR parameters
MESSAGE	BITONAL TILE
FONT PROPERTIES	TILE
GLYPH MAPPING	
SYMBOL LIBRARY LIST	
BEGIN APPLICATION STRUCTURE	
APPLICATION STRUCTURE ATTRIBUTE	
PICTURE DIRECTORY	
APPLICATION STRUCTURE DIRECTORY	

9.5.4.7 Data record strings

Length

The PPF requires that profiles specify a finite limit for the maximum string length (bytes) in the data record or specify that there is no limit.

Content

The PPF requires that profiles require the SDR-coding techniques (see annex C.2.2) for the data records.

The elements affected by this rule are:

GENERALIZED DRAWING PRIMITIVE

ESCAPE

APPLICATION DATA

APPLICATION STRUCTURE ATTRIBUTE

NOTE 1 The string length in bytes equals the number of character codes for single-byte character sets, but not for multi-byte character sets.

NOTE 2 Data Record (D) is a data type which is coded in parts 3 and 4 of ISO/IEC 8632 with delimiting syntax equivalent to the coding syntax for S and SF data types.

NOTE 3 The binary encoding of the enumerated (E) data type in SDR uses integers, (e.g., 0, 1, 2); whereas the clear text encoding of the enumerated data type traditionally uses words (e.g., ‘on’, ‘off’). This difference may prevent blind intertranslation between encodings. It is recommended that the use of enumerated (E) data type in SDR coding for ESCAPE and GDP elements be avoided. Type index (I) should be used instead. Although this is now a recommended practice, there are some existing registered items which use the word-encoded method for E within SDR.

NOTE 4 Graphical Registration requires the SDR formatting of D parameters. Registration of these elements is required in order for a profile to be considered as an ISP.

9.5.5 Individual element rules

All rules are contained in the PPF tables 15 through 24.

9.5.6 Generator implementation requirements

The information in this subclause explains and supplements the rules in the PPF table 25.

The PPF requires that profile writers address Generator Implementation Requirements (GIR). Conforming generators shall be consistent with a specific profile or the Model Profile specified herein.

The implementor's documentation shall specify the profile to which the generator conforms.

Generators shall produce conforming metafiles whose contents accurately represent the source picture according to the semantic rules of ISO/IEC 8632 and the profile as identified in the Metafile Descriptor and the generator product documentation.

9.5.6.1 Relationship to other profiles

Profiles shall not prohibit a generator implementation from implementing more than one particular profile provided that the generator can be commanded to operate in a mode that produces metafiles that conform to the profile as identified in the Metafile Descriptor.

9.5.6.2 Generator fidelity

The PPF requires that profiles address the fidelity requirements for the mapping of application graphical objects to metafile elements.

Categories that may be addressed include, without limitation, the geometric accuracy of the mapping of graphical primitives and the realization of primitive attributes.

9.5.6.2.1 Generator colour requirements

Cases may arise in which the colours of application pictures cannot be encoded in an obvious and straightforward manner into the colour facilities permitted in the metafile by a profile.

EXAMPLES

1 — The application has many more colours than the metafile.

2 — The meaning of the Version 1 metafile RGB specification (1,0,0), "full red" is ambiguous (i.e., the colour is uncalibrated).

The PPF requires that profiles address, without limitation, the following aspects of mapping application colour into metafile colour specifications:

a) reduction of the number of colours used by the application to match metafile limits imposed by the profile;

b) definition of mapping algorithms, metrics to be used, and colour space in which the mapping algorithms and metrics are to be applied (e.g., CIEXYZ, regardless of the metafile colour space);

c) implicit colour calibration specifications to be used in mapping application colours to metafile colour specifications, for Version 1 and Version 2 profiles; Version 3 profiles shall use or require the explicit Version 3 elements.

9.5.6.2.2 Generator geometric accuracy and latitude

The PPF requires that profiles address the geometric accuracy required of conforming generators and the latitude permitted when application graphical primitives are mapped to CGM graphical primitive elements.

Aspects of the geometric accuracy which may be addressed include placement of the primitive and the size of the primitive. Geometric accuracy may be measured by reference to the VDC extent or by reference to the size or extent of the primitive itself.

9.5.6.2.3 Generator text accuracy and latitude

The PPF requires that profiles address, independently of overall "Generator geometric primitives accuracy and latitude", the accuracy and latitude of mapping the application text strings into metafile specifications.

9.5.6.2.4 Generator font substitution

The PPF requires that profiles address the substitution of fonts permitted in conforming metafiles for fonts in the application picture.

Specific substitution methods may be addressed. If addressed, substitution methods and latitudes may include, without limitation, discussion of similarity of visual characteristics of fonts, as well as font metrics and individual glyph metrics. Visual characteristics, if addressed, shall be addressed at least according to the set of font properties defined in the FONT PROPERTIES element of clause 7.

9.5.6.3 Preservation of primitives

The PPF requires that profiles address the preservation of the identities of graphical primitive elements. The "preservation of grahical primitive elements" refers to the CGM primitives which are used to represent the graphical characteristics of the application picture. For example, using CGM polygons to represent application ellipses. Profiles may explicitly state which primitive substitutions are allowed and which are not.

9.5.6.4 Semantic latitude

The PPF requires that profiles address those elements where semantic latitude is permitted in ISO/IEC 8632.

Profiles shall address at least the following:

a) drawing priority and mode: whether the order in which the graphical characteristics of the application picture are created is reflected in the metafile by the ordering of the primitives;

b) clipping: interaction between VDC extent and clip rectangle for all metafiles;

c) precise meaning of the predefined line types, edge types, and hatch styles.

9.5.6.5 Generator error processing

The PPF requires that profiles address the action taken if an error occurs while generating the CGM. Profiles may address, without limitation:

— classification of error severity;

— requirements for error recovery;

— requirements for error reporting.

9.5.6.6 Reporting

The PPF requires that profiles address whether reporting shall be required.

Profiles may address the method and format of the reporting (e.g., continuous log file, on-screen operator message, dump files).

Profiles may require the reporting of any substitution, error, fallback behaviour, mappings, or other behaviour.

9.5.6.7 Degeneracies

The PPF requires that profiles address the permissibility of generation of metafile graphical primitive elements which are degenerate in the senses discussed in Annex D subclauses D.2.2, D.2.3, D.4.5.4 through D.4.5.8, D.4.5.11, and D.4.5.12.

Profiles may address individually the two possible sources of such degeneracies:

— intrinsic: the degeneracy already exists in the specification of the application graphical object;

— computational: the object is not intrinsically degenerate, but it degenerates during the computation and data preparation which generates the metafile element.

9.5.7 Interpreter implementation requirements

The information in this subclause explains and supplements the rules in the PPF table 26.

The PPF requires that profile writers address Interpreter Implementation Requirements (IIR). Conforming interpreters shall be consistent with a specific profile or the Model Profile specified herein.

The implementor's documentation shall specify the profile to which the interpreter conforms.

Interpreters shall produce a target picture whose appearance agrees with the semantic rules of the metafile elements as defined in ISO/IEC 8632 and the profile.

9.5.7.1 Relationship to other profiles

Profiles shall not prohibit an interpreter implementation from implementing more than one particular profile provided that the implementation can be commanded to operate in a mode that interprets metafiles that conform to the profile as identified in the Metafile Descriptor.

9.5.7.2 Number of pictures

If the profile permits zero pictures in a metafile, then the profile shall specify the behaviour of the interpreter when zero pictures are encountered.

9.5.7.3 Empty pictures

If the profile permits empty pictures, then the profile shall specify the behaviour of the interpreter when empty pictures are encountered.

9.5.7.4 Interpreter fidelity

The PPF requires that profiles address the fidelity requirements for interpretation of all elements containing graphical information.

NOTE It is recommended, if a profile intends to allow approximations for a particular element, that the specification of clause D.4 be endorsed (if one exists for the particular element), in the absence of specific profile requirements to the contrary.

Categories which may be addressed include, without limitation, the geometric characteristics of graphical primitives and the realization of primitive attributes.

9.5.7.4.1 Interpreter colour requirements

For the purposes of these profile rules, the presentation of metafile colour information by interpreters is modelled as a two step process.

In the first step, the interpreter maps the metafile colour information, whether indexed or direct, to an intermediate abstract discrete colour space, designated as "the Required Interpreter Support Set". This is called the *colour mapping step*. For example, if the profile permits an unlimited number (within the constraints imposed by the numerical precisions) of direct colours in the metafile, the profile might require all conforming interpreters to support, with a unique and distinguishable representation, the 125 colours which are represented by the 5x5x5 uniform grid applied to the RGB unit cube.

In the second step of the colour presentation model, the interpreter choses a representation for each colour in the Required Interpreter Support Set and renders the colour to the device. This is called the *colour rendering step*. For example, a CMYK printer uses colour dither algorithm to approximate the 4096 colours in a 16x16x16 gridding of the RGB cube; whereas a black-and-white raster laser printer interpreter maps all foreground colour information to black.

The PPF requires that profiles address at least two aspects of the interpretation of colour information in the metafile.

Specific points that may be addressed include, without limitation:

a) permissibility of mapping metafile colour to one or more of the sets:
   — a colour set with fewer colours than those in the metafile;
   — a greyscale set;
   — a monochrome (two-colour) set.

b) implicit colour calibration specifications to be used in the colour mapping step for Version 1 and Version 2 profiles; Version 3 profiles shall use or require the explicit Version 3 elements.

c) if the colour mapping step is permitted to be other than the identity mapping, the definition of the Required Interpreter Support Set (or possibly multiple sets, if such are allowed).

d) the definition of mapping algorithms, metrics to be used, and colour space in which the mapping algorithms and metrics are to be applied (e.g., CIEXYZ, regardless of the metafile colour space).

9.5.7.4.2 Interpreter geometric accuracy and latitude

The PPF requires that profiles address the geometric accuracy required of conforming interpreters and the latitude permitted in placement of primitives and realization of geometric aspects when geometric primitive elements are rendered.

NOTE A profile may specify several sets of requirements according to the application needs with respect to the interpretation. For example, accuracy may be different for previewing and for final drawing.

9.5.7.4.3 Interpreter text rendering

The PPF requires that profiles address, independently of overall "Interpreter geometric primitives accuracy and latitude", the accuracy and latitude of mapping the metafile text specifications into application text strings.

Profiles may address the precision of text rendering. Profiles may require all text to be rendered according to a specified precision regardless of the value of the metafile TEXT PRECISION element.

9.5.7.4.4 Font substitution

The PPF requires that profiles address whether or not interpreters are permitted to substitute other fonts for fonts specified in the FONT LIST element of the metafile.

If the profile prohibits substitution, then the font name in the FONT LIST shall refer to a specific and unambiguously defined font resource, and interpreters shall use that font resource.

If the profile permits substitution, then the profile may address specific methods of substitution. If addressed, substitution methods and latitudes may include, without limitation, discussion of similarity of visual characteristics of fonts, as well as accuracy of matching the font metrics and individual glyph metrics (see also 9.5.7.4.3). Visual characteristics, if addressed, shall be addressed according to the set of font properties defined in the FONT PROPERTIES element of clause 7, as a minimum.

If font substitution is permitted, then the profile shall contain a reference set of font and glyph metrics which correspond to the canonical instances of the substitutable fonts.

9.5.7.5 Semantic latitude

The PPF requires that profiles address those elements where semantic latitude is permitted in ISO/IEC 8632.

Profiles shall address at least the following:

a) drawing priority and mode: whether primitives occurring later in the metafile are displayed "on top of" primitives occurring earlier, and whether they replace earlier primitive parts or combine with them in some way;

b) view surface clearing at picture start;

c) clipping: clip modes for Version 1 metafiles; interaction between VDC extent and clip rectangle for all metafiles;

d) precise meaning of the predefined line types, edge types, and hatch styles;

e) the value ‘unspecified' in LINE JOIN, LINE CAP, EDGE JOIN, EDGE CAP for Version 1 and Version 2 metafiles;

f) algorithms for RESTRICTED TEXT TYPE in Version 1 and Version 2 metafiles.

9.5.7.6 Interpreter error processing

The PPF requires that profiles address the action taken if an error occurs while interpreting the CGM. Profiles may address, without limitation:

— classification of error severity;

— requirements for error recovery;

— requirements for error reporting.

9.5.7.7 Reporting

The PPF requires that profiles address whether reporting shall be required.

Profiles may address the method and format of the reporting (e.g., continuous log file, on-screen operator message, dump files).

Profiles may require the reporting of any substitution, error, fallback behaviour, mappings, or other behaviour.

9.5.7.8 Degeneracies

The PPF requires that profiles address the interpretation and rendering of metafile graphical primitive elements which are degenerate in the senses discussed in annex D subclauses: D.2.2, D.2.3, D.4.5.4 through D.4.5.8, D.4.5.11, and D.4.5.12.

Profiles may address individually the two possible sources of such degeneracies:

intrinsic: the degeneracy already exists as the element is specified in the metafile;

computational: the element is not intrinsically degenerate, but it degenerates during the computation of the interpreting and rendering process.

9.5.7.9 Transparency

The PPF requires that profiles address the interpretation of transparency. The profile shall specify whether it is as described in the 2nd and 3rd paragraphs of the description in 7.5.4 or, if not, how transparency is implemented.

9.5.7.10 Interpretation of application structures and directories

The PPF requires that profiles address the interpretation of Application Structure elements and Directory elements (PICTURE DIRECTORY and APPLICATION STRUCTURE DIRECTORY).

9.5.8 PPF Tables

The Profile Proforma is contained in tables 13 through 26 of Annex I. These tables when completed by the author of the profile, contain the normative specifications of the profile.

The PPF tables have 3 columns:

— The first identifies the element or functionality to be addressed.

— The second is the template for the profile writer to complete.

— The third is the completed specification for the Model Profile.

Each item to be addressed — element or functionality — comprises a "row" of the table.

The first column of each row contains:

1) A unique identifier for that row, T.n.m, indicating that this is row "m" of table "n". For example, row T.16.2 is the second row of table 16 (the METAFILE DESCRIPTION element).

2) The name of the item for that row, for example, METAFILE DESCRIPTION.

3) If appropriate for an item, its metafile version (v1, v2, or v3). This is the lowest metafile version for which the item is defined. If the state rules of an element are different depending upon the metafile version, then multiple version numbers are given.

4) References to other sections of this clause, clause 7, and the annexes for additional normative or informative material.

A second column is the PPF template for profile authors. For each row (element or functionality) it contains:

5) A check box indicating that all specifications for this row for this profile are exactly the same as those for this row in the Model Profile.

6) Check boxes to indicate whether the item is required, permitted, or prohibited in metafiles conforming to the profile. If the check box choice is limited, then only the allowable check boxes are given. For example, if an element shall not be prohibited, then the "prohibited" check box is omitted.

7) One or more rules which are to be addressed by all profile authors. If the state rules or parameter values of an element are different depending upon the metafile version, then the PPF rule is qualified by the version number.

8) A general category, "Other:", in which profile authors may add any additional specifications which are consistent with the rules for profiles in this clause.

A third column is the Model Profile specification. Each row contains:

9) A checked box indicating the item's status.

10) The specifications of the Model Profile.

If the "Same as Model Profile" box is checked for a row, then no further information need be supplied for the profile in that row — all specifications for that row match the Model Profile. Otherwise, the profile shall have complete information for all column 2 rules within a row. It is acceptable in most cases to simply refer to the Model Profile, with the words "as Model Profile".

Rules presented as statements and ending with a semicolin (":"), shall be completed with specific information. In most cases, these rules may be prefaced with "Profiles shall specify...". Rules presented as questions and ending with a question mark ("?"), are optional, and shall be completed with either specific information or the word "none". In most cases, these rules may be prefaced with "Profiles may specify any ...".

The category "Other:" shall be completed with either the word "none" or with specific information.

It is possible that specific information for some rules may be too much to fit into the table space provided. In this case, the table entry shall specify (assuming that this is row T.n.m), "see Attachment n.m", and the specification shall be put into an attachment labeled "Attachment n.m".

Profile authors shall complete all required information in the template, column 2 of the PPF tables. Profiles may contain any other specifications, parameter restrictions, etc., unless explicitly prohibited by the rules of this clause and the PPF tables.

Annex A
(normative)

Formal grammar of the functional specification of version 1 metafiles

A.1 Introduction

This grammar is a formal definition of a standard CGM syntax for Version 1 metafiles. The encoding-independent and the encoding-dependent productions are separated, and there are subsections showing the syntax of each of the standardized encoding schemes. Details on the encoding of terminal symbols can be found in the parts of this International Standard that deal with the particular encoding schemes.

A.2 Notation used

<symbol>	nonterminal
<SYMBOL>	terminal
<symbol>*	0 or more occurrences
<symbol>+	1 or more occurrences
<symbol>o	optional (0 or 1 occurrences)
<symbol>(n)	exactly n occurrences, n=2,3,...
<symbol-1> ::= <symbol-2>	symbol-1 has the syntax of symbol-2
<symbol-1> | <symbol-2>	symbol-1 or alternatively symbol-2
<symbol: meaning>	symbol with the stated meaning
{comment}	explanation of a symbol or a production

A.3 Detailed grammar

A.3.1 Metafile structure

```
metafile>   ::= <BEGIN METAFILE>
                <metafile identifier>
              <metafile descriptor>
              <metafile contents>*
              <END METAFILE>

<metafile identifier>   ::=   <string fixed>

<metafile contents>     ::=   <extra element>*
              <picture>
              <extra element>*

<extra element> ::=     <external element>
             | <escape element>

<picture>   ::= <BEGIN PICTURE>
                <picture identifier>
              <picture descriptor element>*
              <BEGIN PICTURE BODY>
              <picture element>*
              <END PICTURE>

<picture identifier>    ::=   <string fixed>
```

```
<picture element> ::=   <control element>
             | <graphical element>
             | <primitive attribute element>
             | <pattern table element>
             | <colour table element>
             | <extra element>
```

A.3.2 Metafile descriptor elements

```
<metafile descriptor>   ::=   <<optional descriptor element>*
              <version>
              <optional descriptor element>*
              <element list>
              <optional descriptor element>*>
             | <<optional descriptor element>*
              <element list>
              <optional descriptor element>*
              <version>
              <optional descriptor element>*>

<version>   ::= <METAFILE VERSION>
                <integer>

<element list>::= <METAFILE ELEMENT LIST>
                  <MEL item>*

<MEL item>   ::=   <element name>*
                |       <element name shorthand enumerated>*

<element name shorthand
        enumerated>     ::=   <DRAWING SET>
             | <DRAWING PLUS CONTROL SET>

<optional descriptor element> ::=   <description>
             | <VDC TYPE>
                <vdc type enumerated>
             | <MAXIMUM COLOUR INDEX>
                <colour index>
             | <COLOUR VALUE EXTENT>
                <red green blue>(2)
             | <METAFILE DEFAULTS REPLACEMENT>
                <element default>+
             | <FONT LIST>
                <font name>+
             | <CHARACTER SET LIST>
                <character set definition>+
             | <CHARACTER CODING ANNOUNCER>
                <coding technique enumerated>
             | <scalar precision>
             | <extra element>

<description> ::= <METAFILE DESCRIPTION>
                <string fixed>

<vdc type enumerated>   ::=   <INTEGER>
             | <REAL>

<element default> ::=   <control element>
             | <picture descriptor element>
             | <primitive attribute element>
             | <extra element>
```

```
<font name> ::= <string fixed>

<character set definition>    ::=   <character set enumerated>
              <designation sequence>

<index>     ::= <standard index value>
             | <private index value>

<standard index value>  ::=   <positive integer>
<non-negative integer>  ::=   <integer>     {greater than or equal to 0}
<positive integer>      ::=   <integer>     {greater than 0}
<private index value>   ::=   <negative integer>
<negative integer>      ::=   <integer>     {less than 0}
<positive index>  ::=   <positive integer>

<character set enumerated>    ::=   <94 CHAR>
             | <96 CHAR>
             | <MULTI-BYTE 94 CHAR>
             | <MULTI-BYTE 96 CHAR>
             | <COMPLETE CODE>

<coding technique enumerated> ::=   <BASIC 7-BIT>
             | <BASIC 8-BIT>
             | <EXTENDED 7-BIT>
             | <EXTENDED 8-BIT>

<designation sequence>  ::=   <string fixed>

<scalar precision>      ::=   <INTEGER PRECISION>
                <integer precision value>
             | <REAL PRECISION>
                <real precision value>
             | <INDEX PRECISION>
                <index precision value>
             | <COLOUR PRECISION>
                <colour precision value>
             | <COLOUR INDEX PRECISION>
                <colour index precision value>
              {these elements have encoding}
              {dependent parameters}
```

NOTE The following order is recommended for the Metafile Descriptor elements: METAFILE VERSION, METAFILE ELEMENT LIST, (possible multiple occurrences of) METAFILE DESCRIPTION.

A.3.3 Picture descriptor elements

```
<picture descriptor element>  ::=   <SCALING MODE>
                <scaling specification mode enumerated>
                <metric scale factor>
             | <VDC EXTENT>
                <point> (2)
             | <BACKGROUND COLOUR>
                <red green blue>
             | <specification element>
             | <extra element>

<specification element> ::=   <COLOUR SELECTION MODE>
                <colour selection mode enumerated>
             | <LINE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>
             | <MARKER SIZE SPECIFICATION MODE>
                <specification mode enumerated>
```

```
            | <EDGE WIDTH SPECIFICATION MODE>
               <specification mode enumerated>

<colour selection mode
   enumerated>::= <INDEXED>
            | <DIRECT>

<scaling specification mode
   enumerated>::= <ABSTRACT>
            | <METRIC>

<metric scale factor>   ::=   <real>

<specification mode enumerated>     ::=   <ABSOLUTE>
            | <SCALED>
```

A.3.4 Control elements

```
<control element> ::=   <vdc precision>
            | <AUXILIARY COLOUR>
               <colour>
            | <TRANSPARENCY>
               <off-on indicator enumerated>
            | <CLIP RECTANGLE>
               <point>(2)
            | <CLIP INDICATOR>
               <off-on indicator enumerated>

<off-on indicator enumerated> ::=   <OFF>
            | <ON>

colour>     ::= <colour index>
            | <red green blue>

<point>     ::= <vdc value>(2)

<vdc precision> ::=     <VDC INTEGER PRECISION>
               <vdc integer precision value>
            | <VDC REAL PRECISION>
               <vdc real precision value>
              {these elements have encoding}
              {dependent parameters}
```

A.3.5 Graphical elements

```
<graphical element>     ::=   <polypoint element>
            | <text element>
            | <cell element>
            | <gdp element>
            | <rectangle element>
            | <circular element>
            | <elliptical element>

<polypoint element>     ::=   <POLYLINE>
               <point pair>
               <point list>
            | <DISJOINT POLYLINE>
               <point pair>
               <point pair list>
            | <POLYMARKER>
               <point>
               <point list>
```

```
             | <POLYGON>
                <point>(3)
                <point list>
             | <POLYGON SET>
                <point edge pair>(3)
                <point edge pair list>

<point list> ::= <point>*

<point pair list> ::=   <point pair>*

<point pair> ::= <point>(2)

<point edge pair> ::=   <point><edge out flag>

<point edge pair list>  ::=   <point edge pair>*

<edge out flag> ::=     <INVISIBLE>
             | <VISIBLE>
             | <CLOSE INVISIBLE>
             | <CLOSE VISIBLE>

<text element>::= <TEXT>
                <point>
                <text tail>
             | <restricted text element>

<restricted text element>     ::=   <RESTRICTED TEXT>
                <extent>
                <point>
                <text tail>

<extent>    ::= <non-negative vdc value>(2)

<text tail> ::= <final character list>
             | <nonfinal character list>

<final character list>  ::=   <FINAL>
                <string>

<nonfinal character list>     ::=   <NOT FINAL>
                <string>
                <partial text attribute element>*
                <spanned text>

<spanned text>::= <APPEND TEXT>
                <text tail>

<cell element>::= <CELL ARRAY>
                <point>(3)
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)

              {this element has an encoding}
              {dependent parameter}

<local colour precision>      ::=   <colour precision value>
             | <colour index precision value>
             | <default colour precision indicator>
```

```
<gdp element> ::= <GDP>
                <gdp identifier>
                <point list>
                <data record>

<gdp identifier>  ::=   <integer>

<rectangle element>     ::=   <RECTANGLE>
                <point pair>

<circular element>      ::=   <CIRCLE>
                <point>
                <radius>
             | <CIRCULAR ARC 3 POINT>
                <point>(3)
             | <CIRCULAR ARC 3 POINT CLOSE>
                <point>(3)
                <close type>
             | <CIRCULAR ARC CENTRE>
                <point>
                <valid vdc vector>(2)
                <radius>
             | <CIRCULAR ARC CENTRE CLOSE>
                <point>
                <valid vdc vector>(2)
                <radius>
                <close type>

<radius>    ::= <non-negative vdc value>

<valid vdc vector>      ::=   <<non-zero vdc value>
                <vdc value>>
             | <<vdc value>
                <non-zero vdc value>>

<non-zero vdc value>    ::=   <vdc value> {greater than or less than 0}

<non-negative vdc value>      ::=   <vdc value> {greater than or equal to 0}

<close type> ::= <PIE>
             | <CHORD>

<elliptical element>    ::=   <ELLIPSE>
                <point>(3)
             | <ELLIPTICAL ARC>
                <point>(3)
                <valid vdc vector>(2)
             | <ELLIPTICAL ARC CLOSE>
                <point>(3)
                <valid vdc vector>(2)
                <close type>
```

A.3.6 Attribute elements

```
<primitive attribute element> ::=   <line attribute element>
             | <marker attribute element>
             | <text attribute element>
             | <filled-area attribute element>
             | <aspect source flags>

<line attribute element>      ::=   <LINE BUNDLE INDEX>
                <positive index>
```

```
             | <LINE TYPE>
                 <index>
             | <LINE WIDTH>
                 <size value>
             | <LINE COLOUR>
                 <colour>

<size value> ::= <non-negative vdc value>
             | <non-negative real>

<non-negative real>     ::=   <real>  {greater than or equal to 0}

<marker attribute element>    ::=   <MARKER BUNDLE INDEX>
                 <positive index>
             | <MARKER TYPE>
                 <index>
             | <MARKER SIZE>
                 <size value>
             | <MARKER COLOUR>
                 <colour>

<partial text attribute element>    ::=   <TEXT BUNDLE INDEX>
                 <positive index>
             | <TEXT FONT INDEX>
                 <positive index>
             | <TEXT PRECISION>
                 <text precision enumerated>
             | <CHARACTER EXPANSION FACTOR>
                 <non-negative real>
             | <CHARACTER SPACING>
                 <real>
             | <TEXT COLOUR>
                 <colour>
             | <CHARACTER HEIGHT>
                 <non-negative vdc value>
             | <CHARACTER SET INDEX>
                 <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                 <positive index>
             | <AUXILIARY COLOUR>
                 <colour>
             | <TRANSPARENCY>
                 <off-on indicator enumerated>
             | <escape element>

<text attribute element>      ::=   <TEXT BUNDLE INDEX>
                 <positive index>
             | <TEXT FONT INDEX>
                 <positive index>
             | <TEXT PRECISION>
                 <text precision enumerated>
             | <CHARACTER EXPANSION FACTOR>
                 <non-negative real>
             | <CHARACTER SPACING>
                 <real>
             | <TEXT COLOUR>
                 <colour>
             | <CHARACTER HEIGHT>
                 <non-negative vdc value>
             | <CHARACTER ORIENTATION>
                 <valid vdc vector>(2)
             | <TEXT PATH>
```

```
                <path enumerated>
             | <TEXT ALIGNMENT>
                <horizontal alignment enumerated>
                <vertical alignment enumerated>
                <continuous alignment value> (2)
             | <CHARACTER SET INDEX>
                <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                <positive index>

<text precision enumerated>   ::=   <STRING>
             | <CHARACTER>
             | <STROKE>

<path enumerated> ::=   <RIGHT>
             | <LEFT>
             | <UP>
             | <DOWN>

<horizontal alignment enumerated>   ::=   <NORMAL HORIZONTAL>
             | <LEFT>
             | <CENTRE>
             | <RIGHT>
             | <CONTINUOUS HORIZONTAL>

<vertical alignment enumerated>     ::=   <NORMAL VERTICAL>
             | <TOP>
             | <CAP>
             | <HALF>
             | <BASE>
             | <BOTTOM>
             | <CONTINUOUS VERTICAL>

<continuous alignment value>  ::=   <real>

<filled-area attribute element>     ::=   <FILL BUNDLE INDEX>
                <positive index>
             | <INTERIOR STYLE>
                <interior style enumerated>
             | <FILL COLOUR>
                <colour>
             | <HATCH INDEX>
                <index>
             | <PATTERN INDEX>
                <positive index>
             | <EDGE BUNDLE INDEX>
                <positive index>
             | <EDGE TYPE>
                <index>
             | <EDGE WIDTH>
                <size value>
             | <EDGE COLOUR>
                <colour>
             | <EDGE VISIBILITY>
                <off-on indicator enumerated>
             | <FILL REFERENCE POINT>
                <point>
             | <PATTERN SIZE>
                <valid vdc vector>(2)

<interior style enumerated>   ::=   <HOLLOW>
             | <SOLID>
```

```
             | <PATTERN>
             | <HATCH>
             | <EMPTY>

<colour table element>  ::=   <COLOUR TABLE>
               <starting index>
               <red green blue>+

<pattern table element> ::=   <PATTERN TABLE>
               <positive index>
               <positive integer>(2)
               <local colour precision>
               <colour>(integer1 x integer2)
              {this element has an encoding}
              {dependent parameter}

<starting index>  ::=   <colour index>

<aspect source flags>   ::=   <ASPECT SOURCE FLAGS>
               <asf pair>+

<asf pair>  ::= <asf type enumerated>
              <asf enumerated>

<asf type enumerated>   ::=   <LINE TYPE ASF>
             | <LINE WIDTH ASF>
             | <LINE COLOUR ASF>
             | <MARKER TYPE ASF>
             | <MARKER SIZE ASF>
             | <MARKER COLOUR ASF>
             | <TEXT FONT ASF>
             | <TEXT PRECISION ASF>
             | <CHARACTER EXPANSION FACTOR ASF>
             | <CHARACTER SPACING ASF>
             | <TEXT COLOUR ASF>
             | <INTERIOR STYLE ASF>
             | <FILL COLOUR ASF>
             | <HATCH INDEX ASF>
             | <PATTERN INDEX ASF>
             | <EDGE TYPE ASF>
             | <EDGE WIDTH ASF>
             | <EDGE COLOUR ASF>

<asf enumerated>  ::=   <INDIVIDUAL>
             | <BUNDLED>
```

A.3.7 Escape elements

```
<escape element>  ::=   <ESCAPE>
               <identifier>
               <data record>

<identifier> ::= <integer>
```

A.3.8 External elements

```
<external element>      ::=   <MESSAGE>
               <action flag enumerated>
               <string fixed>
             | <APPLICATION DATA>
               <integer>
               <data record>
```

```
<action flag enumerated>      ::=   <NO>
             | <YES>
```

A.4 Terminal symbols

The following are the terminals in this grammar. Their representation is dependent on the encoding scheme used. In annex A of the subsequent parts of this Standard, these encoding-dependent symbols are further described.

<element name>
<integer>
<real>
<vdc value>
<string>
<string fixed>
<colour index>
<red green blue>
<integer precision value>
<real precision value>
<index precision value>
<colour precision value>
<colour index precision value>
<default colour precision indicator>
<vdc integer precision value>
<vdc real precision value>
<data record>

The CGM extended opcodes are encoding dependent. A complete list of them can be found in the productions for <element name enumerated> below.

The enumerated types are:

<INTEGER>
<REAL>
<ON>
<OFF>
<INDEXED>
<DIRECT>
<ABSTRACT>
<METRIC>
<ABSOLUTE>
<SCALED>
<94 CHAR>
<96 CHAR>
<MULTI-BYTE 94 CHAR>
<MULTI-BYTE 96 CHAR>
<COMPLETE CODE>
<BASIC 7-BIT>
<BASIC 8-BIT>
<EXTENDED 7-BIT>
<EXTENDED 8-BIT>
<INVISIBLE>
<VISIBLE>
<CLOSE INVISIBLE>
<CLOSE VISIBLE>
<PIE>
<CHORD>
<FINAL>
<NOT FINAL>
<INDIVIDUAL>

```
                                   <BUNDLED>
                                   <HOLLOW>
                                   <SOLID>
                                   <PATTERN>
                                   <HATCH>
                                   <EMPTY>
                                   <STRING>
                                   <CHARACTER>
                                   <STROKE>
                                   <LEFT>
                                   <RIGHT>
                                   <UP>
                                   <DOWN>
                                   <NORMAL HORIZONTAL>
                                   <CENTRE>
                                   <CONTINUOUS HORIZONTAL>
                                   <NORMAL VERTICAL>
                                   <TOP>
                                   <CAP>
                                   <HALF>
                                   <BASE>
                                   <BOTTOM>
                                   <CONTINUOUS VERTICAL>
                                   <YES>
                                   <NO>
                                   <LINE TYPE ASF>
                                   <LINE WIDTH ASF>
                                   <LINE COLOUR ASF>
                                   <MARKER TYPE ASF>
                                   <MARKER SIZE ASF>
                                   <MARKER COLOUR ASF>
                                   <TEXT FONT ASF>
                                   <TEXT PRECISION ASF>
                                   <CHARACTER EXPANSION FACTOR ASF>
                                   <CHARACTER SPACING ASF>
                                   <TEXT COLOUR ASF>
                                   <INTERIOR STYLE ASF>
                                   <HATCH INDEX ASF>
                                   <PATTERN INDEX ASF>
                                   <FILL COLOUR ASF>
                                   <EDGE TYPE ASF>
                                   <EDGE WIDTH ASF>
                                   <EDGE COLOUR ASF>
                                   <DRAWING SET>
                                   <DRAWING PLUS CONTROL SET>

<element name enumerated>     ::=   <BEGIN METAFILE>
             | <END METAFILE>
             | <BEGIN PICTURE>
             | <BEGIN PICTURE BODY>
             | <END PICTURE>
             | <METAFILE VERSION>
             | <METAFILE DESCRIPTION>
             | <VDC TYPE>
             | <INTEGER PRECISION>
             | <REAL PRECISION>
             | <INDEX PRECISION>
             | <COLOUR PRECISION>
             | <COLOUR INDEX PRECISION>
             | <MAXIMUM COLOUR INDEX>
             | <COLOUR VALUE EXTENT>
```

```
| <METAFILE ELEMENT LIST>
| <METAFILE DEFAULTS REPLACEMENT>
| <FONT LIST>
| <CHARACTER SET LIST>
| <CHARACTER CODING ANNOUNCER>
| <SCALING MODE>
| <COLOUR SELECTION MODE>
| <LINE WIDTH SPECIFICATION MODE>
| <MARKER SIZE SPECIFICATION MODE>
| <EDGE WIDTH SPECIFICATION MODE>
| <VDC EXTENT>
| <BACKGROUND COLOUR>
| <VDC INTEGER PRECISION>
| <VDC REAL PRECISION>
| <AUXILIARY COLOUR>
| <TRANSPARENCY>
| <CLIP RECTANGLE>
| <CLIP INDICATOR>
| <POLYLINE>
| <DISJOINT POLYLINE>
| <POLYMARKER>
| <TEXT>
| <RESTRICTED TEXT>
| <APPEND TEXT>
| <POLYGON>
| <POLYGON SET>
| <CELL ARRAY>
| <GDP>
| <RECTANGLE>
| <CIRCLE>
| <CIRCULAR ARC 3 POINT>
| <CIRCULAR ARC 3 POINT CLOSE>
| <CIRCULAR ARC CENTRE>
| <CIRCULAR ARC CENTRE CLOSE>
| <ELLIPSE>
| <ELLIPTICAL ARC>
| <ELLIPTICAL ARC CLOSE>
| <LINE BUNDLE INDEX>
| <LINE TYPE>
| <LINE WIDTH>
| <LINE COLOUR>
| <MARKER BUNDLE INDEX>
| <MARKER TYPE>
| <MARKER SIZE>
| <MARKER COLOUR>
| <TEXT BUNDLE INDEX>
| <TEXT FONT INDEX>
| <TEXT PRECISION>
| <CHARACTER EXPANSION FACTOR>
| <CHARACTER SPACING>
| <TEXT COLOUR>
| <CHARACTER HEIGHT>
| <CHARACTER ORIENTATION>
| <TEXT PATH>
| <TEXT ALIGNMENT>
| <CHARACTER SET INDEX>
| <ALTERNATE CHARACTER SET INDEX>
| <FILL BUNDLE INDEX>
| <INTERIOR STYLE>
| <FILL COLOUR>
| <HATCH INDEX>
| <PATTERN INDEX>
```

```
| <EDGE BUNDLE INDEX>
| <EDGE TYPE>
| <EDGE WIDTH>
| <EDGE COLOUR>
| <EDGE VISIBILITY>
| <FILL REFERENCE POINT>
| <PATTERN TABLE>
| <PATTERN SIZE>
| <COLOUR TABLE>
| <ASPECT SOURCE FLAGS>
| <ESCAPE>
| <MESSAGE>
| <APPLICATION DATA>
```

Annex B
(normative)

Formal Grammar of the functional specification of version 2 metafiles

B.1 Introduction

This grammar is a formal definition of a standard CGM extended syntax for version 2 metafiles. The encoding-independent and the encoding- dependent productions are separated, and there are subsections showing the syntax of each of the standardized encoding schemes. Details on the encoding of terminal symbols can be found in the parts of this International Standard that deal with the particular encoding schemes.

B.2 Notation used

<symbol>	nonterminal
<SYMBOL>	terminal
<symbol>*	0 or more occurrences
<symbol>+	1 or more occurrences
<symbol>o	optional (0 or 1 occurrences)
<symbol>(n)	exactly n occurrences, n=2,3,...
<symbol-1> ::= <symbol-2>	symbol-1 has the syntax of symbol-2
<symbol-1> | <symbol-2>	symbol-1 or alternatively symbol-2
<symbol: meaning>	symbol with the stated meaning
{comment}	explanation of a symbol or a production

B.3 Detailed grammar

B.3.1 Metafile structure

```
<metafile>  ::= <BEGIN METAFILE>
                <metafile identifier>
              <metafile descriptor>
              <metafile contents>*
              <END METAFILE>

<metafile identifier>   ::=   <string fixed>

<metafile contents>     ::=   <extra element>*
              <picture>
              <extra element>*

<extra element> ::=     <external element>
             | <escape element>

<picture>   ::= <BEGIN PICTURE>
                <picture identifier>
              <picture descriptor element>*
              <BEGIN PICTURE BODY>
              <picture content>*
              <END PICTURE>

<picture identifier>    ::=   <string fixed>
```

```
&lt;picture content&gt; ::=   &lt;picture element&gt;
             | &lt;segment&gt;

&lt;picture element&gt; ::=   &lt;control element&gt;
             | &lt;graphical element&gt;
             | &lt;closed figure&gt;
             | &lt;primitive attribute element&gt;
             | &lt;pattern table element&gt;
             | &lt;colour table element&gt;
             | &lt;specification element&gt;
             | &lt;segment control element&gt;
             | &lt;extra element&gt;

&lt;segment&gt;   ::= &lt;BEGIN SEGMENT&gt;
                &lt;segment identifier&gt;
              &lt;segment attribute element&gt;*
              &lt;eligible picture element&gt;*
              &lt;END SEGMENT&gt;

&lt;segment identifier&gt;    ::= &lt;name&gt;

&lt;eligible picture element&gt;    ::=   &lt;control element&gt;
             | &lt;graphical element&gt;
             | &lt;closed figure&gt;
             | &lt;primitive attribute element&gt;
             | &lt;specification element&gt;
             | &lt;segment control element&gt;
             | &lt;extra element&gt;
```

B.3.2 Metafile descriptor elements

```
&lt;metafile descriptor&gt;   ::=   &lt;&lt;optional descriptor element&gt;*
              &lt;version&gt;
              &lt;optional descriptor element&gt;*
              &lt;element list&gt;
              &lt;optional descriptor element&gt;*&gt;
             | &lt;&lt;optional descriptor element&gt;*
              &lt;element list&gt;
              &lt;optional descriptor element&gt;*
              &lt;version&gt;
              &lt;optional descriptor element&gt;*&gt;

&lt;version&gt;   ::= &lt;METAFILE VERSION&gt;
                &lt;integer&gt;

&lt;element list&gt;::= &lt;METAFILE ELEMENT LIST&gt;
                  &lt;MEL item&gt;*

&lt;MEL item&gt;   ::=   &lt;element name&gt;*
                |       &lt;element name shorthand enumerated&gt;*

&lt;element name shorthand
        enumerated&gt;     ::=   &lt;DRAWING SET&gt;
             | &lt;DRAWING PLUS CONTROL SET&gt;
             | &lt;VERSION 2  SET&gt;
             | &lt;EXTENDED PRIMITIVES SET&gt;
             | &lt;VERSION 2 GKSM SET&gt;

&lt;optional descriptor element&gt; ::=   &lt;description&gt;
             | &lt;VDC TYPE&gt;
                &lt;vdc type enumerated&gt;
             | &lt;MAXIMUM COLOUR INDEX&gt;
```

```
                <colour index>
             | <COLOUR VALUE EXTENT>
                <red green blue>(2)
             | <METAFILE DEFAULTS REPLACEMENT>
                <element default>+
             | <FONT LIST>
                <font name>+
             | <CHARACTER SET LIST>
                <character set definition>+
             | <CHARACTER CODING ANNOUNCER>
                <coding technique enumerated>
             | <scalar precision>
             | <MAXIMUM VDC EXTENT>
                <point> (2)
             | &lt;SEGMENT PRIORITY EXTENT&gt;
                <minimum extent>
                <maximum extent>
             | &lt;segment&gt;
             | <extra element>

<description> ::= <METAFILE DESCRIPTION>
                <string fixed>

<vdc type enumerated>   ::=   <INTEGER>
             | <REAL>

<element default> ::=   <control element>
             | <picture descriptor element>
             | <primitive attribute element>
             | &lt;segment attribute element&gt;
             | &lt;segment control element&gt;
             | <extra element>

<font name> ::= <string fixed>

<character set definition>    ::=   <character set enumerated>
              <designation sequence>

<index>     ::= <standard index value>
             | <private index value>

<standard index value>  ::=   <positive integer>
<non-negative integer>  ::=   <integer>     {greater than or equal to 0}
<positive integer>      ::=   <integer>     {greater than 0}
<private index value>   ::=   <negative integer>
<negative integer>      ::=   <integer>     {less than 0}
<positive index>  ::=   <positive integer>

<character set enumerated>    ::=   <94 CHAR>
             | <96 CHAR>
             | <MULTI-BYTE 94 CHAR>
             | <MULTI-BYTE 96 CHAR>
             | <COMPLETE CODE>

<coding technique enumerated> ::=   <BASIC 7-BIT>
             | <BASIC 8-BIT>
             | <EXTENDED 7-BIT>
             | <EXTENDED 8-BIT>

<designation sequence>  ::=   <string fixed>
```

```
<scalar precision>      ::=   <INTEGER PRECISION>
                <integer precision value>
             | <REAL PRECISION>
                <real precision value>
             | <INDEX PRECISION>
                <index precision value>
             | <COLOUR PRECISION>
                <colour precision value>
             | <COLOUR INDEX PRECISION>
                <colour index precision value>
             | <NAME PRECISION>
                <name precision value>
              {these elements have encoding}
              {dependent parameters}

<point>     ::= <vdc value> (2)

<minimum extent>  ::= <non-negative integer>

<maximum extent>  ::= <non-negative integer>
```

NOTE The following order is recommended for the Metafile Descriptor elements: METAFILE VERSION, METAFILE ELEMENT LIST, (possible multiple occurrences of) METAFILE DESCRIPTION.

B.3.3 Picture descriptor elements

```
<picture descriptor element>  ::=   <SCALING MODE>
                <scaling specification mode enumerated>
                <metric scale factor>
             | <VDC EXTENT>
                <point> (2)
             | <DEVICE VIEWPORT>
                <viewport point>(2)
             | <DEVICE VIEWPORT SPECIFICATION MODE
                <VC specifier enumerated>
                <metric scale factor>
             | <DEVICE VIEWPORT MAPPING>
                <isotropy flag  enumerated>
                <horizontal alignment flag  enumerated>
                <vertical alignment flag  enumerated>
             | <BACKGROUND COLOUR>
                <red green blue>
             | <specification element>
             | <representation element>
             | <pattern table element>
             | <colour table element>
             | <extra element>

<specification element> ::=   <COLOUR SELECTION MODE>
                <colour selection mode enumerated>
             | <LINE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>
             | <MARKER SIZE SPECIFICATION MODE>
                <specification mode enumerated>
             | <EDGE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>

<colour selection mode
   enumerated>::= <INDEXED>
             | <DIRECT>
```

```
<scaling specification mode
   enumerated>::= <ABSTRACT>
             | <METRIC>

<metric scale factor>   ::=   <real>

<isotropy flag enumerated>    ::=   <NOT FORCED>
             | <FORCED>

<horizontal alignment flag
    enumerated> ::=     <LEFT>
             | <CENTRE>
             | <RIGHT>

<vertical alignment flag
    enumerated> ::=     <BOTTOM>
             | <CENTRE>
             | <TOP>

<specification mode enumerated>     ::=   <ABSOLUTE>
             | <SCALED>

<viewport point>  ::=   <vc value> (2)

<VC specifier enumerated>     ::=   <FRACTION OF DISPLAY SURFACE>
             | <MILLIMETRES WITH SCALE FACTOR>
             | <PHYSICAL DEVICE COORDINATES>

<representation element>      ::=   <LINE REPRESENTATION>
                <positive index>
                <index>  {line type}
                <size value>  {line width}
                <colour>
             | <MARKER REPRESENTATION>
                <positive index>
                <index>  {marker type}
                <size value>
                <colour>
             | <TEXT REPRESENTATION>
                <positive index>
                <positive index>  {font}
                <text precision enumerated>
                <real>   {character spacing}
                <non-negative real>   {expansion factor}
                <colour>
             | <FILL REPRESENTATION>
                <positive index>
                <interior style enumerated>
                <colour>
                <index> {hatch index}
                <positive index>{pattern index}
             | <EDGE REPRESENTATION>
                <positive index>
                <index>  {edge type}
                <size value>  {edge width}
                <colour>

<size value> ::=<non-negative vdc value>
             | <non-negative real>

<non-negative vdc value>      ::=   <vdc value>  {greater than or equal to 0}
```

```
<non-negative real>     ::=   <real>  {greater than or equal to 0}

<colour>    ::= <colour index>
             | <red green blue>

<text precision enumerated>   ::=   <STRING>
             | <CHARACTER>
             | <STROKE>

<interior style enumerated>   ::=   <HOLLOW>
             | <SOLID>
             | <PATTERN>
             | <HATCH>
             | <EMPTY>
```

B.3.4 Control elements

```
<control element> ::=   <vdc precision>
             | <AUXILIARY COLOUR>
                <colour>
             | <TRANSPARENCY>
                <off-on indicator enumerated>
             | <CLIP RECTANGLE>
                <point>(2)
             | <CLIP INDICATOR>
                <off-on indicator enumerated>
             | <LINE CLIPPING MODE>
                <clip mode enumerated>
             | <MARKER CLIPPING MODE>
                <clip mode enumerated>
             | <EDGE CLIPPING MODE>
                <clip mode enumerated>
             | <NEW REGION>
             | <SAVE PRIMITIVE CONTEXT>
                <context name>
             | <RESTORE PRIMITIVE CONTEXT>
                <context name>

<off-on indicator enumerated> ::=   <OFF>
             | <ON>

<vdc precision> ::=     <VDC INTEGER PRECISION>
                <vdc integer precision value>
             | <VDC REAL PRECISION>
                <vdc real precision value>
               {these elements have encoding}
               {dependent parameters}

<clip mode enumerated>  ::=   <LOCUS>
             | <SHAPE>
             | <LOCUS THEN SHAPE>

<context name>::= <name>
```

B.3.5 Graphical elements

```
<graphical element>     ::=   <polypoint element>
             | <polymarker element>
             | <text element>
             | <cell element>
             | <gdp element>
             | <rectangle element>
```

```
             | <circular element>
             | <elliptical element>

<polymarker element>    ::=   <POLYMARKER>
                <point>
                <point list>

<polypoint element>     ::=   <polyline element>
             | <POLYGON>
                <point>(3)
                <point list>
             | <POLYGON SET>
                <point edge pair>(3)
                <point edge pair list>

<polyline element>      ::=   <POLYLINE>
                <point pair>
                <point list>
             | <DISJOINT POLYLINE>
                <point pair>
                <point pair list>

<point list> ::= <point>*

<point pair list> ::=   <point pair>*

<point pair> ::= <point>(2)

<point edge pair> ::=   <point><edge out flag>

<point edge pair list>  ::=   <point edge pair>*

<edge out flag> ::=     <INVISIBLE>
             | <VISIBLE>
             | <CLOSE INVISIBLE>
             | <CLOSE VISIBLE>

<text element>::= <TEXT>
                <point>
                <text tail>
             | <restricted text element>

<restricted text element>     ::=   <RESTRICTED TEXT>
                <extent>
                <point>
                <text tail>

<extent>    ::= <non-negative vdc value>(2)

<text tail> ::= <final character list>
             | <nonfinal character list>

<final character list>  ::=   <FINAL>
                <string>

<nonfinal character list>     ::=   <NOT FINAL>
                <string>
                <partial text attribute element>*
                <spanned text>

<spanned text>::= <APPEND TEXT>
                <text tail>
```

```
<cell element>::= <CELL ARRAY>
                <point>(3)
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)

              {this element has an encoding}
              {dependent parameter}

<local colour precision>      ::=   <colour precision value>
             | <colour index precision value>
             | <default colour precision indicator>

<gdp element> ::= <GDP>
                <gdp identifier>
                <point list>
                <data record>

<gdp identifier>  ::=   <integer>

<rectangle element>     ::=   <RECTANGLE>
                <point pair>

<circular element>      ::=   <CIRCLE>
                <point>
                <radius>
             | <CIRCULAR ARC 3 POINT>
                <point>(3)
             | <CIRCULAR ARC 3 POINT CLOSE>
                <point>(3)
                <close type>
             | <CIRCULAR ARC CENTRE>
                <point>
                <valid vdc vector>(2)
                <radius>
             | <CIRCULAR ARC CENTRE CLOSE>
                <point>
                <valid vdc vector>(2)
                <radius>
                <close type>
             | <CIRCULAR ARC CENTRE REVERSED>
                <point>
                <valid vdc vector>(2)
                <radius>

<radius>    ::= <non-negative vdc value>

<valid vdc vector>      ::=   <<non-zero vdc value>
                <vdc value>>
             | <<vdc value>
                <non-zero vdc value>>

<non-zero vdc value>    ::=   <vdc value> {greater than or less than 0}

<close type> ::= <PIE>
             | <CHORD>

<elliptical element>    ::=   <ELLIPSE>
                <point>(3)
             | <ELLIPTICAL ARC>
                <point>(3)
                <valid vdc vector>(2)
```

```
            | <ELLIPTICAL ARC CLOSE>
                <point>(3)
                <valid vdc vector>(2)
                <close type>
```

B.3.6 Attribute elements

```
<primitive attribute element> ::=   <line attribute element>
            | <marker attribute element>
            | <text attribute element>
            | <filled-area attribute element>
            | <aspect source flags>
            | <pick identifier>

<line attribute element>      ::=   <LINE BUNDLE INDEX>
                <positive index>
            | <LINE TYPE>
                <index>
            | <LINE WIDTH>
                <size value>
            | <LINE COLOUR>
                <colour>

<marker attribute element>    ::=   <MARKER BUNDLE INDEX>
                <positive index>
            | <MARKER TYPE>
                <index>
            | <MARKER SIZE>
                <size value>
            | <MARKER COLOUR>
                <colour>

<partial text attribute element>    ::=   <TEXT BUNDLE INDEX>
                <positive index>
            | <TEXT FONT INDEX>
                <positive index>
            | <TEXT PRECISION>
                <text precision enumerated>
            | <CHARACTER EXPANSION FACTOR>
                <non-negative real>
            | <CHARACTER SPACING>
                <real>
            | <TEXT COLOUR>
                <colour>
            | <CHARACTER HEIGHT>
                <non-negative vdc value>
            | <CHARACTER SET INDEX>
                <positive index>
            | <ALTERNATE CHARACTER SET INDEX>
                <positive index>
            | <AUXILIARY COLOUR>
                <colour>
            | <TRANSPARENCY>
                <off-on indicator enumerated>
            | <escape element>

<text attribute element>      ::=   <TEXT BUNDLE INDEX>
                <positive index>
            | <TEXT FONT INDEX>
                <positive index>
            | <TEXT PRECISION>
                <text precision enumerated>
```



```
             | <CHARACTER EXPANSION FACTOR>
                <non-negative real>
             | <CHARACTER SPACING>
                <real>
             | <TEXT COLOUR>
                <colour>
             | <CHARACTER HEIGHT>
                <non-negative vdc value>
             | <CHARACTER ORIENTATION>
                <valid vdc vector>(2)
             | <TEXT PATH>
                <path enumerated>
             | <TEXT ALIGNMENT>
                <horizontal alignment enumerated>
                <vertical alignment enumerated>
                <continuous alignment value> (2)
             | <CHARACTER SET INDEX>
                <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                <positive index>

<path enumerated> ::=   <RIGHT>
             | <LEFT>
             | <UP>
             | <DOWN>

<horizontal alignment
     enumerated>  ::=   <NORMAL HORIZONTAL>
             | <LEFT>
             | <CENTRE>
             | <RIGHT>
             | <CONTINUOUS HORIZONTAL>

<vertical alignment enumerated>     ::=   <NORMAL VERTICAL>
             | <TOP>
             | <CAP>
             | <HALF>
             | <BASE>
             | <BOTTOM>
             | <CONTINUOUS VERTICAL>

<continuous alignment value>  ::=   <real>

<filled-area attribute element>     ::=   <FILL BUNDLE INDEX>
                <positive index>
             | <INTERIOR STYLE>
                <interior style enumerated>
             | <FILL COLOUR>
                <colour>
             | <HATCH INDEX>
                <index>
             | <PATTERN INDEX>
                <positive index>
             | <FILL REFERENCE POINT>
                <point>
             | <PATTERN SIZE>
             | <valid vdc vector>(2)
             | <edge attribute element>

<edge attribute element>      ::=   <EDGE BUNDLE INDEX>
                <positive index>
             | <EDGE TYPE>
```

```
                <index>
             | <EDGE WIDTH>
                <size value>
             | <EDGE COLOUR>
                <colour>
             | <EDGE VISIBILITY>
                <off-on indicator enumerated>

<colour table element>  ::=   <COLOUR TABLE>
                <starting index>
                <red green blue>+

<pattern table element> ::=   <PATTERN TABLE>
                <positive index>
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)
                {this element has an encoding}
                {dependent parameter}

<starting index>  ::=   <colour index>

<aspect source flags>   ::=   <ASPECT SOURCE FLAGS>
                <asf pair>+

<asf pair>  ::= <asf type enumerated>
              <asf enumerated>

<asf type enumerated>   ::=   <LINE TYPE ASF>
             | <LINE WIDTH ASF>
             | <LINE COLOUR ASF>
             | <MARKER TYPE ASF>
             | <MARKER SIZE ASF>
             | <MARKER COLOUR ASF>
             | <TEXT FONT ASF>
             | <TEXT PRECISION ASF>
             | <CHARACTER EXPANSION FACTOR ASF>
             | <CHARACTER SPACING ASF>
             | <TEXT COLOUR ASF>
             | <INTERIOR STYLE ASF>
             | <FILL COLOUR ASF>
             | <HATCH INDEX ASF>
             | <PATTERN INDEX ASF>
             | <EDGE TYPE ASF>
             | <EDGE WIDTH ASF>
             | <EDGE COLOUR ASF>

<asf enumerated>  ::=   <INDIVIDUAL>
             | <BUNDLED>

<pick identifier> ::= <PICK IDENTIFIER>
                <name>
```

B.3.7 Closed figure element

```
<closed figure> ::=     <BEGIN FIGURE>
              <eligible element within closed figure>
              <END FIGURE>

<eligible element within closed figure>
            ::= <vdc precision>
             | <AUXILIARY COLOUR>
```

```
                <colour>
             | <TRANSPARENCY>
                <off-on indicator enumerated>
             | <NEW REGION>
             | <polypoint element>
             | <gdp element>
             | <rectangle element>
             | <circular element>
             | <elliptical element>
             | <pointless element>
             | <edge attribute element>
             | <edge asf>
             | <extra element>

<pointless element>     ::=   <CONNECTING EDGE>

<edge asf>  ::= <ASPECT SOURCE FLAGS>
                <edge asf pair>+

<edge asf pair> ::=     <edge asf type enumerated>
                <asf enumerated>

<edge asf type enumerated>    ::=   <EDGE TYPE ASF>
             | <EDGE WIDTH ASF>
             | <EDGE COLOUR ASF>
```

B.3.8 Escape elements

```
<escape element>  ::=   <ESCAPE>
                <identifier>
                <data record>

<identifier> ::= <integer>
```

B.3.9 External elements

```
<external element>      ::=   <MESSAGE>
                <action flag enumerated>
                <string fixed>
             | <APPLICATION DATA>
                <integer>
                <data record>

<action flag enumerated>      ::=   <NO>
             | <YES>
```

B.3.10 Segment elements

<segment control element> ::= <COPY SEGMENT>
<segment identifier>
<copy transformation matrix>
<segment transformation application>
| <inheritance element>

<inheritance element> ::= <INHERITANCE FILTER>
<filter selection list enumerated>*
<selection setting enumerated>
| <CLIP INHERITANCE>
<clip inheritance enumerated>

<segment attribute element> ::= <SEGMENT TRANSFORMATION>
<segment identifier>

<transformation matrix>
| <SEGMENT HIGHLIGHTING>
<segment identifier>
<highlighting enumerated>
| <SEGMENT DISPLAY PRIORITY>
<segment identifier>
<segment display priority>
| <SEGMENT PICK PRIORITY>
<segment identifier>
<segment pick priority>

<copy transformation matrix> ::= <transformation matrix>

<transformation matrix> ::= <2 x 2 matrix of reals>
<2 x 1 matrix of vdcs>

<segment transformation application>
::= <NO>
| <YES>

<filter selection list
enumerated> ::= <attribute and control name enumerated>
| <attribute and control group enumerated>
| <asf name enumerated>
| <asf group enumerated>

<attribute and control
name enumerated> ::= <IH LINE BUNDLE INDEX>
| <IH LINE TYPE>
| <IH LINE WIDTH>
| <IH LINE COLOUR>
| <IH LINE CLIPPING MODE>
| <IH MARKER BUNDLE INDEX>
| <IH MARKER TYPE>
| <IH MARKER SIZE>
| <IH MARKER COLOUR>
| <IH MARKER CLIPPING MODE>
| <IH TEXT BUNDLE INDEX>
| <IH TEXT FONT INDEX>
| <IH TEXT PRECISION>
| <IH CHARACTER EXPANSION FACTOR>
| <IH CHARACTER SPACING>
| <IH TEXT COLOUR>
| <IH CHARACTER HEIGHT>
| <IH CHARACTER ORIENTATION>
| <IH TEXT PATH>
| <IH TEXT ALIGNMENT>
| <IH FILL BUNDLE INDEX>
| <IH INTERIOR STYLE>
| <IH FILL COLOUR>
| <IH HATCH INDEX>
| <IH PATTERN INDEX>
| <IH EDGE BUNDLE INDEX>
| <IH EDGE TYPE>
| <IH EDGE WIDTH>
| <IH EDGE COLOUR>
| <IH EDGE VISIBILITY>
| <IH EDGE CLIPPING MODE>
| <IH FILL REFERENCE POINT>
| <IH PATTERN SIZE>
| <IH AUXILIARY COLOUR>
| <IH TRANSPARENCY>

```
<attribute and control
        group enumerated>     ::=   <LINE ATTRIBUTES>
             | <MARKER ATTRIBUTES>
             | <TEXT PRESENTATION AND PLACEMENT ATTRIBUTES>
             | <TEXT PLACEMENT AND ORIENTATION ATTRIBUTES>
             | <FILL ATTRIBUTES>
             | <EDGE ATTRIBUTES>
             | <PATTERN ATTRIBUTES>
             | <OUTPUT CONTROL>
             | <PICK IDENTIFIER>
             | <ALL ATTRIBUTES AND CONTROL>
             | <ALL>

<selection setting enumerated>      ::=   <STATE LIST>
             | &lt;SEGMENT&gt;

<asf name enumerated>   ::=   <IH LINE TYPE ASF>
             | <IH LINE WIDTH ASF>
             | <IH LINE COLOUR ASF>
             | <IH MARKER TYPE ASF>
             | <IH MARKER SIZE ASF>
             | <IH MARKER COLOUR ASF>
             | <IH TEXT FONT INDEX ASF>
             | <IH TEXT PRECISION ASF>
             | <IH CHARACTER EXPANSION FACTOR ASF>
             | <IH CHARACTER SPACING ASF>
             | <IH TEXT COLOUR ASF>
             | <IH INTERIOR STYLE ASF>
             | <IH FILL COLOUR ASF>
             | <IH HATCH INDEX ASF>
             | <IH PATTERN INDEX ASF>
             | <IH EDGE TYPE ASF>
             | <IH EDGE WIDTH ASF>
             | <IH EDGE COLOUR ASF>

<asf group enumerated>  ::=   <LINE ASFS>
             | <MARKER ASFS>
             | <TEXT ASFS>
             | <FILL ASFS>
             | <EDGE ASFS>
             | <ALL ASFS>

<clip inheritance enumerated> ::=   <STATE LIST>
             | <INTERSECTION>

<highlighting enumerated>     ::=   <NORMAL>
             | <HIGHLIGHTED>

&lt;segment display priority&gt;    ::=   <non-negative integer>

&lt;segment pick priority&gt; ::=   <non-negative integer>
```

B.4 Terminal symbols

The following are the terminals in this grammar. Their representation is dependent on the encoding scheme used. In Annex A of the subsequent parts of this Standard, these encoding-dependent symbols are further described.

<element name>
<integer>
<real>
<vdc value>

<string>
<string fixed>
<colour index>
<red green blue>
<integer precision value>
<real precision value>
<index precision value>
<colour precision value>
<colour index precision value>
<name precision value>
<default colour precision indicator>
<vdc integer precision value>
<vdc real precision value>
<data record>
<name>
<vc value>
<2 x 2 matrix of reals>
<2 x 1 matrix of vdcs>

The CGM extended opcodes are encoding dependent. A complete list of them can be found in the productions for <element name> below.

The enumerated types are:

<INTEGER>
<REAL>
<ON>
<OFF>
<INDEXED>
<DIRECT>
<ABSTRACT>
<METRIC>
<ABSOLUTE>
<SCALED>
<94 CHAR>
<96 CHAR>
<MULTI-BYTE 94 CHAR>
<MULTI-BYTE 96 CHAR>
<COMPLETE CODE>
<BASIC 7-BIT>
<BASIC 8-BIT>
<EXTENDED 7-BIT>
<EXTENDED 8-BIT>
<FRACTION OF DISPLAY SURFACE>
<MILLIMETRES WITH SCALE FACTOR>
<PHYSICAL DEVICE COORDINATES>
<NOT FORCED>
<FORCED>
<LEFT>
<RIGHT>
<CENTRE>
<BOTTOM>
<TOP>
<LOCUS>
<SHAPE>
<LOCUS THEN SHAPE>
<INVISIBLE>
<VISIBLE>
<CLOSE INVISIBLE>
<CLOSE VISIBLE>
<PIE>

<CHORD>
<FINAL>
<NOT FINAL>
<INDIVIDUAL>
<BUNDLED>
<HOLLOW>
<SOLID>
<PATTERN>
<HATCH>
<EMPTY>
<STRING>
<CHARACTER>
<STROKE>
<UP>
<DOWN>
<NORMAL HORIZONTAL>
<CONTINUOUS HORIZONTAL>
<NORMAL VERTICAL>
<CAP>
<HALF>
<BASE>
<CONTINUOUS VERTICAL>
<YES>
<NO>
<LINE TYPE ASF>
<LINE WIDTH ASF>
<LINE COLOUR ASF>
<MARKER TYPE ASF>
<MARKER SIZE ASF>
<MARKER COLOUR ASF>
<TEXT FONT ASF>
<TEXT PRECISION ASF>
<CHARACTER EXPANSION FACTOR ASF>
<CHARACTER SPACING ASF>
<TEXT COLOUR ASF>
<INTERIOR STYLE ASF>
<HATCH INDEX ASF>
<PATTERN INDEX ASF>
<FILL COLOUR ASF>
<EDGE TYPE ASF>
<EDGE WIDTH ASF>
<EDGE COLOUR ASF>
<LINE ATTRIBUTES>
<MARKER ATTRIBUTES>
<TEXT PRESENTATION AND PLACEMENT ATTRIBUTES>
<TEXT PLACEMENT AND ORIENTATION ATTRIBUTES>
<FILL ATTRIBUTES>
<EDGE ATTRIBUTES>
<PATTERN ATTRIBUTES>
<OUTPUT CONTROL>
<PICK IDENTIFIER>
<ALL ATTRIBUTES AND CONTROL>
<ALL>
<LINE BUNDLE INDEX>
<LINE TYPE>
<LINE WIDTH>
<LINE COLOUR>
<LINE CLIPPING MODE>
<MARKER BUNDLE INDEX>
<MARKER TYPE>
<MARKER SIZE>

<MARKER COLOUR>
<MARKER CLIPPING MODE>
<TEXT BUNDLE INDEX>
<TEXT FONT INDEX>
<TEXT PRECISION>
<CHARACTER EXPANSION FACTOR>
<CHARACTER SPACING>
<TEXT COLOUR>
<CHARACTER HEIGHT>
<CHARACTER ORIENTATION>
<TEXT PATH>
<TEXT ALIGNMENT>
<FILL BUNDLE INDEX>
<INTERIOR STYLE>
<FILL COLOUR>
<HATCH INDEX>
<PATTERN INDEX>
<EDGE BUNDLE INDEX>
<EDGE TYPE>
<EDGE WIDTH>
<EDGE COLOUR>
<EDGE VISIBILITY>
<EDGE CLIPPING MODE>
<FILL REFERENCE POINT>
<PATTERN SIZE>
<AUXILIARY COLOUR>
<TRANSPARENCY>
<STATE LIST>
<INTERSECTION>
<SEGMENT>
<LINE ASFS>
<MARKER ASFS>
<TEXT ASFS>
<FILL ASFS>
<EDGE ASFS>
<ALL ASFS>
<NORMAL>
<HIGHLIGHTED>
<DRAWING SET>
<DRAWING PLUS CONTROL SET>
<VERSION 2 SET>
<EXTENDED PRIMITIVES SET>
<VERSION 2 GKSM SET>
<IH LINE BUNDLE INDEX>
<IH LINE TYPE>
<IH LINE WIDTH>
<IH LINE COLOUR>
<IH LINE CLIPPING MODE>
<IH MARKER BUNDLE INDEX>
<IH MARKER TYPE>
<IH MARKER SIZE>
<IH MARKER COLOUR>
<IH MARKER CLIPPING MODE>
<IH TEXT BUNDLE INDEX>
<IH TEXT FONT INDEX>
<IH TEXT PRECISION>
<IH CHARACTER EXPANSION FACTOR>
<IH CHARACTER SPACING>
<IH TEXT COLOUR>
<IH CHARACTER HEIGHT>
<IH CHARACTER ORIENTATION>

```
                              <IH TEXT PATH>
                              <IH TEXT ALIGNMENT>
                              <IH FILL BUNDLE INDEX>
                              <IH INTERIOR STYLE>
                              <IH FILL COLOUR>
                              <IH HATCH INDEX>
                              <IH PATTERN INDEX>
                              <IH EDGE BUNDLE INDEX>
                              <IH EDGE TYPE>
                              <IH EDGE WIDTH>
                              <IH EDGE COLOUR>
                              <IH EDGE VISIBILITY>
                              <IH EDGE CLIPPING MODE>
                              <IH FILL REFERENCE POINT>
                              <IH PATTERN SIZE>
                              <IH AUXILIARY COLOUR>
                              <IH TRANSPARENCY>
                              <IH LINE TYPE ASF>
                              <IH LINE WIDTH ASF>
                              <IH LINE COLOUR ASF>
                              <IH MARKER TYPE ASF>
                              <IH MARKER SIZE ASF>
                              <IH MARKER COLOUR ASF>
                              <IH TEXT FONT INDEX ASF>
                              <IH TEXT PRECISION ASF>
                              <IH CHARACTER EXPANSION FACTOR ASF>
                              <IH CHARACTER SPACING ASF>
                              <IH TEXT COLOUR ASF>
                              <IH INTERIOR STYLE ASF>
                              <IH FILL COLOUR ASF>
                              <IH HATCH INDEX ASF>
                              <IH PATTERN INDEX ASF>
                              <IH EDGE TYPE ASF>
                              <IH EDGE WIDTH ASF>
                              <IH EDGE COLOUR ASF>

<element name>::= <BEGIN METAFILE>
             | <END METAFILE>
             | <BEGIN PICTURE>
             | <BEGIN PICTURE BODY>
             | <END PICTURE>
             | <BEGIN SEGMENT>
             | <END SEGMENT>
             | <BEGIN FIGURE>
             | <END FIGURE>
             | <METAFILE VERSION>
             | <METAFILE DESCRIPTION>
             | <VDC TYPE>
             | <INTEGER PRECISION>
             | <REAL PRECISION>
             | <INDEX PRECISION>
             | <COLOUR PRECISION>
             | <COLOUR INDEX PRECISION>
             | <NAME PRECISION>
             | <MAXIMUM COLOUR INDEX>
             | <COLOUR VALUE EXTENT>
             | <METAFILE ELEMENT LIST>
             | <METAFILE DEFAULTS REPLACEMENT>
             | <FONT LIST>
             | <CHARACTER SET LIST>
             | <CHARACTER CODING ANNOUNCER>
```

```
| <MAXIMUM VDC EXTENT>
| &lt;SEGMENT PRIORITY EXTENT&gt;
| <SCALING MODE>
| <COLOUR SELECTION MODE>
| <LINE WIDTH SPECIFICATION MODE>
| <MARKER SIZE SPECIFICATION MODE>
| <EDGE WIDTH SPECIFICATION MODE>
| <VDC EXTENT>
| <BACKGROUND COLOUR>
| <DEVICE VIEWPORT>
| <DEVICE VIEWPORT SPECIFICATION MODE>
| <DEVICE VIEWPORT MAPPING>
| <LINE REPRESENTATION>
| <MARKER REPRESENTATION>
| <TEXT REPRESENTATION>
| <FILL REPRESENTATION>
| <EDGE REPRESENTATION>
| <VDC INTEGER PRECISION>
| <VDC REAL PRECISION>
| <AUXILIARY COLOUR>
| <TRANSPARENCY>
| <CLIP RECTANGLE>
| <CLIP INDICATOR>
| <LINE CLIPPING MODE>
| <MARKER CLIPPING MODE>
| <EDGE CLIPPING MODE>
| <NEW REGION>
| <SAVE PRIMITIVE CONTEXT>
| <RESTORE PRIMITIVE CONTEXT>
| <POLYLINE>
| <DISJOINT POLYLINE>
| <POLYMARKER>
| <TEXT>
| <RESTRICTED TEXT>
| <APPEND TEXT>
| <POLYGON>
| <POLYGON SET>
| <CELL ARRAY>
| <GDP>
| <RECTANGLE>
| <CIRCLE>
| <CIRCULAR ARC 3 POINT>
| <CIRCULAR ARC 3 POINT CLOSE>
| <CIRCULAR ARC CENTRE>
| <CIRCULAR ARC CENTRE CLOSE>
| <CIRCULAR ARC CENTRE REVERSED>
| <ELLIPSE>
| <ELLIPTICAL ARC>
| <ELLIPTICAL ARC CLOSE>
| <CONNECTING EDGE>
| <LINE BUNDLE INDEX>
| <LINE TYPE>
| <LINE WIDTH>
| <LINE COLOUR>
| <MARKER BUNDLE INDEX>
| <MARKER TYPE>
| <MARKER SIZE>
| <MARKER COLOUR>
| <TEXT BUNDLE INDEX>
| <TEXT FONT INDEX>
| <TEXT PRECISION>
| <CHARACTER EXPANSION FACTOR>
```

| <CHARACTER SPACING>
| <TEXT COLOUR>
| <CHARACTER HEIGHT>
| <CHARACTER ORIENTATION>
| <TEXT PATH>
| <TEXT ALIGNMENT>
| <CHARACTER SET INDEX>
| <ALTERNATE CHARACTER SET INDEX>
| <FILL BUNDLE INDEX>
| <INTERIOR STYLE>
| <FILL COLOUR>
| <HATCH INDEX>
| <PATTERN INDEX>
| <EDGE BUNDLE INDEX>
| <EDGE TYPE>
| <EDGE WIDTH>
| <EDGE COLOUR>
| <EDGE VISIBILITY>
| <FILL REFERENCE POINT>
| <PATTERN TABLE>
| <PATTERN SIZE>
| <COLOUR TABLE>
| <ASPECT SOURCE FLAGS>
| <PICK IDENTIFIER>
| <COPY SEGMENT>
| <INHERITANCE FILTER>
| <CLIP INHERITANCE>
| <SEGMENT TRANSFORMATION>
| <SEGMENT HIGHLIGHTING>
| <SEGMENT DISPLAY PRIORITY>
| <SEGMENT PICK PRIORITY>
| <ESCAPE>
| <MESSAGE>
| <APPLICATION DATA>

Annex C
(normative)

Formal grammar of the functional specification of version 3 metafiles

C.1 Introduction

This grammar is a formal definition of a standard CGM extended syntax for Version 3 metafiles. The encoding-independent and the encoding-dependent productions are separated, and there are subsections showing the syntax of each of the standardized encoding schemes. Details on the encoding of terminal symbols can be found in the parts of this International Standard that deal with the particular encoding schemes.

C.2 Definitions

C.2.1 Notation Used

<symbol>	nonterminal
<SYMBOL>	terminal
<symbol>*	0 or more occurrences
<symbol>+	1 or more occurrences
<symbol>o	optional (0 or 1 occurrences)
<symbol>(n)	exactly n occurrences, n non-negative
<symbol-1> ::= <symbol-2>	symbol-1 has the syntax of symbol-2
<symbol-1> | <symbol-2>	symbol-1 or alternatively symbol-2
<symbol: meaning>	symbol with the stated meaning
{comment}	explanation of a symbol or a production

C.2.2 Structured Data Records

```
<structured data record>      ::=   <member>*


<member>    ::= <typed sequence>

<typed sequence>  ::=   <data type index>
            <data element count>
            <data element: of type "data type">(element count)

<data element count>    ::=   <integer: number of elements in list>

<data type index> ::=   <1: SDR>
            | <2: CI>
            | <3: CD>
            | <4: N>
            | <5: E>
            | <6: I>
            | <7: reserved>
            | <8: IF8>
            | <9: IF16>
            | <10: IF32>
            | <11: IX>
            | <12: R>
            | <13: S>
            | <14: SF>
```

```
| <15: VC>
| <16: VDC>
| <17: CCO>
| <18: UI8>
| <19: UI32>
| <20: BS>
| <21: CL>
| <22: UI16>
```

The integer of the "data element count" and the index of the "data type index" are represented respectively at the current Integer Precision and the current Index Precision of the metafile.

NOTE 1 The assigned index values of the "data type index" have been harmonized with the specifications of the Computer Graphics Interface (CGI) standard (ISO/IEC 9636) and with values used in Graphical Registration. The "reserved" values in the sequence of the index values correspond to values in CGI which are not relevant to CGM. The N data type of CGM corresponds to the CSN data type of CGI. The SDR data type of CGM corresponds to the D data type of CGI.

NOTE 2 All of the data types above, with the exception of CL, IF8, IF16, IF32, and UI16 are defined and specified in clause 7. CL denotes "Colour List", and is used to refer to parameters such as *cell colour specifiers* of the CELL ARRAY element. These parameters are eligible in most of the encodings (parts 3, and 4) for compression techniques. UI16 represents Unsigned Integer at a fixed precision equivalent to 16 binary bits. IF8, IF16, and IF32 represent Signed Integers at fixed precisions (not subject to Integer Precision) equivalent respectively to 8, 16, and 32 binary bits.

NOTE 3 The binary and character encodings of the enumerated (E) data type in SDR use integers, (e.g. 0,1,2), whereas the clear text encoding of the enumerated data type traditionally uses words (e.g. 'on', 'off'). This difference may prevent blind intertranslation between encodings. It is recommended that the use of enumerated (E) data type in SDR coding for ESCAPE, APPLICATION DATA and GDP elements be avoided. Type index (I) should be used instead. Although a recommended practice, there are some existing registered items which use the word-encoded method for E within SDR.

C.3 Detailed Grammar

C.3.1 Metafile structure

```
<metafile>  ::= <BEGIN METAFILE>
                <metafile identifier>
              <metafile descriptor>
              <metafile contents>*
              <END METAFILE>

<metafile identifier>   ::=   <string fixed>

<metafile contents>     ::=   <extra element>*
              <picture>
              <extra element>*

<extra element> ::=     <external element>
             | <escape element>

<picture>   ::= <BEGIN PICTURE>
                <picture identifier>
              <picture descriptor element>*
              <BEGIN PICTURE BODY>
              <picture content>*
              <END PICTURE>

<picture identifier>    ::=   <string fixed>

<picture content> ::=   <picture element>
             | &lt;segment&gt;

<picture element> ::=   <control element>
             | <graphical element>
```

| <closed figure>
| <primitive attribute element>
| <pattern table element>
| <colour table element>
| <specification element>
| <segment control element>
| <compound line>
| <compound text path>
| <protection region>
| <tile array>
| <extra element>

<segment> ::= <BEGIN SEGMENT>
<segment identifier>
<segment attribute element>*
<eligible picture element>*
<compound line>*
<compound text path>*
<protection region>*
<END SEGMENT>

<segment identifier> ::= <name>

<eligible picture element> ::= <control element>
| <graphical element>
| <closed figure>
| <primitive attribute element>
| <specification element>
| <segment control element>
| <extra element>

<closed figure> ::= <BEGIN FIGURE>
<eligible element within closed figure>*
<END FIGURE>

<eligible element within closed figure>
::= <vdc precision>
| <AUXILIARY COLOUR>
<colour>
| <TRANSPARENCY>
<off-on indicator enumerated>
| <NEW REGION>
| <polypoint element>
| <gdp element>
| <rectangle element>
| <circular element>
| <elliptical element>
| <pointless element>
| <curve element>
| <edge attribute element>
| <edge asf>
| <extra element>

<pointless element> ::= <CONNECTING EDGE>

<edge asf> ::= <ASPECT SOURCE FLAGS>
<edge asf pair>+

<edge asf pair> ::= <edge asf type enumerated>
<asf enumerated>

```
<edge asf type enumerated>    ::=   <EDGE TYPE ASF>
             | <EDGE WIDTH ASF>
             | <EDGE COLOUR ASF>

<compound line> ::=     <BEGIN COMPOUND LINE>
              <eligible element within compound path>*
              <END COMPOUND LINE>

<eligible element within compound path>
            ::= <vdc precision>
             | <polyline element>
             | <gdp element>
             | <circular arc element>
             | <elliptical arc element>
             | <curve element>
             | <extra element>

<compound text path>    ::=   <BEGIN COMPOUND TEXT PATH>
              <eligible element within compound path>*
              <END COMPOUND TEXT PATH>

<protection region>     ::=   <BEGIN PROTECTION REGION>
                <positive index>
              <eligible element within protection region>*
              <END PROTECTION REGION>

<eligible element within protection region>
            ::= <vdc precision>
             | <NEW REGION>
             | <polypoint element>
             | <gdp element>
             | <rectangle element>
             | <circular element>
             | <elliptical element>
             | <curve element>
             | <extra element>

<tile array> ::= <BEGIN TILE ARRAY>
                <position>
                <cell path direction enumerated>
                <line progression direction enumerated>
                <number of tiles in path direction>
                <number of tiles in line direction>
                <number of cells/tile in path direction>
                <number of cells/tile in line direction>
                <cell size in path direction>
                <cell size in line direction>
                <image offset in path direction>
                <image offset in line direction>
                <image number of cells in path direction>
                <image number of cells in line direction>
              <eligible element within tile array>*
              <END TILE ARRAY>

<position>  ::= <point>

<cell path direction enumerated>    ::=   <0 DEGREES>
             | <90 DEGREES>
             | <180 DEGREES>
             | <270 DEGREES>
```

```
<line progression direction enumerated>   ::=   <90 DEGREES>
             | <270 DEGREES>

<number of tiles in path direction> ::=   <positive integer>

<number of tiles in line direction> ::=   <positive integer>

<number of cells/tile in path direction>  ::=   <positive integer>

<number of cells/tile in line direction>  ::=   <positive integer>

<cell size in path direction> ::=   <positive real>

<cell size in line direction> ::=   <positive real>

<image offset in path direction>    ::=   <non-negative integer>

<image offset in line direction>    ::=   <non-negative integer>

<image number of cells in path direction>
            ::= <positive integer>

<image number of cells in line direction>
            ::= <positive integer>

<eligible element within tile array>
            ::= <tile element>
             | <extra element>

<tile element>::= <BITONAL TILE>
                <compression type>
                <row padding indicator>
                <cell background colour>
                <cell foreground colour>
                <method-specific parameters>
                <compressed colour specifiers>
             | <TILE>
                <compression type>
                <row padding indicator>
                <cell colour precision>
                <method-specific parameters>
                <compressed colour specifiers>

              {these elements each have an }
              {encoding dependent parameter}

<compression type>      ::=   <non-negative index>

<row padding indicator> ::=   <non-negative integer>

<cell background colour>      ::=   <colour>

<cell foreground colour>      ::=   <colour>

<method-specific parameters>  ::=   <structured data record>

<cell colour precision> ::=   <colour precision value>
             | <colour index precision value>
             | <default colour precision indicator>

<compressed colour specifiers>      ::=   <bitstream>
```

C.3.2 Metafile descriptor elements

```
<metafile descriptor>   ::=   <<optional descriptor element>*
              <version>
              <optional descriptor element>*
              <element list>
              <optional descriptor element>*>
            | <<optional descriptor element>*
              <element list>
              <optional descriptor element>*
              <version>
              <optional descriptor element>*>

<version>   ::= <METAFILE VERSION>
                <integer>

<element list>::= <METAFILE ELEMENT LIST>
                  <MEL item>*

<MEL item>   ::=  <element name>*
               |       <element name shorthand enumerated>*

<element name shorthand
        enumerated>     ::=   <DRAWING SET>
            | <DRAWING PLUS CONTROL SET>
            | <VERSION 2 SET>
            | <EXTENDED PRIMITIVES SET>
            | <VERSION 2 GKSM SET>
            | <VERSION 3 SET>

<optional descriptor element> ::=   <description>
            | <VDC TYPE>
                <vdc type enumerated>
            | <MAXIMUM COLOUR INDEX>
                <colour index>
            | <COLOUR VALUE EXTENT>
                <colour value mapping specifier>
            | <METAFILE DEFAULTS REPLACEMENT>
                <element default>+
            | <FONT LIST>
                <font name>+
            | <CHARACTER SET LIST>
                <character set definition>+
            | <CHARACTER CODING ANNOUNCER>
                <coding technique enumerated>
            | <scalar precision>
            | <MAXIMUM VDC EXTENT>
                <point> (2)
            | &lt;SEGMENT PRIORITY EXTENT>
                <minimum extent>
                <maximum extent>
            | &lt;segment&gt;
            | <COLOUR MODEL>
                <positive index>
            | <COLOUR CALIBRATION>
                <calibration selection>
                <calibration data>
            | <FONT PROPERTIES>
                <font property 3-tuple>+
            | <GLYPH MAPPING>
                <character set index>
                <character set type enumerated>
```

<designation sequence>
<octets per code>
<glyph source identifier>
<glyph code association>
| <SYMBOL LIBRARY LIST>
<symbol library name>+
| <extra element>

<description> ::= <METAFILE DESCRIPTION>
<string fixed>

<vdc type enumerated> ::= <INTEGER>
| <REAL>

<element default> ::= <control element>
| <picture descriptor element>
| <primitive attribute element>
| <segment attribute element>
| <segment control element>
| <extra element>

<font name> ::= <string fixed>

<character set definition> ::= <character set enumerated>
<designation sequence>

<index> ::= <standard index value>
| <private index value>

<character set index> ::= <standard index value>

<standard index value> ::= <positive integer>
<non-negative integer> ::= <integer> {greater than or equal to 0}
<positive integer> ::= <integer> {greater than 0}
<private index value> ::= <negative integer>
<negative integer> ::= <integer> {less than 0}
<positive index> ::= <positive integer>
<positive real> ::= <real> {greater than 0}

<character set enumerated> ::= <94 CHAR>
| <96 CHAR>
| <MULTI-BYTE 94 CHAR>
| <MULTI-BYTE 96 CHAR>
| <COMPLETE CODE>

<coding technique enumerated> ::= <BASIC 7-BIT>
| <BASIC 8-BIT>
| <EXTENDED 7-BIT>
| <EXTENDED 8-BIT>

designation sequence> ::= <string fixed>

<scalar precision> ::= <INTEGER PRECISION>
<integer precision value>
| <REAL PRECISION>
<real precision value>
| <INDEX PRECISION>
<index precision value>
| <COLOUR PRECISION>
<colour precision value>
| <COLOUR INDEX PRECISION>

```
                <colour index precision value>
             | <NAME PRECISION>
                <name precision value>

              {these elements have encoding}
              {dependent parameters}

<point>     ::= <vdc value> (2)

<minimum extent>  ::=   <non-negative integer>

<maximum extent>  ::=   <non-negative integer>

<colour value mapping specifier>    ::=   <colour direct>(2)
             | (<colour scale><colour offset>)(3)

<colour scale>::= <real>

<colour offset> ::=     <real>

<font property 3-tuple> ::=   <property indicator>
              <priority>
              <property value record>

<property indicator>    ::=   <standard index value>

<priority>  ::= <non-negative integer>

<property value record> ::=   <structured data record>

<octets per code> ::=   <positive integer>

<glyph source identifier>     ::=   <standard index value>

<glyph-code association>      ::=   <structured data record>

<calibration selection> ::=   <positive index>

<calibration data>      ::=   <reference white>
              <rgb and rgb-related calibration data>
              <cmyk calibration data>

<reference white> ::=   <real>(3)

<rgb and rgb-related calibration data>    ::=   <3x3 matrix of reals>(2)
              <non-negative integer> {=n}
              <lookup table entry for red>(n)
              <lookup table entry for green>(n)
              <lookup table entry for blue>(n)

<cmyk calibration data> ::=   <positive integer> {=m}
              <cmyk grid location>(m)
              <ciexyz grid location>(m)

<lookup table entry for red>  ::=   <cco value>(2)

<lookup table entry for green>      ::=   <cco value>(2)

<lookup table entry for blue> ::=   <cco value>(2)

<cmyk grid location>    ::=   <colour direct>

<ciexyz grid location>  ::=   <real>(3)
```

```
symbol library name>    ::=   <string fixed>
```

NOTE The following order is recommended for the Metafile Descriptor elements: METAFILE VERSION, METAFILE ELEMENT LIST, (possible multiple occurrences of) METAFILE DESCRIPTION.

C.3.3 Picture descriptor elements

```
<picture descriptor element>  ::=   <SCALING MODE>
                <scaling specification mode enumerated>
                <metric scale factor>
             | <VDC EXTENT>
                <point> (2)
             | <DEVICE VIEWPORT>
                <viewport point>(2)
             | <DEVICE VIEWPORT SPECIFICATION MODE>
                <VC specifier enumerated>
                <metric scale factor>
             | <DEVICE VIEWPORT MAPPING>
                <isotropy flag  enumerated>
                <horizontal alignment flag  enumerated>
                <vertical alignment flag  enumerated>
             | <BACKGROUND COLOUR>
                <colour direct>
             | <specification element>
             | <representation element>
             | <pattern table element>
             | <colour table element>
             | <extra element>
             | <definition element>
             | &lt;segment&gt;

<colour>    ::= <colour index>
             | <colour direct>

<specification element> ::=   <COLOUR SELECTION MODE>
                <colour selection mode enumerated>
             | <LINE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>
             | <MARKER SIZE SPECIFICATION MODE>
                <specification mode enumerated>
             | <EDGE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>
             | <INTERIOR STYLE SPECIFICATION MODE>
                <specification mode enumerated>

<definition element>    ::=   <LINE AND EDGE TYPE DEFINITION>
                <negative index>
                <positive size value>
                <non-negative integer>+
             | <HATCH STYLE DEFINITION>
                <negative index>
                <style indicator enumerated>
                <valid size vector>(2)
                <positive size value>
                <number of hatch lines>
                <gap widths>
                <line types>
             | <GEOMETRIC PATTERN DEFINITION>
                <positive index>
                &lt;segment identifier&gt;
                <point pair>
```

```
<style indicator enumerated>  ::=   <PARALLEL>
             | <CROSSHATCH>

<number of hatch lines> ::=   <positive integer> {=n}

<gap widths> ::= <positive integer>(n)

<line types> ::= <index>(n)

<size value> ::= <non-negative vdc value>
             | <non-negative real>

<valid size vector>     ::=   <valid real vector>
             | <valid vdc vector>

<valid real vector>     ::=   <<non-zero real value>
                <real value>>
             | <<real value>
                <non-zero real value>>

<valid vdc vector>      ::=   <<non-zero vdc value>
                <vdc value>>
             | <<vdc value>
                <non-zero vdc value>>

<non-zero vdc value>    ::=   <vdc value> {greater than or less than 0}

<non-zero real value>   ::=   <real value> {greater than or less than 0}

<non-zero size value>   ::=   <non-zero vdc value>
             | <non-zero real value>

<positive size value>   ::=   <positive vdc value>
             | <positive real value>

<positive vdc value>    ::=   <vdc value>  {greater than 0}

<non-negative vdc value>      ::=   <vdc value>  {greater than or equal to 0}

<non-negative real>     ::=   <real>  {greater than or equal to 0}

<colour selection mode
      enumerated> ::=   <INDEXED>
             | <DIRECT>

<scaling specification mode
   enumerated>::= <ABSTRACT>
             | <METRIC>

<metric scale factor>   ::=   <real>

<isotropy flag enumerated>    ::=   <NOT FORCED>
             | <FORCED>

<horizontal alignment flag
    enumerated> ::=     <LEFT>
             | <CENTRE>
             | <RIGHT>

<vertical alignment flag
    enumerated> ::=     <BOTTOM>
```

```
             | <CENTRE>
             | <TOP>

<specification mode enumerated>     ::=   <ABSOLUTE>
             | <SCALED>
             | <FRACTIONAL>
             | <MILLIMETRES>

<viewport point>  ::=   <vc value> (2)

<VC specifier enumerated>     ::=   <FRACTION OF DISPLAY SURFACE>
             | <MILLIMETRES WITH SCALE FACTOR>
             | <PHYSICAL DEVICE COORDINATES>

<representation element>      ::=   <LINE REPRESENTATION>
                <positive index>
                <index>  {line type}
                <size value>  {line width}
                <colour>
             | <MARKER REPRESENTATION>
                <positive index>
                <index>  {marker type}
                <size value>
                <colour>
             | <TEXT REPRESENTATION>
                <positive index>
                <positive index>  {font}
                <text precision enumerated>
                <real>   {character spacing}
                <non-negative real>   {expansion factor}
                <colour>
             | <FILL REPRESENTATION>
                <positive index>
                <interior style enumerated>
                <colour>
                <index> {hatch index}
                <positive index>{pattern index}
             | <EDGE REPRESENTATION>
                <positive index>
                <index>  {edge type}
                <size value>  {edge width}
                <colour>

<text precision enumerated>   ::=   <STRING>
             | <CHARACTER>
             | <STROKE>

<interior style enumerated>   ::=   <HOLLOW>
             | <SOLID>
             | <PATTERN>
             | <HATCH>
             | <EMPTY>
             | <GEOMETRIC PATTERN>
             | <INTERPOLATED>
```

C.3.4 Control elements

```
<control element> ::=   <vdc precision>
             | <AUXILIARY COLOUR>
                <colour>
             | <TRANSPARENCY>
                <off-on indicator enumerated>
```

```
             | <CLIP RECTANGLE>
                 <point>(2)
             | <CLIP INDICATOR>
                 <off-on indicator enumerated>
             | <LINE CLIPPING MODE>
                 <clip mode enumerated>
             | <MARKER CLIPPING MODE>
                 <clip mode enumerated>
             | <EDGE CLIPPING MODE>
                 <clip mode enumerated>
             | <NEW REGION>
             | <SAVE PRIMITIVE CONTEXT>
                 <context name>
             | <RESTORE PRIMITIVE CONTEXT>
                 <context name>
             | <PROTECTION REGION INDICATOR>
                 <region index>
                 <region indicator>
             | <GENERALIZED TEXT PATH MODE>
                 <text path mode enumerated>
             | <MITRE LIMIT>
                 <non-negative-real>
             | <TRANSPARENT CELL COLOUR>
                 <transparency indicator enumerated>
                 <colour>

<off-on indicator enumerated> ::=   <OFF>
             | <ON>

<vdc precision> ::=     <VDC INTEGER PRECISION>
                 <vdc integer precision value>
             | <VDC REAL PRECISION>
                 <vdc real precision value>
               {these elements have encoding}
               {dependent parameters}

<clip mode enumerated>  ::=   <LOCUS>
             | <SHAPE>
             | <LOCUS THEN SHAPE>

<context name>::= <name>

<region index>::= <positive index>

<region indicator>      ::=   <positive index>

<text path mode enumerated>   ::=   <OFF>
             | <NON-TANGENTIAL>
             | <AXIS-TANGENTIAL>

<transparency indicator enumerated> ::=   <off-on indicator enumerated>
```

C.3.5 Graphical elements

```
<graphical element>     ::=   <polypoint element>
             | <polymarker element>
             | <text element>
             | <cell element>
             | <gdp element>
             | <rectangle element>
             | <circular element>
             | <elliptical element>
```

```
            | <curve element>
            | <symbol element>

<polymarker element>    ::=   <POLYMARKER>
               <point>
               <point list>

<polypoint element>     ::=   <polyline element>
            | <POLYGON>
               <point>(3)
               <point list>
            | <POLYGON SET>
               <point edge pair>(3)
               <point edge pair list>

<polyline element>      ::=   <POLYLINE>
               <point pair>
               <point list>
            | <DISJOINT POLYLINE>
               <point pair>
               <point pair list>

<point list> ::= <point>*

<point pair list> ::=   <point pair>*

<point pair> ::= <point>(2)

<point edge pair> ::=   <point><edge out flag>

<point edge pair list>  ::=   <point edge pair>*

<edge out flag> ::=     <INVISIBLE>
            | <VISIBLE>
            | <CLOSE INVISIBLE>
            | <CLOSE VISIBLE>

<text element>::= <TEXT>
               <point>
               <text tail>
            | <restricted text element>

<restricted text element>     ::=   <RESTRICTED TEXT>
               <extent>
               <point>
               <text tail>

<extent>    ::= <non-negative vdc value>(2)

<text tail> ::= <final character list>
            | <nonfinal character list>

<final character list>  ::=   <FINAL>
               <string>

<nonfinal character list>     ::=   <NOT FINAL>
               <string>
               <partial text attribute element>*
               <spanned text>

<spanned text>::= <APPEND TEXT>
               <text tail>
```

```
<cell element>::= <CELL ARRAY>
                <point>(3)
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)

              {this element has an encoding}
              {dependent parameter}

<local colour precision>      ::=   <colour precision value>
             | <colour index precision value>
             | <default colour precision indicator>

<gdp element> ::= <GDP>
                <gdp identifier>
                <point list>
                <data record>

<gdp identifier>  ::=   <integer>

<rectangle element>     ::=   <RECTANGLE>
                <point pair>

<circular element>      ::=   <CIRCLE>
                <point>
                <radius>
             | <CIRCULAR ARC 3 POINT CLOSE>
                <point>(3)
                <close type>
             | <CIRCULAR ARC CENTRE CLOSE>
                <point>
                <valid vdc vector>(2)
                <radius>
                <close type>
             | <circular arc element>

<radius>    ::= <non-negative vdc value>

<close type> ::= <PIE>
             | <CHORD>

<circular arc element>  ::=   <CIRCULAR ARC 3 POINT>
                <point>(3)
             | <CIRCULAR ARC CENTRE>
                <point>
                <valid vdc vector>(2)
                <radius>
             | <CIRCULAR ARC CENTRE REVERSED>
                <point>
                <valid vdc vector>(2)
                <radius>

<elliptical element>    ::=   <ELLIPSE>
                <point>(3)
             | <ELLIPTICAL ARC CLOSE>
                <point>(3)
                <valid vdc vector>(2)
                <close type>
                <elliptical arc element>
```

```
<elliptical arc element>      ::=   <ELLIPTICAL ARC>
                <point>(3)
                <valid vdc vector>(2)

<curve element> ::=     <HYPERBOLIC ARC>
                <point>(3)
                <valid vdc vector>(2)
             | <PARABOLIC ARC>
                <point>(3)
             | <NON-UNIFORM B-SPLINE>
                <spline order>
                <number of control points>
                <control points>
                <list of knots>
                <parameter start value>
                <parameter end value>
             | <NON-UNIFORM RATIONAL B-SPLINE>
                <spline order>
                <number of control points>
                <control points>
                <list of knots>
                <parameter start value>
                <parameter end value>
                <weights>
             | <POLYBEZIER>
                <positive index>
                <control point>(4)
                <control point list>

<spline order>::= <positive integer>  {m}

<number of control points>    ::=   <positive integer>  {n≥m}

<control points>  ::=   <point>(n)

<list of knots> ::=     <real>(n+m)

<parameter start value> ::=   <real>

<parameter end value>   ::=   <real>

<weights>   ::= <real>(n)

<control point> ::=     <point>

<control point list>    ::=   <<control point>(4)>*
             | <<control point>(3)>*

<symbol element>  ::=         <POLYSYMBOL>
                <positive index>
                <point>
                <point list>
```

C.3.6 Attribute elements

```
<primitive attribute element> ::=   <line attribute element>
             | <marker attribute element>
             | <text attribute element>
             | <filled-area attribute element>
             | <aspect source flags>
             | <pick identifier>
             | <symbol attribute element>
```

```
<line attribute element>      ::=   <LINE BUNDLE INDEX>
                <positive index>
             | <LINE TYPE>
                <index>
             | <LINE WIDTH>
                <size value>
             | <LINE COLOUR>
                <colour>
             | <LINE CAP>
                <positive index>
                <positive index>
             | <LINE JOIN>
                <positive index>
             | <LINE TYPE CONTINUATION>
                <positive index>
             | <LINE TYPE INITIAL OFFSET>
                <non-negative real>

<marker attribute element>    ::=   <MARKER BUNDLE INDEX>
                <positive index>
             | <MARKER TYPE>
                <index>
             | <MARKER SIZE>
                <size value>
             | <MARKER COLOUR>
                <colour>

<partial text attribute element>    ::=   <TEXT BUNDLE INDEX>
                <positive index>
             | <TEXT FONT INDEX>
                <positive index>
             | <TEXT PRECISION>
                <text precision enumerated>
             | <CHARACTER EXPANSION FACTOR>
                <non-negative real>
             | <CHARACTER SPACING>
                <real>
             | <TEXT COLOUR>
                <colour>
             | <CHARACTER HEIGHT>
                <non-negative vdc value>
             | <CHARACTER SET INDEX>
                <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                <positive index>
             | <TEXT SCORE TYPE>
                <type-indicator pair>+
             | <AUXILIARY COLOUR>
                <colour>
             | <TRANSPARENCY>
                <off-on indicator enumerated>
             | <escape element>

<text attribute element>      ::=   <TEXT BUNDLE INDEX>
                <positive index>
             | <TEXT FONT INDEX>
                <positive index>
             | <TEXT PRECISION>
                <text precision enumerated>
             | <CHARACTER EXPANSION FACTOR>
                <non-negative real>
             | <CHARACTER SPACING>
```

```
                <real>
             | <TEXT COLOUR>
                <colour>
             | <CHARACTER HEIGHT>
                <non-negative vdc value>
             | <CHARACTER ORIENTATION>
                <valid vdc vector>(2)
             | <TEXT PATH>
                <text path mode enumerated>
             | <TEXT ALIGNMENT>
                <horizontal alignment enumerated>
                <vertical alignment enumerated>
                <continuous alignment value> (2)
             | <CHARACTER SET INDEX>
                <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                <positive index>
             | <TEXT SCORE TYPE>
                <type-indicator pair>+
             | <RESTRICTED TEXT TYPE>
                <positive index>

<type-indicator pair>   ::=   <score type index>
              <off-on indicator enumerated>

<path enumerated> ::=   <RIGHT>
             | <LEFT>
             | <UP>
             | <DOWN>

<horizontal alignment
     enumerated>  ::=   <NORMAL HORIZONTAL>
             | <LEFT>
             | <CENTRE>
             | <RIGHT>
             | <CONTINUOUS HORIZONTAL>

<vertical alignment enumerated>     ::=   <NORMAL VERTICAL>
             | <TOP>
             | <CAP>
             | <HALF>
             | <BASE>
             | <BOTTOM>
             | <CONTINUOUS VERTICAL>

<score type index>      ::=   <positive index>

<continuous alignment value>  ::=   <real>

<filled-area attribute element>     ::=   <FILL BUNDLE INDEX>
                <positive index>
             | <INTERIOR STYLE>
                <interior style enumerated>
             | <FILL COLOUR>
                <colour>
             | <HATCH INDEX>
                <index>
             | <PATTERN INDEX>
                <positive index>
             | <FILL REFERENCE POINT>
                <point>
             | <PATTERN SIZE>
```

```
                <valid size vector>(2)
             | <INTERPOLATED INTERIOR>
                <positive index>
                <geometry definition>
                <number of stages>
                <stage designators>
                <reference colour list>
             | <edge attribute element>

<geometry definition>   ::=   <non-zero size value>(2n) {n>0}

<number of stages>      ::=   <positive integer>

<stage designators>     ::=   <real>(m)

<reference colour list> ::=   <colour>(k) {k≥0}
```

NOTE The type of this data is different in this Version 3 grammar, reflecting the new INTERIOR STYLE SPECIFICATION MODE (ISSM) which selects the units to be ‘absolute', ‘scaled', ‘fractional', or ‘mm'. In Version 1 and Version 2 it is <integer> or <real> (reflecting ‘vdc' in clause 7), which is effectively the same as ‘absolute' (the default ISSM).

```
<edge attribute element>      ::=   <EDGE BUNDLE INDEX>
                <positive index>
             | <EDGE TYPE>
                <index>
             | <EDGE WIDTH>
                <size value>
             | <EDGE COLOUR>
                <colour>
             | <EDGE VISIBILITY>
                <off-on indicator enumerated>
             | <EDGE CAP>
                <positive index>
                <positive index>
             | <EDGE JOIN>
                <positive index>
             | <EDGE TYPE CONTINUATION>
                <positive index>
             | <EDGE TYPE INITIAL OFFSET>
                <real>

<colour table element>  ::=   <COLOUR TABLE>
                <starting index>
                <colour direct>+

<starting index>  ::=   <colour index>

<pattern table element> ::=   <PATTERN TABLE>
                <positive index>
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)

                {this element has an encoding}
                {dependent parameter}

<aspect source flags>   ::=   <ASPECT SOURCE FLAGS>
                <asf pair>+

<asf pair>  ::= <asf type enumerated>
              <asf enumerated>
```

```
<asf type enumerated>   ::=   <LINE TYPE ASF>
             | <LINE WIDTH ASF>
             | <LINE COLOUR ASF>
             | <MARKER TYPE ASF>
             | <MARKER SIZE ASF>
             | <MARKER COLOUR ASF>
             | <TEXT FONT ASF>
             | <TEXT PRECISION ASF>
             | <CHARACTER EXPANSION FACTOR ASF>
             | <CHARACTER SPACING ASF>
             | <TEXT COLOUR ASF>
             | <INTERIOR STYLE ASF>
             | <FILL COLOUR ASF>
             | <HATCH INDEX ASF>
             | <PATTERN INDEX ASF>
             | <EDGE TYPE ASF>
             | <EDGE WIDTH ASF>
             | <EDGE COLOUR ASF>

<asf enumerated>  ::=   <INDIVIDUAL>
             | <BUNDLED>

<pick identifier> ::= <PICK IDENTIFIER>
                <name>

<symbol attribute element>    ::=   <SYMBOL LIBRARY INDEX>
                <positive index>
              <SYMBOL COLOUR>
                <colour>
              <SYMBOL SIZE>
                <scale enumerated>
                <positive vdc value>(2)
              <SYMBOL ORIENTATION>
                <valid vdc vector>(2)

<scale enumerated>      ::=   <HEIGHT>
             | <WIDTH>
             | <BOTH>
```

C.3.7 Escape elements

```
<escape element>  ::=   <ESCAPE>
                <identifier>
                <data record>

<identifier> ::= <integer>
```

C.3.8 External elements

```
<external element>      ::=   <MESSAGE>
                <action flag enumerated>
                <string fixed>
             | <APPLICATION DATA>
                <integer>
                <data record>

<action flag enumerated>      ::=   <NO>
             | <YES>
```

C.3.9 Segment elements

<segment control element> ::= <COPY SEGMENT>
<segment identifier>
<copy transformation matrix>
<segment transformation application>
| <inheritance element>

<inheritance element> ::= <INHERITANCE FILTER>
<filter selection list enumerated>*
<selection setting enumerated>
| <CLIP INHERITANCE>
<clip inheritance enumerated>

<segment attribute element> ::= <SEGMENT TRANSFORMATION>
<segment identifier>
<transformation matrix>
| <SEGMENT HIGHLIGHTING>
<segment identifier>
<highlighting enumerated>
| <SEGMENT DISPLAY PRIORITY>
<segment identifier>
<segment display priority>
| <SEGMENT PICK PRIORITY>
<segment identifier>
<segment pick priority>

<copy transformation matrix> ::= <transformation matrix>

<transformation matrix> ::= <2 x 2 matrix of reals>
<2 x 1 matrix of vdcs>

<segment transformation application> ::= <NO>
| <YES>

<filter selection list enumerated> ::= <attribute and control name enumerated>
| <attribute and control group enumerated>
| <asf name enumerated>
| <asf group enumerated>

<attribute and control name enumerated> ::= <IH LINE BUNDLE INDEX>
| <IH LINE TYPE>
| <IH LINE WIDTH>
| <IH LINE COLOUR>
| <IH LINE CLIPPING MODE>
| <IH MARKER BUNDLE INDEX>
| <IH MARKER TYPE>
| <IH MARKER SIZE>
| <IH MARKER COLOUR>
| <IH MARKER CLIPPING MODE>
| <IH TEXT BUNDLE INDEX>
| <IH TEXT FONT INDEX>
| <IH TEXT PRECISION>
| <IH CHARACTER EXPANSION FACTOR>
| <IH CHARACTER SPACING>
| <IH TEXT COLOUR>
| <IH CHARACTER HEIGHT>
| <IH CHARACTER ORIENTATION>
| <IH TEXT PATH>
| <IH TEXT ALIGNMENT>

```
             | <IH FILL BUNDLE INDEX>
             | <IH INTERIOR STYLE>
             | <IH FILL COLOUR>
             | <IH HATCH INDEX>
             | <IH PATTERN INDEX>
             | <IH EDGE BUNDLE INDEX>
             | <IH EDGE TYPE>
             | <IH EDGE WIDTH>
             | <IH EDGE COLOUR>
             | <IH EDGE VISIBILITY>
             | <IH EDGE CLIPPING MODE>
             | <IH FILL REFERENCE POINT>
             | <IH PATTERN SIZE>
             | <IH AUXILIARY COLOUR>
             | <IH TRANSPARENCY>
             | <IH MITRE LIMIT>
             | <IH LINE CAP>
             | <IH LINE JOIN>
             | <IH LINE TYPE CONTINUATION>
             | <IH LINE TYPE INITIAL OFFSET>
             | <IH TEXT SCORE TYPE>
             | <IH RESTRICTED TEXT TYPE>
             | <IH INTERPOLATED INTERIOR>
             | <IH EDGE CAP>
             | <IH EDGE JOIN>
             | <IH EDGE TYPE CONTINUATION>
             | <IH EDGE TYPE INITIAL OFFSET>
             | <IH SYMBOL LIBRARY INDEX>
             | <IH SYMBOL COLOUR>
             | <IH SYMBOL SIZE>
             | <IH SYMBOL ORIENTATION>

<attribute and control
        group enumerated>     ::=   <LINE ATTRIBUTES>
             | <MARKER ATTRIBUTES>
             | <TEXT PRESENTATION AND PLACEMENT ATTRIBUTES>
             | <TEXT PLACEMENT AND ORIENTATION ATTRIBUTES>
             | <FILL ATTRIBUTES>
             | <EDGE ATTRIBUTES>
             | <PATTERN ATTRIBUTES>
             | <OUTPUT CONTROL>
             | <PICK IDENTIFIER>
             | <ALL ATTRIBUTES AND CONTROL>
             | <ALL>
             | <SYMBOL ATTRIBUTES>

<selection setting enumerated>      ::=   <STATE LIST>
             | &lt;SEGMENT&gt;

<asf name enumerated>   ::=   <IH LINE TYPE ASF>
             | <IH LINE WIDTH ASF>
             | <IH LINE COLOUR ASF>
             | <IH MARKER TYPE ASF>
             | <IH MARKER SIZE ASF>
             | <IH MARKER COLOUR ASF>
             | <IH TEXT FONT INDEX ASF>
             | <IH TEXT PRECISION ASF>
             | <IH CHARACTER EXPANSION FACTOR ASF>
             | <IH CHARACTER SPACING ASF>
             | <IH TEXT COLOUR ASF>
             | <IH INTERIOR STYLE ASF>
             | <IH FILL COLOUR ASF>
```

| <IH HATCH INDEX ASF>
| <IH PATTERN INDEX ASF>
| <IH EDGE TYPE ASF>
| <IH EDGE WIDTH ASF>
| <IH EDGE COLOUR ASF>

<asf group enumerated> ::= <LINE ASFS>
| <MARKER ASFS>
| <TEXT ASFS>
| <FILL ASFS>
| <EDGE ASFS>
| <ALL ASFS>

clip inheritance enumerated> ::= <STATE LIST>
| <INTERSECTION>

<highlighting enumerated> ::= <NORMAL>
| <HIGHLIGHTED>

<segment display priority> ::= <non-negative integer>

<segment pick priority> ::= <non-negative integer>

C.4 Terminal symbols

The following are the terminals in this grammar. Their representation is dependent on the encoding scheme used. In Annex A of the subsequent parts of this Standard, these encoding-dependent symbols are further described.

<element name>
<integer>
<real>
<vdc value>
<string>
<string fixed>
<colour index>
<colour component>
<colour direct>
<integer precision value>
<real precision value>
<index precision value>
<colour precision value>
<colour index precision value>
<name precision value>
<default colour precision indicator>
<vdc integer precision value>
<vdc real precision value>
<data record>
<name>
<vc value>
<bitstream>
<2 x 2 matrix of reals>
<2 x 1 matrix of vdcs>
<3 x 3 matrix of reals>

The CGM extended opcodes are encoding dependent. A complete list of them can be found in the productions for <element name> below.

The enumerated types are:

<INTEGER>

<REAL>
<ON>
<OFF>
<INDEXED>
<DIRECT>
<ABSTRACT>
<METRIC>
<ABSOLUTE>
<SCALED>
<94 CHAR>
<96 CHAR>
<MULTI-BYTE 94 CHAR>
<MULTI-BYTE 96 CHAR>
<COMPLETE CODE>
<BASIC 7-BIT>
<BASIC 8-BIT>
<EXTENDED 7-BIT>
<EXTENDED 8-BIT>
<FRACTION OF DISPLAY SURFACE>
<MILLIMETRES WITH SCALE FACTOR>
<PHYSICAL DEVICE COORDINATES>
<NOT FORCED>
<FORCED>
<LEFT>
<RIGHT>
<CENTRE>
<BOTTOM>
<TOP>
<LOCUS>
<SHAPE>
<LOCUS THEN SHAPE>
<INVISIBLE>
<VISIBLE>
<CLOSE INVISIBLE>
<CLOSE VISIBLE>
<PIE>
<CHORD>
<FINAL>
<NOT FINAL>
<INDIVIDUAL>
<BUNDLED>
<HOLLOW>
<SOLID>
<PATTERN>
<HATCH>
<EMPTY>
<GEOMETRIC PATTERN>
<INTERPOLATED>
<STRING>
<CHARACTER>
<STROKE>
<UP>
<DOWN>
<NORMAL HORIZONTAL>
<CONTINUOUS HORIZONTAL>
<NORMAL VERTICAL>
<CAP>
<HALF>
<BASE>
<CONTINUOUS VERTICAL>
<YES>

<NO>
<LINE TYPE ASF>
<LINE WIDTH ASF>
<LINE COLOUR ASF>
<MARKER TYPE ASF>
<MARKER SIZE ASF>
<MARKER COLOUR ASF>
<TEXT FONT ASF>
<TEXT PRECISION ASF>
<CHARACTER EXPANSION FACTOR ASF>
<CHARACTER SPACING ASF>
<TEXT COLOUR ASF>
<INTERIOR STYLE ASF>
<HATCH INDEX ASF>
<PATTERN INDEX ASF>
<FILL COLOUR ASF>
<EDGE TYPE ASF>
<EDGE WIDTH ASF>
<EDGE COLOUR ASF>
<LINE ATTRIBUTES>
<MARKER ATTRIBUTES>
<TEXT PRESENTATION AND PLACEMENT ATTRIBUTES>
<TEXT PLACEMENT AND ORIENTATION ATTRIBUTES>
<FILL ATTRIBUTES>
<EDGE ATTRIBUTES>
<PATTERN ATTRIBUTES>
<OUTPUT CONTROL>
<PICK IDENTIFIER>
<ALL ATTRIBUTES AND CONTROL>
<ALL>
<LINE BUNDLE INDEX>
<LINE TYPE>
<LINE WIDTH>
<LINE COLOUR>
<LINE CLIPPING MODE>
<MARKER BUNDLE INDEX>
<MARKER TYPE>
<MARKER SIZE>
<MARKER COLOUR>
<MARKER CLIPPING MODE>
<TEXT BUNDLE INDEX>
<TEXT FONT INDEX>
<TEXT PRECISION>
<CHARACTER EXPANSION FACTOR>
<CHARACTER SPACING>
<TEXT COLOUR>
<CHARACTER HEIGHT>
<CHARACTER ORIENTATION>
<TEXT PATH>
<TEXT ALIGNMENT>
<FILL BUNDLE INDEX>
<INTERIOR STYLE>
<FILL COLOUR>
<HATCH INDEX>
<PATTERN INDEX>
<EDGE BUNDLE INDEX>
<EDGE TYPE>
<EDGE WIDTH>
<EDGE COLOUR>
<EDGE VISIBILITY>
<EDGE CLIPPING MODE>

<FILL REFERENCE POINT>
<PATTERN SIZE>
<AUXILIARY COLOUR>
<TRANSPARENCY>
<LINE CAP>
<LINE JOIN>
<LINE TYPE CONTINUATION>
<LINE TYPE INITIAL OFFSET>
<TEXT SCORE TYPE>
<RESTRICTED TEXT TYPE>
<INTERPOLATED INTERIOR>
<EDGE CAP>
<EDGE JOIN>
<EDGE TYPE CONTINUATION>
<EDGE TYPE INITIAL OFFSET>
<MITRE LIMIT>
<SYMBOL LIBRARY INDEX>
<SYMBOL COLOUR>
<SYMBOL SIZE>
<SYMBOL ORIENTATION>
<SYMBOL ATTRIBUTES>
<STATE LIST>
<INTERSECTION>
<SEGMENT>
<LINE ASFS>
<MARKER ASFS>
<TEXT ASFS>
<FILL ASFS>
<EDGE ASFS>
<ALL ASFS>
<NORMAL>
<HIGHLIGHTED>
<DRAWING SET>
<DRAWING PLUS CONTROL SET>
<VERSION 2 SET>
<EXTENDED PRIMITIVES SET>
<VERSION 2 GKSM SET>
<VERSION 3 SET>
<0 DEGREES>
<90 DEGREES>
<180 DEGREES>
<270 DEGREES>
<FRACTIONAL>
<MILLIMETRES>
<NON-TANGENTIAL>
<AXIS-TANGENTIAL>
<PARALLEL>
<CROSSHATCH>
<HEIGHT>
<WIDTH>
<BOTH>
<IH LINE BUNDLE INDEX>
<IH LINE TYPE>
<IH LINE WIDTH>
<IH LINE COLOUR>
<IH LINE CLIPPING MODE>
<IH MARKER BUNDLE INDEX>
<IH MARKER TYPE>
<IH MARKER SIZE>
<IH MARKER COLOUR>
<IH MARKER CLIPPING MODE>

<IH TEXT BUNDLE INDEX>
<IH TEXT FONT INDEX>
<IH TEXT PRECISION>
<IH CHARACTER EXPANSION FACTOR>
<IH CHARACTER SPACING>
<IH TEXT COLOUR>
<IH CHARACTER HEIGHT>
<IH CHARACTER ORIENTATION>
<IH TEXT PATH>
<IH TEXT ALIGNMENT>
<IH FILL BUNDLE INDEX>
<IH INTERIOR STYLE>
<IH FILL COLOUR>
<IH HATCH INDEX>
<IH PATTERN INDEX>
<IH EDGE BUNDLE INDEX>
<IH EDGE TYPE>
<IH EDGE WIDTH>
<IH EDGE COLOUR>
<IH EDGE VISIBILITY>
<IH EDGE CLIPPING MODE>
<IH FILL REFERENCE POINT>
<IH PATTERN SIZE>
<IH AUXILIARY COLOUR>
<IH TRANSPARENCY>
<IH MITRE LIMIT>
<IH LINE CAP>
<IH LINE JOIN>
<IH LINE TYPE CONTINUATION>
<IH LINE TYPE INITIAL OFFSET>
<IH TEXT SCORE TYPE>
<IH RESTRICTED TEXT TYPE>
<IH INTERPOLATED INTERIOR>
<IH EDGE CAP>
<IH EDGE JOIN>
<IH EDGE TYPE CONTINUATION>
<IH EDGE TYPE INITIAL OFFSET>
<IH SYMBOL LIBRARY INDEX>
<IH SYMBOL COLOUR>
<IH SYMBOL SIZE>
<IH SYMBOL ORIENTATION>
<IH LINE TYPE ASF>
<IH LINE WIDTH ASF>
<IH LINE COLOUR ASF>
<IH MARKER TYPE ASF>
<IH MARKER SIZE ASF>
<IH MARKER COLOUR ASF>
<IH TEXT FONT INDEX ASF>
<IH TEXT PRECISION ASF>
<IH CHARACTER EXPANSION FACTOR ASF>
<IH CHARACTER SPACING ASF>
<IH TEXT COLOUR ASF>
<IH INTERIOR STYLE ASF>
<IH FILL COLOUR ASF>
<IH HATCH INDEX ASF>
<IH PATTERN INDEX ASF>
<IH EDGE TYPE ASF>
<IH EDGE WIDTH ASF>
<IH EDGE COLOUR ASF>

```
element name>::= <BEGIN METAFILE>
             | <END METAFILE>
             | <BEGIN PICTURE>
             | <BEGIN PICTURE BODY>
             | <END PICTURE>
             | <BEGIN SEGMENT>
             | <END SEGMENT>
             | <BEGIN FIGURE>
             | <END FIGURE>
             | <BEGIN PROTECTION REGION>
             | <END PROTECTION REGION>
             | <BEGIN COMPOUND LINE>
             | <END COMPOUND LINE>
             | <BEGIN COMPOUND TEXT PATH>
             | <END COMPOUND TEXT PATH>
             | <BEGIN TILE ARRAY>
             | <END TILE ARRAY>
             | <METAFILE VERSION>
             | <METAFILE DESCRIPTION>
             | <VDC TYPE>
             | <INTEGER PRECISION>
             | <REAL PRECISION>
             | <INDEX PRECISION>
             | <COLOUR PRECISION>
             | <COLOUR INDEX PRECISION>
             | <NAME PRECISION>
             | <MAXIMUM COLOUR INDEX>
             | <COLOUR VALUE EXTENT>
             | <METAFILE ELEMENT LIST>
             | <METAFILE DEFAULTS REPLACEMENT>
             | <FONT LIST>
             | <CHARACTER SET LIST>
             | <CHARACTER CODING ANNOUNCER>
             | <MAXIMUM VDC EXTENT>
             | &lt;SEGMENT PRIORITY EXTENT&gt;
             | <COLOUR MODEL>
             | <COLOUR CALIBRATION>
             | <FONT PROPERTIES>
             | <GLYPH MAPPING>
             | <SYMBOL LIBRARY LIST>
             | <SCALING MODE>
             | <COLOUR SELECTION MODE>
             | <LINE WIDTH SPECIFICATION MODE>
             | <MARKER SIZE SPECIFICATION MODE>
             | <EDGE WIDTH SPECIFICATION MODE>
             | <VDC EXTENT>
             | <BACKGROUND COLOUR>
             | <DEVICE VIEWPORT>
             | <DEVICE VIEWPORT SPECIFICATION MODE>
             | <DEVICE VIEWPORT MAPPING>
             | <LINE REPRESENTATION>
             | <MARKER REPRESENTATION>
             | <TEXT REPRESENTATION>
             | <FILL REPRESENTATION>
             | <EDGE REPRESENTATION>
             | <INTERIOR STYLE SPECIFICATION MODE>
             | <LINE AND EDGE TYPE DEFINITION>
             | <HATCH STYLE DEFINITION>
             | <GEOMETRIC PATTERN DEFINITION>
             | <VDC INTEGER PRECISION>
             | <VDC REAL PRECISION>
             | <AUXILIARY COLOUR>
```

```
| <TRANSPARENCY>
| <CLIP RECTANGLE>
| <CLIP INDICATOR>
| <LINE CLIPPING MODE>
| <MARKER CLIPPING MODE>
| <EDGE CLIPPING MODE>
| <NEW REGION>
| <SAVE PRIMITIVE CONTEXT>
| <RESTORE PRIMITIVE CONTEXT>
| <PROTECTION REGION INDICATOR>
| <GENERALIZED TEXT PATH MODE>
| <MITRE LIMIT>
| <TRANSPARENT CELL COLOUR>
| <POLYLINE>
| <DISJOINT POLYLINE>
| <POLYMARKER>
| <TEXT>
| <RESTRICTED TEXT>
| <APPEND TEXT>
| <POLYGON>
| <POLYGON SET>
| <CELL ARRAY>
| <GDP>
| <RECTANGLE>
| <CIRCLE>
| <CIRCULAR ARC 3 POINT>
| <CIRCULAR ARC 3 POINT CLOSE>
| <CIRCULAR ARC CENTRE>
| <CIRCULAR ARC CENTRE CLOSE>
| <CIRCULAR ARC CENTRE REVERSED>
| <ELLIPSE>
| <ELLIPTICAL ARC>
| <ELLIPTICAL ARC CLOSE>
| <CONNECTING EDGE>
| <HYPERBOLIC ARC>
| <PARABOLIC ARC>
| <NON-UNIFORM B-SPLINE>
| <NON-UNIFORM RATIONAL B-SPLINE>
| <POLYBEZIER>
| <POLYSYMBOL>
| <BITONAL TILE>
| <TILE>
| <LINE BUNDLE INDEX>
| <LINE TYPE>
| <LINE WIDTH>
| <LINE COLOUR>
| <MARKER BUNDLE INDEX>
| <MARKER TYPE>
| <MARKER SIZE>
| <MARKER COLOUR>
| <TEXT BUNDLE INDEX>
| <TEXT FONT INDEX>
| <TEXT PRECISION>
| <CHARACTER EXPANSION FACTOR>
| <CHARACTER SPACING>
| <TEXT COLOUR>
| <CHARACTER HEIGHT>
| <CHARACTER ORIENTATION>
| <TEXT PATH>
| <TEXT ALIGNMENT>
| <CHARACTER SET INDEX>
| <ALTERNATE CHARACTER SET INDEX>
```

| <FILL BUNDLE INDEX>
| <INTERIOR STYLE>
| <FILL COLOUR>
| <HATCH INDEX>
| <PATTERN INDEX>
| <EDGE BUNDLE INDEX>
| <EDGE TYPE>
| <EDGE WIDTH>
| <EDGE COLOUR>
| <EDGE VISIBILITY>
| <FILL REFERENCE POINT>
| <PATTERN TABLE>
| <PATTERN SIZE>
| <COLOUR TABLE>
| <ASPECT SOURCE FLAGS>
| <PICK IDENTIFIER>
| <COPY SEGMENT>
| <INHERITANCE FILTER>
| <CLIP INHERITANCE>
| <SEGMENT TRANSFORMATION>
| <SEGMENT HIGHLIGHTING>
| <SEGMENT DISPLAY PRIORITY>
| <SEGMENT PICK PRIORITY>
| <ESCAPE>
| <MESSAGE>
| <APPLICATION DATA>
| <LINE CAP>
| <LINE JOIN>
| <LINE TYPE CONTINUATION>
| <LINE TYPE INITIAL OFFSET>
| <TEXT SCORE TYPE>
| <RESTRICTED TEXT TYPE>
| <INTERPOLATED INTERIOR>
| <EDGE CAP>
| <EDGE JOIN>
| <EDGE TYPE CONTINUATION>
| <EDGE TYPE INITIAL OFFSET>
| <SYMBOL LIBRARY INDEX>
| <SYMBOL COLOUR>
| <SYMBOL SIZE>
| <SYMBOL ORIENTATION>

Annex D
(informative)

Guidelines for metafile generators and interpreters

D.1 Introduction

NOTE To unambiguously resolve interoperability and open interchange issues, this International Standard defines rules for conformance of metafiles, metafile generators, and metafile interpreters. See clause 2 and clause 9. The material in this annex predates the addition of the rigorous conformance specifications and should not be considered to be applicable in environments where open interchange and interoperability are important. Nevertheless, some of the contents of this annex may be useful in some closed environments where interoperability and conformance are not issues; and, this annex contains some geometric degeneracy specifications which are normatively referenced from clause 9 and the Model Profile of annex I.

The CGM standardizes the contents, syntax, and semantics of a set of CGM elements. In some situations, it may not be possible for a metafile interpreter to accurately render the contents of a metafile. These situations include cases where the interpreter lacks functionality to render correctly specified elements in the metafile, as well as cases where the contents of the metafile are improperly specified (errors and degeneracies in the metafile data).

In some closed environments, for example generation and interpretation of metafiles by a GKS system, the environment itself may dictate what is to be done in some or all of these situations. In other environments there may not be any rules or guidelines for implementors to follow. For predictability and uniformity of results, it is thus useful to suggest a common approach to situations in which a metafile interpreter cannot accurately render the contents of a metafile, and in which other criteria for deciding a response are lacking.

Clause D.2 deals with errors and degeneracies at a general level. Clauses D.3 and D.4 contain recommended approximations for the interpretation of CGM elements where no one-to-one mapping exists between a CGM element and display device capability. These sub-clauses also deal with some particular mathematical degeneracies and ambiguities.

Dynamic effects are avoided by limiting the position of elements with potentially dynamic effects. Thus, bundle table definitions may appear only in the picture descriptor. In a metafile, changes to the representations of COLOUR TABLE and PATTERN TABLE indexes have no effect on any previous graphical primitive elements that may have used the affected indexes. For Version 1 metafiles, colour table and pattern table definitions may appear only in the picture body; for Version 2 and higher metafiles, these elements may also appear in the Picture Descriptor. In order to improve the portability and predictability of CGM exchange, the use of colour table and pattern table in the picture body is discouraged for metafiles of Version 2 or higher.

D.2 Errors and degeneracies

Three categories of degeneracies are identified in the specification of metafile elements:

a) syntax errors;

b) geometrically degenerate primitives;

c) mathematical singularities and ambiguities.

Subsequent sub-clauses of this annex contain some recommendations for implementors of interpreters for dealing with such exception conditions in metafile contents. Regardless of the strategy chosen by an implementor, it is recommended that it be documented for users of metafile interpreters.

D.2.1 Syntax errors

Errors of syntax include such conditions as the wrong amount of data for an element, or negative values in a parameter whose legal values are non-negative.

The general recommendation to interpreters regarding syntax errors is to recover as much information as possible, and if there is a reasonable interpretation of the element (particularly in the case of primitives) to generate some visible output "do the best you can". In all cases of syntax errors, it is suggested that interpreters employ some mechanism to log or report such conditions when encountered, and that this mechanism be documented for users.

Some syntax errors may make it impossible to meaningfully parse the remainder of a metafile, e.g., a data count does not correspond to the amount of data actually encoded in the element. This class of syntax error is particularly sensitive to the specific encoding.

Other types of syntax error are less severe, in that an interpreter is able to continue parsing the metafile. In some cases assumptions can be made and some meaningful output usually generated. In other cases, e.g., invalid values in attribute setting elements, there may be an infinite number of equally likely possibilities with nothing to recommend between them, and it is suggested that the element be ignored.

EXAMPLE 1 DISJOINT POLYLINE: If the number of points is odd and greater than 1, it is suggested that the interpreter assume that the last point is the odd one, and display the primitive derived by discarding the last point.

EXAMPLE 2 CELL ARRAY: If nx or ny is zero in the cell array specification, it is suggested that no output be generated by the metafile interpreter.

D.2.2 Geometrically degenerate primitives

Geometric degeneracies include elements that are properly specified syntactically and mathematically, but whose defining data yields a geometrically degenerate result.

Zero-length and zero-area specifications comprise the majority of such degeneracies. Typical examples of such specifications are a POLYLINE element in which all data points are coincident (zero length), a POLYGON element where all of the data points alternate between two distinct points (zero area), or a CIRCLE element with zero radius (zero area).

It is recommended in most cases that some visible output be generated for geometrically degenerate primitives. In some cases the application may have reason to render the degeneracy in a particular way. For example, there may be a style that is a natural or "continuous" limiting appearance of the primitive as it approaches degeneracy. In cases in which the application has no strong reasons to pick a particular rendering, it is suggested that the recommendations below be followed.

Primitives may become degenerate after VDC to DC mapping or other transformations. In such cases the transformed locus of the primitive should be drawn, not an interpreter-dependent default. However, when the result is a zero-length line or a zero-area filled primitive, it should be treated as indicated below.

D.2.2.1 Zero length

Zero-length degeneracies apply to the line elements (see 6.6 for a list of line elements). When the specification of a line element degenerates to zero length, it is recommended that the interpreter display a dot in the current line colour and at the size of the current line width. See clauses D.3 and D.4 for interpreter fallback in the case that these attributes cannot be exactly honoured.

If a zero width is specified in the metafile for a line or edge, or a zero size for a marker, or zero height for characters, it is recommended that interpreters select the minimum available width or size for affected primitives.

D.2.2.2 Zero area

Zero-area degeneracies apply to the filled-area elements (see 6.6 for a list of filled-area elements). Two subcategories are recognized — the primitive degenerates to a dot or the primitive degenerates to a line. It is recommended that the dot or line be displayed with the FILL COLOUR if EDGE VISIBILITY is ‘off', unless INTERIOR STYLE is ‘empty'. If EDGE VISIBILITY is ‘on' the dot or line is displayed with the edge attributes.

A CELL ARRAY element whose 3 points define a zero-area parallelogram falls in this category as well. If it is a dot, it is displayed using the colour of the first cell. If it is a line, it is displayed using the colours of the first row or column as appropriate.

D.2.3 Mathematical singularities and ambiguities

Exceptions of this category include elements whose defining data yield a mathematically ill-specified or ambiguous result. Three colinear points in 3-point specified circular arc elements are examples in this category. This annex describes, on an element-by-element basis, such conditions and recommends mathematically sensible reactions where such exist.

D.3 General guidelines

D.3.1 Indexes

With the exception of colour, an out-of-range index in an indexed selection element causes selection of the default index value. The default index value may have been defined by the Metafile Defaults Replacement (MDR) element. If that definition is out-of-range (for example, MDR defines default FONT INDEX to equal 10 when the FONT LIST only has 4 defined entries), then the default value is the value specified in clause 8, not the out-of-range value specified by the MDR. This means in particular that there are no interpreter-dependent tables or list for such elements as FONT LIST, CHARACTER SET LIST, etc.

Out-of-range colour selection indexes are mapped to supported colour indexes in a deterministic fashion; implementors of interpreters should document the mapping.

An out-of-range index on an index definition element (for example, COLOUR TABLE) is ignored.

D.3.2 Colour model

There are two key assumptions to the formation of direct and indexed colour specifications:

a) Many computer graphics devices are inherently indexed in regard to colour attribute selection. These devices include the following:

   1) frame buffer video bit maps (index is pixel value)

   2) pen plotters (index is pen identifier)

   3) stroke devices (index is analogue of voltage for refresh intensity or beam penetration).

b) All of these devices map these indexes to visible attributes through a map. These maps are either loadable under program control (type 1 above), operator control (2), or are fixed in hardware (3) in a way encodable in software at some level.

The most direct hardware control over these devices is, thus, by colour selection by index. Direct hardware control (by index control) presumes that software need have enough knowledge of the accessible hardware range to be able to control the device usefully.

To utilize direct colour specification (if, by assumption a, devices implement indexed specifiers), we need to reverse-map the map noted in assumption b.

A "closest match" element between the directly specified colour and the entries of the table could be based on several algorithms. For example:

a) Minimum spatial distance, as computed in the RGB colour cube.

b) Component-by-component comparison by minimum weight XORing. In fixed tables, colours tend to be evenly distributed throughout the colour space, or at least across a plane through the space. This aids the closest match problem, because there is likely to be a colour somewhat near the target.

In loadable tables of indeterminate length, it may be advisable to load colours from disparate points in the colour space early in the table before the colour selection table is exhausted.

Only the COLOUR TABLE element and the BACKGROUND COLOUR element can change the colour map (if it exists). Direct specification requires that closest match be used to protect static attributes in a display, which, for portability, are the default.

D.3.2.1 Mapping to monochrome

In American colour television systems (NTSC encoding), colour signals are translated to grey scale for monochrome reception by the following equation:

$$Y = 0.30R + 0.59G + 0.11B$$

where R, G, and B are the intensity values of the red, green, and blue components, and Y is the resulting luminance value. This mapping is suggested for metafile interpreters.

The integer mapping of this equation would be, approximately

$$Y' = 3R + 6G + 1B$$

D.3.2.2 Specified versus realized colour precision

It is likely that the numerical ranges of the colour components (e.g., RGB) in a metafile (as specified by the COLOUR VALUE EXTENT element) will differ from the component ranges available on a device. If the metafile was generated with different component ranges than available on the device, the metafile components are mapped into the device components by linearly mapping the metafile component ranges onto the device component ranges.

Figure D.1 gives an example of such a mapping for the RGB colour model. The metafile generator declares a COLOUR PRECISION of 3 bits and a COLOUR VALUE EXTENT of (0,0,0),(7,7,7). The devices has 4 bits of precision in each colour component, and a component range of 0-15. The example shows only the mapping for the red component of colour — green and blue map exactly the same way. The metafile COLOUR VALUE EXTENT effectively states that there are eight levels of red expressable in the metafile — evenly spaced from no red component to full red. The device precision of four bits says there are 16 levels provided by the device. The mapping preserves full intensity and zero intensity of the components. Note that this discussion of direct colour specification also applies to the direct colour values used to set colour table entries.

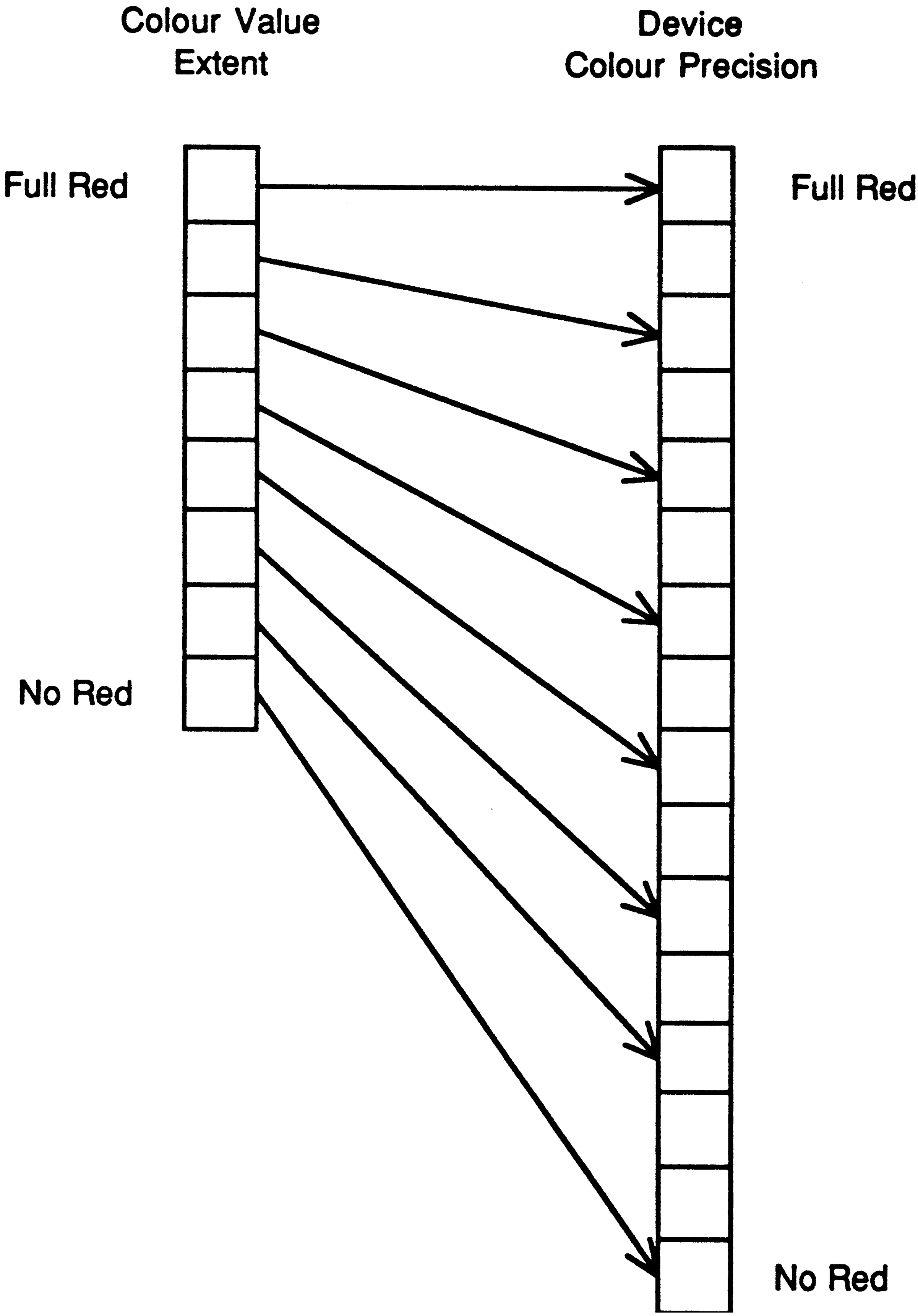


Figure D.1 — Mapping of direct colour specification from metafile to device

D.3.3 Order of metafile descriptor elements

It is recommended that the Metafile Descriptor elements listed below are written first in the descriptor and in the following order:

METAFILE VERSION
METAFILE ELEMENT LIST
METAFILE DESCRIPTION

D.3.4 Unsatisfied references

Elements referring to a non-existing element are ignored.

EXAMPLE 1 COPY SEGMENT: If the segment identifier refers to a non-existing segment.

EXAMPLE 2 RESTORE PRIMITIVE CONTEXT: If an attribute and control set of that context name does not exist.

D.4 Guidelines for element classes

D.4.1 Delimiter elements

It is intended that the BEGIN PICTURE BODY element clear the view surface. This is necessary if pictures are randomly accessed. If the metafile interpreter is composing an image from multiple CGM pictures, the interpreter will clear the view surface only before the first picture in the image.

D.4.1.1 BEGIN PICTURE BODY

The element typically causes the view surface to be cleared to the background colour, as specified by the BACKGROUND COLOUR element if present in the Picture Descriptor or default background colour if not.

D.4.1.2 END PICTURE

It is suggested that interpretation of the END PICTURE element guarantee that the picture reflect the display of all the elements in the picture body.

D.4.2 Metafile descriptor elements

There are no interpreter guidelines for any of the Metafile Descriptor Elements.

D.4.3 Picture descriptor elements

D.4.3.1 DEVICE VIEWPORT

DEVICE VIEWPORT, DEVICE VIEWPORT SPECIFICATION MODE, DEVICE VIEWPORT MAPPING: In the case where the VC specifier in DEVICE VIEWPORT SPECIFICATION MODE is set to either ‘millimetres with scale factor' or ‘physical device coordinates' not all interpreters may be able to interpret the DEVICE VIEWPORT element as specified, and the interpretation becomes implementation dependent.

Since this International Standard does not mandate the behaviour and presentation modes of an interpreter, an application may wish to control the VDC-to-Device mapping by mechanisms external to the CGM picture description, for example, when including CGM pictures in documents.

D.4.3.2 GEOMETRIC PATTERN DEFINITION

If an index has not been defined, or a segment has not been defined an empty segment is used for the geometric pattern. The effect is the same as interior style ‘empty'.

D.4.4 Control elements

D.4.4.1 AUXILIARY COLOUR

This element is intended to address hardware features commonly available in raster display devices. Some devices may have no such capabilities, or may have a subset of the capabilities to which this element pertains. Simulation of such a feature may be very complex and expensive. It is not intended that an interpreter need simulate the feature when it is not available in the hardware or firmware.

D.4.4.2 CLIP RECTANGLE

If any part of the CLIP RECTANGLE is outside VDC EXTENT, clipping should occur at the intersection of CLIP RECTANGLE and VDC EXTENT.

D.4.4.3 CLIPPING MODES

If interpreters cannot implement the ‘locus' clipping mode for LINE CLIPPING MODE, MARKER CLIPPING MODE or EDGE CLIPPING MODE, ‘locus then shape' should be used as a fallback.

D.4.5 Graphical primitive elements

D.4.5.1 APPEND TEXT

It is may be difficult for an implementation to buffer text attribute changes which are permitted between partial text elements. The suggested fallback behaviour for interpreters is to accept the attribute change, accumulate and concatenate the pieces of the string, and align and display the completed text string with the set of attributes in effect at the time the string is completed (an APPEND TEXT with a ‘final' flag).

Changing the TEXT PRECISION in Text Open (partial text) state is likely to lead to unpredictable results. Generators are discouraged from doing this. Interpreters that can otherwise implement text attribute changes in partial text state should ignore this element in that state as a fallback.

D.4.5.2 RESTRICTED TEXT

Some fonts may have kerns extending outside of the character body, either horizontally or vertically. For the RESTRICTED TEXT element, which requires all visible portions of a character symbol to be confined to a parallelogram derived from the element's extent parameters, such kerns are treated in an implementation-dependent way. One way is to treat the extent parallelogram as a clipping boundary as far as kerns are concerned. Another way is for the interpreter to make "worst case" assumptions about the font, and process the text string as if there were an extra half-space or space on either end to allow for kerns this way insures that all portions of the text are drawn and within the parallelogram.

If either the width or height component of the extent parameter is zero, it is recommended that interpreters display a dot, if both are zero, or a line along the non-zero component of the positioned extent, if one is zero. The dot or line should be in the current TEXT COLOUR.

D.4.5.3 CELL ARRAY

It is suggested that a device that cannot display the CELL ARRAY element draw a parallelogram corresponding to the specified area. The parallelogram is drawn according to the filled-area edge attributes, with the exception that EDGE VISIBILITY is ignored.

If the three points defining the CELL ARRAY form a parallelogram and the CELL ARRAY cannot be displayed as a parallelogram, then it is implementation dependent whether the specified parallelogram is drawn or the CELL ARRAY is displayed without skewing.

D.4.5.4 CIRCULAR ARC 3 POINT

A dot is drawn for an ARC with only one distinct point.

If the element has only two distinct points:

— If the intermediate point coincides with the starting point, a line is drawn between the starting point and the ending point.

— If the intermediate point coincides with the ending point, a line is drawn between the starting point and the ending point.

— If the starting point coincides with the ending point, a circle is drawn such that the line from the starting point to the intermediate point is a diameter of the circle.

If the element has three collinear coordinates:

— If the intermediate point lies between the starting point and ending point, on the line through those two points, then a line segment is drawn from the starting point to the ending point.

If the intermediate point does not lie between the starting point and the ending point, but is on the line through those two points, then two different semi-infinite lines are drawn: one begins at the starting point and extends in the direction of the vector from the ending point to the starting point; the other begins at the ending point and extends in the direction of the vector from the starting point to the ending point.

D.4.5.5 CIRCULAR ARC 3 POINT CLOSE

In the case that the starting and ending points coincide and the intermediate point is distinct, the interior of the defined circle (see CIRCULAR ARC 3 POINT) is rendered according to the filled-area interior attributes, and the edge is rendered according to the edge attributes. When in addition the ‘close type' is ‘pie', as well as filling the circle a radius should also be drawn with the current edge attributes.

In all the other cases (see CIRCULAR ARC 3 POINT) the defined arc does not have a unique or finite centre, and an area (e.g., half-plane) cannot be uniquely determined. Therefore in these cases the boundary/edge is generated as specified for each of the singular conditions described above, under the guidelines for CIRCULAR ARC 3 POINT. The suggestions for rendering of the boundary and/or edge of zero-area primitives (see clause D.2) should be applied to the drawing of the specified lines in these cases.

D.4.5.6 CIRCULAR ARC CENTRE

If the start ray and the end ray coincide, it is recommended that the interpreter draw the full circle.

D.4.5.7 CIRCULAR ARC CENTRE CLOSE

If the start ray and the end ray coincide, it is recommended that the interpreter draw and fill the full circle. When in addition the ‘close type' is ‘pie', as well as filling the circle a radius should also be drawn with the current edge attributes.

D.4.5.8 CIRCULAR ARC CENTRE REVERSED

If the start ray and end ray coincide, it is recommended that the interpreter draw the full circle.

D.4.5.9 Elliptical Elements (rendering techniques)

There are a number of techniques for rendering an ellipse specified by the parameterization of ISO/IEC 8632-1. One such technique involves the use of a pair of equations of a single parameter (t) whose coefficients may derived from the conjugate diameters. One equation generates x-coordinates and the other generates y-coordinates as t varies.

NOTE This technique and its use are included in claims covered by patents which are held by Conographic Corporation and which are applicable in many ISO member bodies; by publication of ISO/IEC 8632, no position is taken with respect to the validity of this claim or of any patent rights in connection therewith.

D.4.5.10 Elliptical Elements (supplementary information)

To supplement the syntactic definition of elliptical elements in clauses 6 and 7 of ISO/IEC 8632-1, a mathematical representation of ellipses is given here.

An ellipse is parameterized by its centrepoint and an endpoint from each of the diameters of any CDP. The following equations define the ellipse in terms of these data. For simplicity, the equations are presented for an ellipse centred at the origin. Let *P1 = (x_1, y_1)* and *P2 = (x_2, y_2)* be the two CDP endpoints. The equation for the ellipse is

$$\frac{(x\cos(\alpha) + y\sin(\alpha))^2}{a^2} + \frac{(x\sin(\alpha) + y\cos(\alpha))^2}{b^2} = 1$$

where

$$a = \frac{y_u - y_2}{\sin(\gamma)}$$

$$b = \frac{y_2 - y_u}{\sin(\lambda)}$$

$$\gamma = \arctan\left(\frac{y_c - y_2}{x_2 - x_c}\right)$$

$$\alpha = \arctan\left(-\frac{y_v}{x_v}\right)$$

$$x_c = \frac{x_2 + y_1}{2}$$

$$y_c = \frac{y_2 - x_1}{2}$$

$$y_u = y_c + d\sin(\gamma)$$

$$x_v = x_c + d\cos(\gamma)$$

$$y_v = y_c - d\sin(\gamma)$$

$$d = \frac{x_c}{\cos(\phi)}$$

$$\phi = \arctan\left(\frac{y_c}{x_c}\right)$$

The generalization for an ellipse with an arbitrary centrepoint M = (x_m , y_m) is straightforward.

D.4.5.11 ELLIPTICAL ARC

If the start ray and end ray are coincident, it is recommended that the interpreter draw the full ellipse.

D.4.5.12 ELLIPTICAL ARC CLOSE

If the start ray and end ray are coincident, it is recommended that the interpreter draw and fill the full ellipse. When in addition the 'close type' is 'pie', as well as filling the ellipse a radius should also be drawn with the current edge attributes.

D.4.5.13 Tile Array

If the number of tiles present does not match the count specified by the BEGIN TILE ARRAY parameter, it is recommended that the missing tiles be treated as encoded as "null background".

D.4.6 Attribute elements

D.4.6.1 BUNDLES

Clause 6 describes the component attributes comprising the various attribute bundles. In Version 1 metafiles, there are only "predefined" bundles (the bundle representations may be set in Version 2, Version 3 and Version 4 metafiles). In rendering Version 1 metafiles, it is strongly recommended that interpreters use only these components to achieve distinguishability and not use other, currently nonstandardized attributes.

D.4.6.2 LINE TYPE

If the specified implementation-dependent line type is not available, 'solid' is used.

If line type cannot be maintained continuously across interior vertices of a single polyline element, then restarting the line pattern at each interior vertex is the recommended action.

D.4.6.3 LINE WIDTH

If a device cannot produce a line of the exact specified width, the closest implemented width is chosen. If a zero width is specified in the metafile, it is recommended that interpreters select the minimum available width for affected primitives.

D.4.6.4 MARKER TYPE

If the specified implementation-dependent marker type is not available, the marker type 'asterisk' is used.

D.4.6.5 MARKER SIZE

The marker size is mapped to the nearest available marker size on the device. The effect of MARKER SIZE on implementation-dependent markers is implementation dependent. If a zero size is specified in the metafile, it is recommended that interpreters select the minimum available size for affected primitives.

D.4.6.6 TEXT PRECISION

If a specified text precision is not available, the next more precise implemented text precision is chosen. If no such (more precise) value is available, it is interpreter dependent whether: (1) the next lower value is used, or (2) whether a substitute font will be invoked temporarily to provide the requested text precision. Note that the font may already be a temporary invocation to provide a character set not provided by the current TEXT FONT INDEX (see CHARACTER SET INDEX in this annex).

D.4.6.7 CHARACTER EXPANSION FACTOR

The next available value smaller than or equal to the specified value is selected. If no such value is available, the next larger value is selected. The effective character height and character width are set to values that allow characters to completely fit into an enclosing rectangle determined by the desired character height and width. If this is not possible, the smallest available character size is used.

D.4.6.8 CHARACTER SPACING

The next available value smaller than or equal to the specified value is selected. If no such value is available, the next larger value is selected.

D.4.6.9 CHARACTER HEIGHT

The next available value smaller than or equal to the specified value is selected. If no such value is available, the next larger value is selected. If a zero height is specified in the metafile, it is recommended that interpreters select the minimum available height for affected primitives.

D.4.6.10 CHARACTER ORIENTATION

If the specified character up vector is not available, the nearest available vector is chosen. If two are equally near, the one in a positive angular direction is chosen. When the character up vector and the character base vector are not at right angles, and hardware text is being used which cannot be skewed, the base vector is used to determine character orientation. If the character path is ‘left' or ‘right', the character base vector determines placement of character origins; if the path is ‘up' or ‘down', the character up vector determines placement of character origin.

D.4.6.11 TEXT PATH

The fallback value for ‘left' is ‘right', the fallback value for ‘up' is ‘down', and vice versa. If the fallback value recommended above is not available, ‘right' is chosen.

D.4.6.12 TEXT ALIGNMENT

For the TEXT element, if an alignment value is not available, the closest available value is used. For RESTRICTED TEXT, the interpreter should honour alignment exactly for the positioning of the extent parallelogram. In cases where multiple precisions are specified within a complete string, the interpreter may, if necessary, use a precision lower than the highest overall precision for alignment purposes.

D.4.6.13 CHARACTER SET INDEX

If the selected character set is not available in the selected font at the time of elaboration of TEXT, APPEND TEXT, or RESTRICTED TEXT elements, the font will be temporarily changed to one where the selected character set can be represented.

D.4.6.14 FONT DESIGN AND THE FONT COORDINATE SYSTEM

Sufficient white space should be allowed when setting the limits of the character body relative to the characters in the font to permit characters to be displayed with their bodies flush without producing conflicts or overlaps between ascenders and descenders, and with normal spacing between characters. This permits standardized use of the continuous alignment parameter in TEXT alignment, and supports the intent of spacing with CHARACTER SPACING = 0.0.

D.4.6.15 INTERIOR STYLE

If the requested interior style is not available, ‘hollow' is used.

D.4.6.16 HATCH INDEX

The ideal value of ‘positive slope' is an angle of positive 45°, and of ‘negative slope' is an angle of positive 135° (negative 45°). If the interperter cannot render these lines with these slopes, then lines with slopes that are recognizably similar may be used. Angles of 30°-60° and 120°-150° are reasonable approximations respectively for ‘positive slope' and ‘negative slope'.

D.4.6.17 EDGE TYPE

If the specified implementation-dependent edge type is not available, ‘solid' is used.

If edge type cannot be maintained continuously across interior vertices of a single POLYGON or POLYGON SET element, then restarting the line pattern at each interior vertex is the recommended action.

D.4.6.18 EDGE WIDTH

If a device cannot produce a edge of the exact specified width, the closest implemented width is chosen. If a zero width is specified in the metafile, it is recommended that interpreters select the minimum available width for affected primitives.

It is recommended that the finite-width realized edge be centred on the zero-width ideal edge when the edge is rendered.

D.4.6.19 PATTERN SIZE

If a device cannot produce skewed and/or rotated patterns, then the pattern size is interpreted as if the pattern height vector were vertical and the pattern width vector were horizontal. If a device cannot produce a pattern of the exact specified size, the closest implemented size is chosen.

D.4.6.20 ASPECT SOURCE FLAGS

If the initial ASFs are not altered, the expected behaviour of the interpreter is .

a) as if individual specification of bundled aspects were not a system feature, if the initial values of all the ASFs are ‘bundled'; or

b) as if specification of bundled aspects via a bundle were not a system feature, if the initial values of all the ASFs are ‘individual'.

D.4.6.21 SYMBOL COLOUR

The colour of a symbol which may be inherent in its definition should have precedence over the default of the SYMBOL COLOUR element. A colour which is explicitly set with the SYMBOL COLOUR element should have precedence over colour which is inherent in the symbol definition.

D.4.7 Escape elements

It is recommended that Data Records be coded in a structured manner, using the SDR data type defined in clause 7 and annex C.

D.4.8 External elements

It is recommended that Data Records be coded in a structured manner, using the SDR data type defined in clause 7 and Annex C.

D.4.9 Segment elements

The restriction that segment attributes be set only immediately after the BEGIN SEGMENT element and before any other element avoids any dynamic effects.

D.4.9.1 SEGMENT DISPLAY PRIORITY

If the output device cannot adjust segment display priority on interpretation, segments should be displayed in order of occurrence.

D.4.9.2 COPY SEGMENT with CLIP INHERITANCE ‘intersection'

If the interpreters cannot handle clip rectangles transformed by a copy transformation with non-zero off-diagonal elements (resulting in a parallelogram) the suggested fallback is to clip to an effective clip rectangle which is the smallest axis-aligned rectangle that contains the transformed clip rectangle. Similarly, in the case where multiple parallelograms might be composed (by intersection) to form a general convex polygon, interpreters should intersect the circumscribing rectangles to derive an effective clip rectangle.

Annex E
(informative)

Guidelines for private encodings

NOTE To unambiguously resolve interoperability and open interchange issues, this International Standard defines rules for conformance of metafiles, metafile generators, and metafile interpreters. See clause 2 and clause 9. The material in this annex predates the addition of the rigorous conformance specifications and should not be considered to be applicable in environments where open interchange and interoperability are important. Nevertheless, the contents of this annex may be useful in some closed environments where interoperability and conformance are not issues.

In some closed environments, it may be useful to define and use an encoding other than one of those in this International Standard. For such a encoding to be a candidate for standardization, it should conform to the following conditions.

a) All CGM elements shall have a specified encoding, with the exception of the precision commands, which may not be applicable to a particular encoding. An element that sets an interpretation mode for other elements may be implicit in the commands that it affects, as opposed to being coded as a separate element. (For example, a procedural encoding might include separate calls for LINE_COLOUR_DIRECT(R,G,B) and LINE_COLOUR_INDEX(I) and omit COLOUR SELECTION MODE).

b) All CGM functionality shall be realizable (for example, both integer and real coordinates), except where noted above under (a).

c) The encodings shall utilize sufficient precision to accommodate the parameter ranges defined in the Model Profile.

Furthermore, in keeping with the design guidelines used for developing the standardized encodings, it is suggested that designers of private encodings ensure:

— the ability to translate a metafile encoded in one of the standardized encodings into the private encoding;

— the corresponding ability to translate a privately encoded metafile to one of the standardized bindings.

These requirements should be considered as recommendations for those designing non-standardized encodings. In addition, it is strongly recommended that encodings support the range of coordinate data precisions standardized in ISO/IEC 8632-3.

Annex F
(informative)

Reference models

The CGM is designed to be usable by and useful to a wide range of applications, graphics systems, and devices or workstations. The figures in this annex are context diagrams they illustrate the relationship of the CGM to the other components of a graphics system. They are not meant to be detailed specifications of interfaces and procedures, but rather conceptual illustrations of relationships.

CGM is graphics-system independent. Figure F.1 shows the CGM in the context of a generic device-independent graphics system. The metafile generator exists below and is invoked by the application-callable layer, at the level of the device or workstation driver. The metafile generator records device independent picture descriptions, conceptually in parallel with the presentation of images on actual devices.

By design, the CGM may be interpreted either by a special process that does not involve the high-level facilities of a general purpose graphics system (Figure F.2), or may be interpreted using the application-callable services of a device independent graphics package (Figure F.3). Figure F.2 might illustrate, for example, a scenario where a CGM has been customized to a particular application and device environment. It might also illustrate the case of a target graphics device which has the CGM element set in hardware or firmware. Figure F.3 illustrates an easy and convenient way to render the pictorial content of a metafile in a more device-independent and application-independent manner.

A GKS implementation could be the device-independent graphics system in these context diagrams. The GKS standard, however, specifies the relationship of GKS to metafiles with somewhat more detail than these diagrams contain.

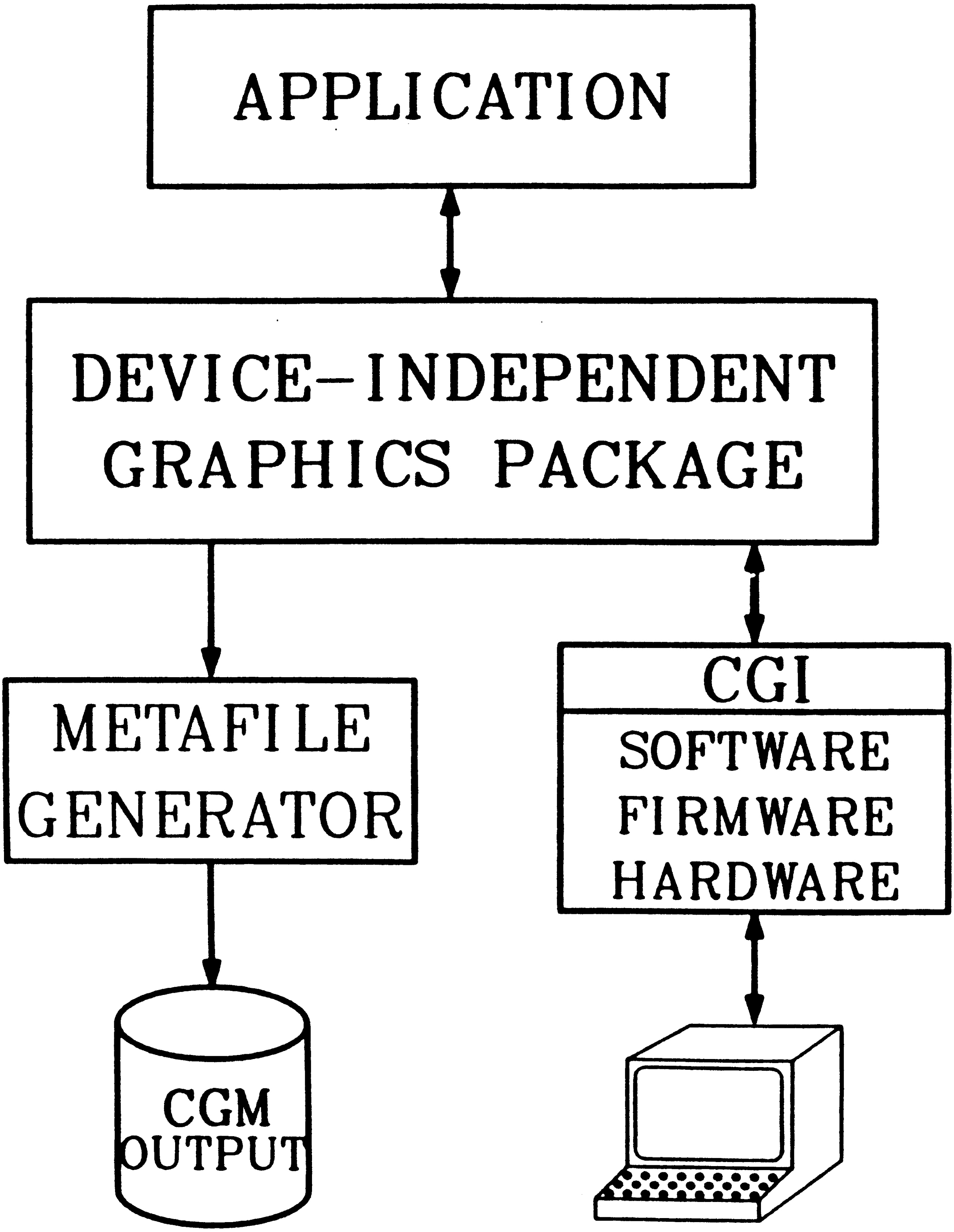


Figure F.1 — Relationship of CGM to traditional graphics package

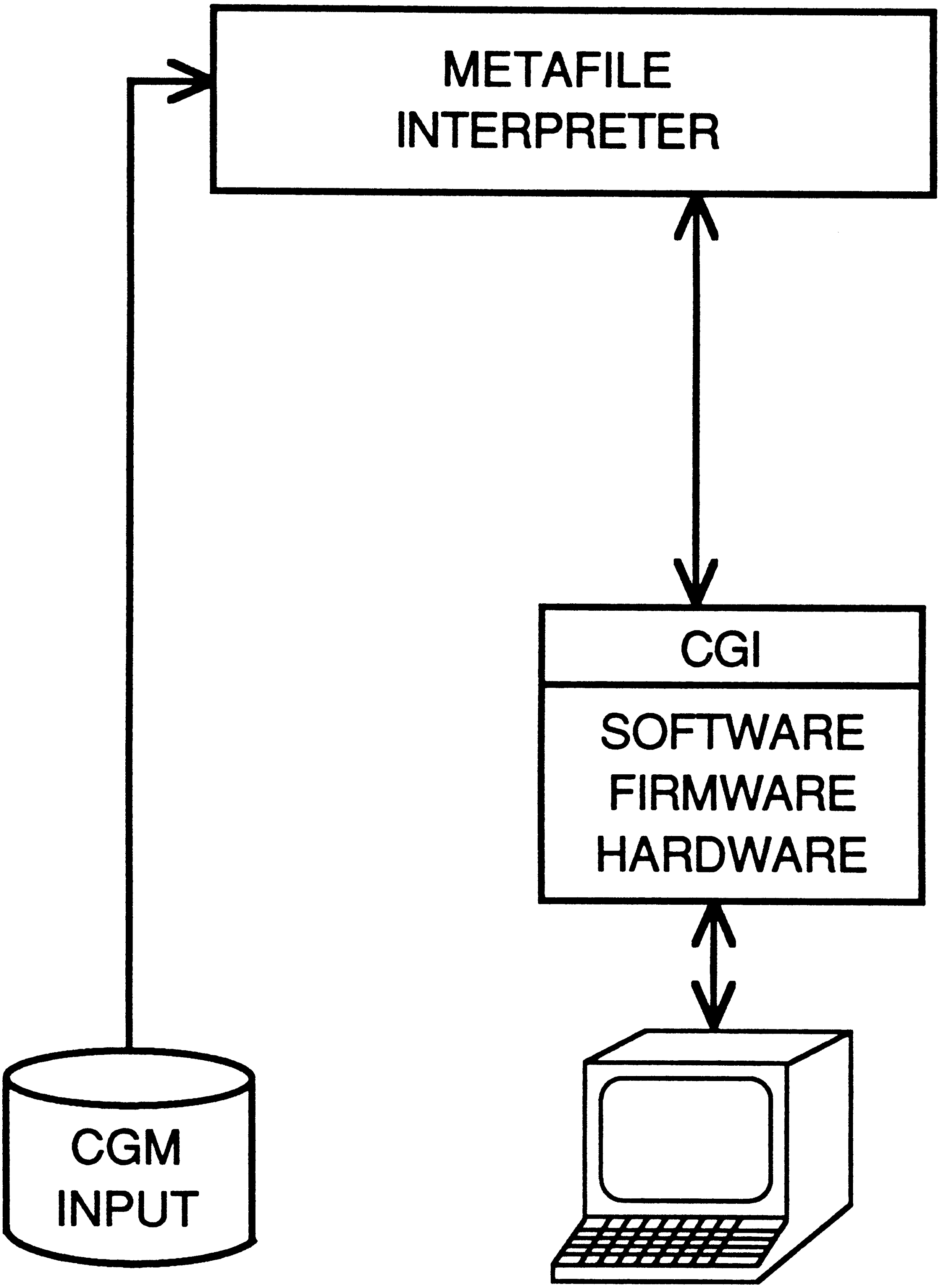


Figure F.2 — Metafile interpretation with no graphics package

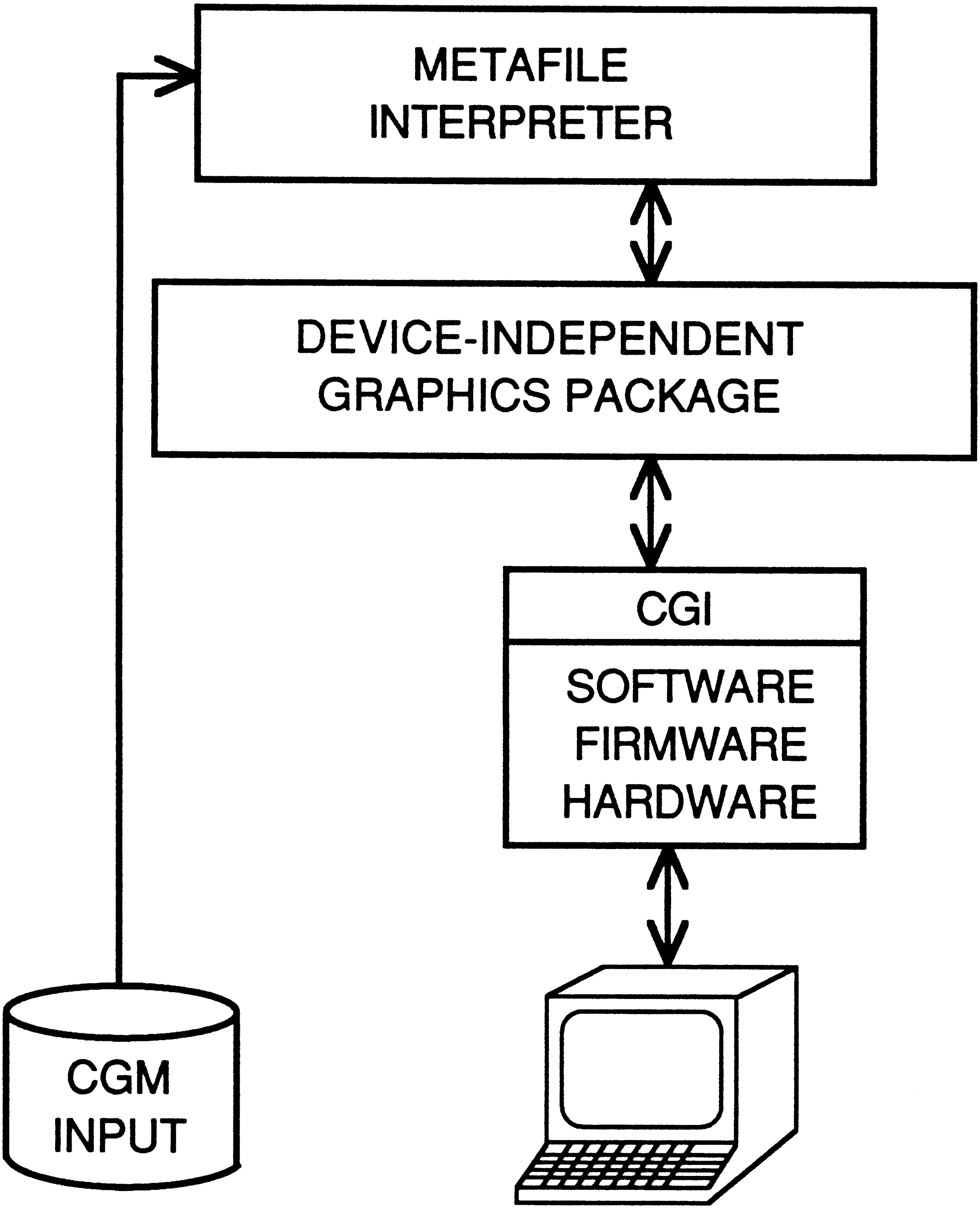


Figure F.3 — Metafile interpretation using traditional graphics package

Annex G
(informative)

Conversions between CIEXYZ reference colour space & metafile colour spaces

G.1 Introduction

This annex is for information only and summarizes the conversion formulae between the CIEXYZ reference colour space and the colour spaces which can be selected with the COLOUR MODEL element. Authoritative references should be consulted for precise specifications to use in implementing colour space conversion.

Examples for use in the COLOUR CALIBRATION element are based either on CIE Standard Illuminant D_{50} or D_{65}.

G.2 CIELUV

G.2.1 Conversion from the CIEXYZ reference colour space to CIELUV

The CIELUV colour space is related to the CIEXYZ reference colour space by the following equations:

$$L^* = 116(Y/Y_n)^{\frac{1}{3}} - 16, \quad \text{for } Y/Y_n > 0.008856$$
$$L^* = 903.3(Y/Y_n), \quad \text{for } Y/Y_n \le 0.008856$$
$$u^* = 13L^*(u' - u'_n)$$
$$v^* = 13L^*(v' - v'_n)$$

with:

$$u' = 4X/(X + 15Y + 3Z)$$
$$v' = 9Y/(X + 15Y + 3Z)$$
$$u'_n = 4X_n/(X_n + 15Y_n + 3Z_n)$$
$$v'_n = 9Y_n/(X_n + 15Y_n + 3Z_n)$$

Where X, Y, Z describe the colour stimulus considered, and X_n, Y_n, Z_n are the tristimulus values of the nominal white stimulus (reference white) which according to CIE recommendation is the perfect white reflecting or transmitting diffuser There may be particular applications where the perfect diffuser is not the best choice for the reference white. One example is reflective papers where the reference white must be defined by a contribution from the substrate as well as the illuminant.

Values of X_n, Y_n, Z_n, u'_n and v'_n for CIE Standard Illuminant D_{50}, the 2° Standard Observer, and the perfect diffuser are given below. The colorimetric parameters for the reference white will depend on the wavelength range and interval of summation. The default values are based on a range of 380 nm to 700 nm in 10 nm intervals. If a different range and/or interval is used, the colorimetric values must be recalculated.

$$X_n = 0.96422$$
$$Y_n = 1.0000$$
$$Z_n = 0.82521$$

$$u'_n = 0.20916$$
$$v'_n = 0.48807$$

G.2.2 Conversion from CIELUV to the CIEXYZ reference colour space

The CIEXYZ reference colour space is related to the CIELUV colour space by the following equations:

$$Y = Y_n \left[\frac{L^* + 16}{116} \right]^3, \quad \text{for } L^* > 8$$

$$Y = Y_n \left[\frac{L^*}{903.3} \right], \quad \text{for } L^* \le 8$$

$$u' = u'_n + \frac{u^*}{13L^*}$$
$$v' = v'_n + \frac{v^*}{13L^*}$$

The x and y chromaticity coordinates can be derived from the u' and v' coordinates by the following equations:

$$x = 9u'/(6u' - 16v' + 12)$$
$$y = 4v'/(6u' - 16v' + 12)$$

The CIEXYZ reference colour space values are then given by

$$X = x(Y/y)$$
$$Z = (1 - x - y)(Y/y)$$

G.3 CIELAB

G.3.1 Conversion from the CIEXYZ reference colour space to CIELAB

The CIELAB colour space is related to the CIEXYZ reference colour space by the following equations:

$$L^* = 116(Y/Y_n)^{\frac{1}{3}} - 16, \quad \text{for } Y/Y_n > 0.008856$$

$$L^* = 903.3(Y/Y_n), \quad \text{for } Y/Y_n \le 0.008856$$

$$a^* = 500(f(X/X_n) - f(Y/Y_n))$$
$$b^* = 200(f(Y/Y_n) - f(Z/Z_n))$$

where

$$f(C) = C^{\frac{1}{3}}, \qquad \text{for } C > 0.008856$$

$$f(C) = 7.787\,C + 16/116, \qquad \text{for } C \le 0.008856$$

with C being either X/X_n, Y/Y_n, Z/Z_n

For the values of X_n, Y_n, Z_n for CIE Standard Illuminant D_{50} see G.2.1.

G.3.2 Conversion from CIELAB to the CIEXYZ reference colour space

The CIEXYZ reference colour space is related to the CIELAB colour space by the following equations:

$$Y = Y_n \left[\frac{L^* + 16}{116}\right]^3, \qquad \text{for } L^* > 8$$

$$Y = Y_n \left[\frac{L^*}{903.3}\right], \qquad \text{for } L^* \le 8$$

$$X = X_n A^3, \qquad \text{for } A > 0.008856^{1/3}$$

$$X = X_n (A - \frac{16}{116}) / 7.787, \qquad \text{for } A \le 0.008856^{1/3}$$

$$Z = Z_n B^3, \qquad \text{for } B > 0.008856^{1/3}$$

$$Z = Z_n (B - \frac{16}{116}) / 7.787, \qquad \text{for } B \le 0.008856^{1/3}$$

where

$$A = f(Y/Y_n) + a^*/500$$
$$B = f(Y/Y_n) - b^*/200$$

$$f(Y/Y_n) = (Y/Y_n)^{1/3}, \qquad \text{for } Y/Y_n > 0.008856$$

$$f(Y/Y_n) = 7.787(Y/Y_n) + \frac{16}{116}, \qquad \text{for } Y/Y_n \le 0.008856$$

G.4 RGB

G.4.1 Conversion from the CIEXYZ reference colour space to RGB

The RGB colour space is related to the CIEXYZ reference colour space by the following equations:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} K_1 & K_2 & K_3 \\ K_4 & K_5 & K_6 \\ K_7 & K_8 & K_9 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix}^{-1} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

where $[X_r, Y_r, Z_r]$, $[X_g, Y_g, Z_g]$, and $[X_b, Y_b, Z_b]$ are tristimulus values of the primaries.

The elements of the matrix K are related to the tristimulus values of the primaries as follows:

$$K_1 = (Y_g Z_b - Y_b Z_g) / K_D$$
$$K_2 = (X_b Z_g - X_g Z_b) / K_D$$
$$K_3 = (X_g Y_b - X_b Y_g) / K_D$$
$$K_4 = (Y_b Z_r - Y_r Z_b) / K_D$$
$$K_5 = (X_r Z_b - X_b Z_r) / K_D$$
$$K_6 = (X_b Y_r - X_r Y_b) / K_D$$
$$K_7 = (Y_r Z_g - Y_g Z_r) / K_D$$
$$K_8 = (X_g Z_r - X_r Z_g) / K_D$$
$$K_9 = (X_r Y_g - X_g Y_r) / K_D$$

$$K_D = Det(K) = X_r (Y_g Z_b - Y_b Z_g) + X_g (Y_b Z_r - Y_r Z_b) + X_b (Y_r Z_g - Y_g Z_r) .$$

Using, for example the tristimulus values of the primaries from SMPTE RP145 and SMPTE RP37, the values of the matrix K are the following.

$$\begin{bmatrix} 3.497 & -1.734 & -0.543 \\ -1.065 & 1.975 & 0.034 \\ 0.055 & -0.197 & 1.051 \end{bmatrix}$$

Analogous to CIELUV, see G.2.1, the matrix element values depend on the wavelength range and interval.

The reference white specified by this transformation through

$$X_n = X_r + X_g + X_b$$
$$Y_n = Y_r + Y_g + Y_b$$
$$Z_n = Z_r + Z_g + Z_b$$

is CIE Standard Illuminant D_{65}, $(X_n = 0.95047,\ Y_n = 1.0000,\ Z_n = 1.0883)$.

NOTE This is not consistent with the illuminant specified for reflections; however it is aligned with current standards and practices for video displays and TV monitors.

In case the initial data are in the form of the CIE chromaticities of the primaries, and the tristimulus values of the reference white X_n, Y_n, and Z_n the following relations hold:

$$K_i = k_i / T_r, \quad i = 1,2,3$$
$$K_i = k_i / T_g, \quad i = 4,5,6$$
$$K_i = k_i / T_b, \quad i = 7,8,9$$

where

$$T_r = k_1 X_n + k_2 Y_n + k_3 Z_n$$
$$T_g = k_4 X_n + k_5 Y_n + k_6 Z_n$$
$$T_b = k_7 X_n + k_8 Y_n + k_9 Z_n$$

and the k_i and k_D are determined by replacing the tristimulus values in the definitions of the k_i and k_D by the corresponding chromaticities.

If conversion to non-linear (gamma-corrected) RGB values is required, the gamma correction may be applied to the resulting linear RGB values. The colour calibration provides look-up tables for this transformation.

G.4.2 Conversion from RGB to the CIEXYZ reference colour space

The CIEXYZ reference colour space is related to the RGB colour space by the following equations:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

The tristimulus values of the primaries computed from SMPTE RP145 are:

$$\begin{bmatrix} 0.394 & 0.365 & 0.192 \\ 0.212 & 0.701 & 0.087 \\ 0.019 & 0.112 & 0.958 \end{bmatrix}$$

The reference white specified by this transformation is CIE Standard Illuminant D_{65} (X_n = 0.95047, Y_n = 1.00000, Z_n = 1.0883).

NOTE This is not consistent with the illuminant specified for reflections; however it is aligned with current standards and practices for video displays and TV monitors.

In case the initial data are in the form of the CIE chromaticities of the primaries, and the tristimulus values of the reference white X_n, Y_n, and Z_n, the following relations hold:

$$X_i = x_i T_i, \ \ i = r, g, b$$
$$Y_i = y_i T_i, \ \ i = r, g, b$$
$$Z_i = z_i T_i, \ \ i = r, g, b$$

where T_i have been defined in G.4.1.

The colour calibration allows storage of non-linear (gamma-corrected) RGB values in a metafile. In this case, conversion to the CIEXYZ reference colour space requires transformation of non-linear RGB values by means of lookup tables to linear RGB values prior to the above matrix application.

G.5 RGB-related

Conversion between RGB-related colour spaces and the CIEXYZ reference colour space in general involves three steps:

1) Convert the RGB-related colour space into the (non-linear) RGB colour space by applying a 3x3 matrix.

2) Apply the lookup tables to transform into linear RGB colour space.

3) Apply the matrices as explained in G.4 to convert to and from CIEXYZ reference colour space.

As an example the conversion between ITU 601 non-linear (gamma-corrected) video signal YC_RC_B and the CIEXYZ reference colour space is shown.

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1.000 & 1.402 & 0.00 \\ 1.000 & -0.714 & -0.344 \\ 1.000 & 0.00 & 1.722 \end{bmatrix} \begin{bmatrix} Y \\ C_R \\ C_B \end{bmatrix}$$

where *Y, C_R, C_B* and *R', G', B'* are non-linear (gamma-corrected) values. Application of lookup tables results in linear RGB values, which are then transformed into CIEXYZ reference colour spaces as described in G.4.2.

NOTE The above matrix is the inverse of the following matrix, which relates gamma-corrected RGB to YC_RC_B:

$$\begin{bmatrix} Y \\ C_R \\ C_B \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.500 & -0.419 & -0.081 \\ -0.169 & -0.331 & 0.500 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

G.6 CMYK

G.6.1 Conversion from CMYK to the CIEXYZ reference colour space

The conversion of CMYK to the CIEXYZ reference colour space is specified by a set of colour values in the reference space measured on a grid in CMYK space.

The interpreter of a CGM is expected to use interpolation for CMYK values not on the grid. The interpolation methods, such as quadralinear or higher order interpolation, are not standardized.

G.6.2 CMYK Calibration data

Because no calibration data are available for a standard ink set, the COLOUR CALIBRATION element is recommended to be present when the colour model indicator is CMYK. As long as no reference colour calibration table is available, the CGM generator should use CMYK colour calibration data based on the specific printing colorant, paper, and illumination conditions.

NOTE Colorimetric definitions are available for standard ink sets, ISO 2846, but are not sufficient in its current form to define the CMYK calibration data.

G.7 Bibliography

The following standards and publications are not normative references of this part of ISO/IEC 8632. They do provide useful information regarding the colour concepts discussed herein.

CIE Publication 17.4, *ISO International Lighting Vocabulary*, 1987.

SMPTE, Recommended Practice RP37, *ISO Colour Temperature for Colour Television Studio Monitors*, 1969.

SMPTE, Recommended Practice RP145, *ISO Colour Monitor Colorimetry*, 1986.

ISO 2846:1975, *ISO Set of printing inks for offset printing - colorimetric characteristics.*

ITU-R BT.601-4, Encoding parameters of digital television for studios, ITU Geneva 1990.

Annex H
(normative)

Formal grammar of the functional specification of version 4 metafiles

H.1 Introduction

This grammar is a formal definition of a standard CGM extended syntax for Version 4 metafiles. The encoding-independent and the encoding-dependent production are separated, and there are subsections showing the syntax of each of the standardized encoding schemes. Details on the encoding of terminal symbols can be found in the parts of this International Standard that deal with the particular encoding schemes.

H.2 Definitions

H.2.1 Notation used

<symbol>	- nonterminal
<SYMBOL>	- terminal
<symbol>*	- 0 or more occurrences
<symbol>+	- 1 or more occurrences
<symbol>o	- optional (0 or 1 occurrences)
<symbol>(n)	- exactly n occurrences, n=2,3,...
<symbol-1>::=<symbol-2>	- symbol-1 has the syntax of symbol-2
<symbol-1>|<symbol-2>	- symbol-1 or alternatively symbol-2
<symbol:meaning>	- symbol with the stated meaning
{comment}	- explanation of a symbol or a production

H.2.2 Structured Data Records:

```
<structured data record>      ::=   <member>*

<member>    ::= <typed sequence>

<typed sequence>  ::=   <data type index>
            <data element count>
            <data element: of type "data type">(element count)

<data element count>    ::=   <integer: number of elements in list>

<data type index> ::=   <1: SDR>
            | <2: CI>
            | <3: CD>
            | <4: N>
            | <5: E>
            | <6: I>
            | <7: reserved>
            | <8: IF8>
            | <9: IF16>
            | <10: IF32>
            | <11: IX>
            | <12: R>
            | <13: S>
            | <14: SF>
```

```
                | <15: VC>
                | <16: VDC>
                | <17: CCO>
                | <18: UI8>
                | <19: UI32>
                | <20: BS>
                | <21: CL>
                | <22: UI16>
```

The integer of the "data count" and the index of the "data type index" are represented respectively at the current Integer Precision and the current Index Precision of the metafile.

NOTE 1 The assigned index values of the "data type index" have been harmonized with the specifications of the Computer Graphics Interface (CGI) standard (ISO/IEC 9636) and with values used in Graphical Registration. The "reserved" values in the sequence of the index values correspond to values in CGI which are not relevant to CGM. The N data type of CGM corresponds to the CSN data type of CGI. The SDR data type of CGM corresponds to the D data type of CGI.

NOTE 2 All of the data types above, with the exception of CL, IF8, IF16, IF32, and UI16 are defined and specified in clause 7. CL denotes "Colour List", and is used to refer to parameters such as *cell colour specifiers* of the CELL ARRAY element. These parameters are eligible in most of the encodings (parts 3, and 4) for compression techniques. UI16 represents Unsigned Integer at a fixed precision equivalent to 16 binary bits. IF8, IF16, and IF32 represent Signed Integers at fixed precisions (not subject to Integer Precision) equivalent respectively to 8, 16, and 32 binary bits.

NOTE 3 The binary and character encodings of the enumerated (E) data type in SDR use integers, (e.g. 0,1,2), whereas the clear text encoding of the enumerated data type traditionally uses words (e.g. ‘on’, ‘off’). This difference may prevent blind intertranslation between encodings. It is recommended that the use of enumerated (E) data type in SDR coding for ESCAPE, APPLICATION DATA and GDP elements be avoided. Type index (I) should be used instead. Although a recommended practice, there are some existing registered items which use the word-encoded method for E within SDR.

H.3 Detailed grammar

H.3.1 Metafile structure

```
<metafile>  ::= <BEGIN METAFILE>
                <metafile identifier>
              <metafile descriptor>
              <metafile contents>*
              <END METAFILE>

<metafile identifier>   ::=   <string fixed>

<metafile contents>     ::=   <extra element>*
              <picture>
              <extra element>*

<extra element> ::=     <external element>
             | <escape element>

<picture>   ::= <BEGIN PICTURE>
                <picture identifier>
              <picture descriptor element>*
              <BEGIN PICTURE BODY>
              <picture content>*
              <END PICTURE>

<picture identifier>    ::=   <string fixed>

<picture content> ::=   <picture element>
             | &lt;segment&gt;

             | <application structure>
```

<application structure> ::= <BEGIN APPLICATION STRUCTURE >
<application structure identifier>
<application structure type>
<inheritance flag enumerated>
<application structure descriptor element>*
<BEGIN APPLICATION STRUCTURE BODY>
<picture element>*
<END APPLICATION STRUCTURE >

<application structure identifier> ::= <string fixed>

<application structure type> ::= <string fixed>

<inheritance flag enumerated> ::= <STATE LIST>
|<APPLICATION STRUCTURE>

<picture element> ::= <control element>
| <graphical element>
| <closed figure>
| <primitive attribute element>
| <pattern table element>
| <colour table element>
| <specification element>
| <segment control element>
| <compound line>
| <compound text path>
| <protection region>
| <tile array>
| <extra element>

<segment> ::= <BEGIN SEGMENT>
<segment identifier>
<segment attribute element>*
<eligible picture element>*
<compound line>*
<compound text path>*
<protection region>*
<END SEGMENT>

<segment identifier> ::= <name>

<eligible picture element> ::= <control element>
| <graphical element>
| <closed figure>
| <primitive attribute element>
| <specification element>
| <segment control element>
| <extra element>

<closed figure> ::= <BEGIN FIGURE>
<eligible element within closed figure>*
<END FIGURE>

<eligible element within closed figure>
::= <vdc precision>
| <AUXILIARY COLOUR>
<colour>
| <TRANSPARENCY>
<off-on indicator enumerated>
| <NEW REGION>

```
            | <polypoint element>
            | <gdp element>
            | <rectangle element>
            | <circular element>
            | <elliptical element>
            | <pointless element>
            | <curve element>
            | <edge attribute element>
            | <edge asf>
            | <extra element>

<pointless element>     ::=   <CONNECTING EDGE>

<edge asf>  ::= <ASPECT SOURCE FLAGS>
                <edge asf pair>+

<edge asf pair> ::=     <edge asf type enumerated>
                <asf enumerated>

<edge asf type enumerated>    ::=   <EDGE TYPE ASF>
            | <EDGE WIDTH ASF>
            | <EDGE COLOUR ASF>

<compound line> ::=     <BEGIN COMPOUND LINE>
              <eligible element within compound path>*
              <END COMPOUND LINE>

<eligible element within compound path>
           ::= <vdc precision>
            | <polyline element>
            | <gdp element>
            | <circular arc element>
            | <elliptical arc element>
            | <curve element>
            | <extra element>

<compound text path>    ::=   <BEGIN COMPOUND TEXT PATH>
              <eligible element within compound path>*
              <END COMPOUND TEXT PATH>

<protection region>     ::=   <BEGIN PROTECTION REGION>
                <positive index>
              <eligible element within protection region>*
              <END PROTECTION REGION>

<eligible element within protection region>
           ::= <vdc precision>
            | <NEW REGION>
            | <polypoint element>
            | <gdp element>
            | <rectangle element>
            | <circular element>
            | <elliptical element>
            | <curve element>
            | <extra element>

<tile array> ::= <BEGIN TILE ARRAY>
                <position>
                <cell path direction enumerated>
                <line progression direction enumerated>
                <number of tiles in path direction>
                <number of tiles in line direction>
```

```
                <number of cells/tile in path direction>
                <number of cells/tile in line direction>
                <cell size in path direction>
                <cell size in line direction>
                <image offset in path direction>
                <image offset in line direction>
                <image number of cells in path direction>
                <image number of cells in line direction>
              <eligible element within tile array>*
              <END TILE ARRAY>

<position>  ::= <point>

<cell path direction enumerated>    ::=   <0 DEGREES>
             | <90 DEGREES>
             | <180 DEGREES>
             | <270 DEGREES>

<line progression direction enumerated>   ::=   <90 DEGREES>
             | <270 DEGREES>

<number of tiles in path direction> ::=   <positive integer>

<number of tiles in line direction> ::=   <positive integer>

<number of cells/tile in path direction>  ::=   <positive integer>

<number of cells/tile in line direction>  ::=   <positive integer>

<cell size in path direction> ::=   <positive real>

<cell size in line direction> ::=   <positive real>

<image offset in path direction>    ::=   <non-negative integer>

<image offset in line direction>    ::=   <non-negative integer>

<image number of cells in path direction>
            ::= <positive integer>

<image number of cells in line direction>
            ::= <positive integer>

<eligible element within tile array>
            ::= <tile element>
             | <extra element>

<tile element>::= <BITONAL TILE>
                <compression type>
                <row padding indicator>
                <cell background colour>
                <cell foreground colour>
                <method-specific parameters>
                <compressed colour specifiers>
             | <TILE>
                <compression type>
                <row padding indicator>
                <cell colour precision>
                <method-specific parameters>
                <compressed colour specifiers>
```

```
              {these elements each have an }
              {encoding dependent parameter}

<compression type>           ::=   <non-negative index>

<row padding indicator>      ::=   <non-negative integer>

<cell background colour>     ::=   <colour>

<cell foreground colour>     ::=   <colour>

<method-specific parameters> ::=   <structured data record>

<cell colour precision> ::=   <colour precision value>
             | <colour index precision value>
             | <default colour precision indicator>

<compressed colour specifiers>     ::=   <bitstream>
```

H.3.2 Metafile descriptor elements

```
<metafile descriptor>   ::=   <<optional descriptor element>*
              <version>
              <optional descriptor element>*
              <element list>
              <optional descriptor element>*>
             | <<optional descriptor element>*
              <element list>
              <optional descriptor element>*
              <version>
              <optional descriptor element>*>

<version>   ::= <METAFILE VERSION>
                <integer>

<element list>::= <METAFILE ELEMENT LIST>
                  <MEL item>*

<MEL item>   ::=  <element name>*
                |      <element name shorthand enumerated>*

<element name shorthand
        enumerated>     ::=   <DRAWING SET>
             | <DRAWING PLUS CONTROL SET>
             | <VERSION 2 SET>
             | <EXTENDED PRIMITIVES SET>
             | <VERSION 2 GKSM SET>
             | <VERSION 3 SET>

<optional descriptor element> ::=   <description>
             | <VDC TYPE>
                <vdc type enumerated>
             | <MAXIMUM COLOUR INDEX>
                <colour index>
             | <COLOUR VALUE EXTENT>
                <colour value mapping specifier>
             | <METAFILE DEFAULTS REPLACEMENT>
                <element default>+
             | <FONT LIST>
                <font name>+
             | <CHARACTER SET LIST>
                <character set definition>+
```

```
            | <CHARACTER CODING ANNOUNCER>
                <coding technique enumerated>
            | <scalar precision>
            | <MAXIMUM VDC EXTENT>
                <point> (2)
            | &lt;SEGMENT PRIORITY EXTENT&gt;
                <minimum extent>
                <maximum extent>
            | &lt;segment&gt;
            | <COLOUR MODEL>
                <positive index>
            | <COLOUR CALIBRATION>
                <calibration selection>
                <calibration data>
            | <FONT PROPERTIES>
                <font property 3-tuple>+
            | <GLYPH MAPPING>
                <character set index>
                <character set type enumerated>
                <designation sequence>
                <octets per code>
                <glyph source identifier>
                <glyph code association>
            | <SYMBOL LIBRARY LIST>
                <symbol library name>+
            | <extra element>

            | <picture directory>

<picture directory>     ::=   <PICTURE DIRECTORY>
              <location data type selector enumerated>
              <<picture identifier>
              <picture location>
              <application structure directory location>>+

<location data type selector enumerated>  ::=   <UI8>|<UI16>|<UI32>

<picture location>                        ::=   <non-negative integer>

<application structure directory location>      ::=   <non-negative integer>

<description> ::= <METAFILE DESCRIPTION>
                <string fixed>

<vdc type enumerated>   ::=   <INTEGER>
            | <REAL>

<element default> ::=   <control element>
            | <picture descriptor element>
            | <primitive attribute element>
            | &lt;segment attribute element&gt;
            | &lt;segment control element&gt;
            | <extra element>

<font name> ::= <string fixed>

<character set definition>    ::=   <character set enumerated>
              <designation sequence>

<index>     ::= <standard index value>
            | <private index value>
```

```
<character set index>   ::=   <standard index value>

<standard index value>  ::=   <positive integer>
<non-negative integer>  ::=   <integer>     {greater than or equal to 0}
<positive integer>      ::=   <integer>     {greater than 0}
<private index value>   ::=   <negative integer>
<negative integer>      ::=   <integer>     {less than 0}
<positive index>  ::=   <positive integer>
<positive real> ::=     <real>     {greater than 0}

<character set enumerated>    ::=   <94 CHAR>
             | <96 CHAR>
             | <MULTI-BYTE 94 CHAR>
             | <MULTI-BYTE 96 CHAR>
             | <COMPLETE CODE>

<coding technique enumerated> ::=   <BASIC 7-BIT>
             | <BASIC 8-BIT>
             | <EXTENDED 7-BIT>
             | <EXTENDED 8-BIT>


designation sequence>   ::=   <string fixed>

<scalar precision>      ::=   <INTEGER PRECISION>
                <integer precision value>
             | <REAL PRECISION>
                <real precision value>
             | <INDEX PRECISION>
                <index precision value>
             | <COLOUR PRECISION>
                <colour precision value>
             | <COLOUR INDEX PRECISION>
                <colour index precision value>
             | <NAME PRECISION>
                <name precision value>

              {these elements have encoding}
              {dependent parameters}

<point>     ::= <vdc value> (2)

<minimum extent>  ::=   <non-negative integer>

<maximum extent>  ::=   <non-negative integer>

<colour value mapping specifier>    ::=   <colour direct>(2)
             | (<colour scale><colour offset>)(3)

<colour scale>::= <real>

<colour offset> ::=     <real>

<font property 3-tuple> ::=   <property indicator>
              <priority>
              <property value record>

<property indicator>    ::=   <standard index value>

<priority>  ::= <non-negative integer>

<property value record> ::=   <structured data record>
```

```
<octets per code> ::=   <positive integer>

<glyph source identifier>     ::=   <standard index value>

<glyph-code association>      ::=   <structured data record>

<calibration selection> ::=   <positive index>

<calibration data>      ::=   <reference white>
              <rgb and rgb-related calibration data>
              <cmyk calibration data>

<reference white> ::=   <real>(3)

<rgb and rgb-related calibration data>    ::=   <3x3 matrix of reals>(2)
              <non-negative integer> {=n}
              <lookup table entry for red>(n)
              <lookup table entry for green>(n)
              <lookup table entry for blue>(n)

<cmyk calibration data> ::=   <positive integer> {=m}
              <cmyk grid location>(m)
              <ciexyz grid location>(m)

<lookup table entry for red>  ::=   <cco value>(2)

<lookup table entry for green>      ::=   <cco value>(2)

<lookup table entry for blue> ::=   <cco value>(2)

<cmyk grid location>    ::=   <colour direct>

<ciexyz grid location>  ::=   <real>(3)

symbol library name>    ::=   <string fixed>
```

NOTE The following order is recommended for the Metafile Descriptor elements: METAFILE VERSION, METAFILE ELEMENT LIST, (possible multiple occurrences of) METAFILE DESCRIPTION.

H.3.3 Picture descriptor elements

```
<picture descriptor element>  ::=   <SCALING MODE>
                <scaling specification mode enumerated>
                <metric scale factor>
             | <VDC EXTENT>
                <point> (2)
             | <DEVICE VIEWPORT>
                <viewport point>(2)
             | <DEVICE VIEWPORT SPECIFICATION MODE
                <VC specifier enumerated>
                <metric scale factor>
             | <DEVICE VIEWPORT MAPPING>
                <isotropy flag  enumerated>
                <horizontal alignment flag  enumerated>
                <vertical alignment flag  enumerated>
             | <BACKGROUND COLOUR>
                <colour direct>
             | <specification element>
             | <representation element>
             | <pattern table element>
             | <colour table element>
             | <extra element>
```

```
            | <definition element>
            | &lt;segment&gt;
            | <application structure directory>

<application structure directory>   ::=   <APPLICATION STRUCTURE DIRECTORY>
                        <location data type selector enumerated>
                        <<application structure identifier>
                        <application structure location>>+

<application structure location>    ::=   <non-negative integer>

<colour>    ::= <colour index>
            | <colour direct>

<specification element> ::=   <COLOUR SELECTION MODE>
                <colour selection mode enumerated>
            | <LINE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>
            | <MARKER SIZE SPECIFICATION MODE>
                <specification mode enumerated>
            | <EDGE WIDTH SPECIFICATION MODE>
                <specification mode enumerated>
            | <INTERIOR STYLE SPECIFICATION MODE>
                <specification mode enumerated>

<definition element>    ::=   <LINE AND EDGE TYPE DEFINITION>
                <negative index>
                <positive size value>
                <non-negative integer>+
            | <HATCH STYLE DEFINITION>
                <negative index>
                <style indicator enumerated>
                <valid size vector>(2)
                <positive size value>
                <number of hatch lines>
                <gap widths>
                <line types>
            | <GEOMETRIC PATTERN DEFINITION>
                <positive index>
                &lt;segment identifier&gt;
                <point pair>

<style indicator enumerated>  ::=   <PARALLEL>
            | <CROSSHATCH>

<number of hatch lines> ::=   <positive integer> {=n}

<gap widths> ::= <positive integer>(n)

<line types> ::= <index>(n)

<size value> ::= <non-negative vdc value>
            | <non-negative real>

<valid size vector>     ::=   <valid real vector>
            | <valid vdc vector>

<valid real vector>     ::=   <<non-zero real value>
                <real value>>
            | <<real value>
                <non-zero real value>>
```

```
<valid vdc vector>      ::=   <<non-zero vdc value>
                <vdc value>>
             | <<vdc value>
                <non-zero vdc value>>

<non-zero vdc value>    ::=   <vdc value> {greater than or less than 0}

<non-zero real value>   ::=   <real value> {greater than or less than 0}

<non-zero size value>   ::=   <non-zero vdc value>
             | <non-zero real value>

<positive size value>   ::=   <positive vdc value>
             | <positive real value>

<positive vdc value>    ::=   <vdc value>  {greater than 0}

<non-negative vdc value>      ::=   <vdc value>  {greater than or equal to 0}

<non-negative real>     ::=   <real>  {greater than or equal to 0}

<colour selection mode
      enumerated> ::=   <INDEXED>
             | <DIRECT>

<scaling specification mode
   enumerated>::= <ABSTRACT>
             | <METRIC>

<metric scale factor>   ::=   <real>

<isotropy flag enumerated>    ::=   <NOT FORCED>
             | <FORCED>

<horizontal alignment flag
    enumerated> ::=     <LEFT>
             | <CENTRE>
             | <RIGHT>

<vertical alignment flag
    enumerated> ::=     <BOTTOM>
             | <CENTRE>
             | <TOP>

<specification mode enumerated>     ::=   <ABSOLUTE>
             | <SCALED>
             | <FRACTIONAL>
             | <MILLIMETRES>

<viewport point>  ::=   <vc value> (2)

<VC specifier enumerated>     ::=   <FRACTION OF DISPLAY SURFACE>
             | <MILLIMETRES WITH SCALE FACTOR>
             | <PHYSICAL DEVICE COORDINATES>

<representation element>      ::=   <LINE REPRESENTATION>
                <positive index>
                <index>  {line type}
                <size value>  {line width}
                <colour>
             | <MARKER REPRESENTATION>
                <positive index>
```

```
                  <index>  {marker type}
                  <size value>
                  <colour>
             | <TEXT REPRESENTATION>
                  <positive index>
                  <positive index>  {font}
                  <text precision enumerated>
                  <real>   {character spacing}
                  <non-negative real>   {expansion factor}
                  <colour>
             | <FILL REPRESENTATION>
                  <positive index>
                  <interior style enumerated>
                  <colour>
                  <index> {hatch index}
                  <positive index>{pattern index}
             | <EDGE REPRESENTATION>
                  <positive index>
                  <index>  {edge type}
                  <size value>  {edge width}
                  <colour>

<text precision enumerated>   ::=   <STRING>
             | <CHARACTER>
             | <STROKE>

<interior style enumerated>   ::=   <HOLLOW>
             | <SOLID>
             | <PATTERN>
             | <HATCH>
             | <EMPTY>
             | <GEOMETRIC PATTERN>
             | <INTERPOLATED>
```

H.3.4 Control elements

```
<control element> ::=   <vdc precision>
             | <AUXILIARY COLOUR>
                  <colour>
             | <TRANSPARENCY>
                  <off-on indicator enumerated>
             | <CLIP RECTANGLE>
                  <point>(2)
             | <CLIP INDICATOR>
                  <off-on indicator enumerated>
             | <LINE CLIPPING MODE>
                  <clip mode enumerated>
             | <MARKER CLIPPING MODE>
                  <clip mode enumerated>
             | <EDGE CLIPPING MODE>
                  <clip mode enumerated>
             | <NEW REGION>
             | <SAVE PRIMITIVE CONTEXT>
                  <context name>
             | <RESTORE PRIMITIVE CONTEXT>
                  <context name>
             | <PROTECTION REGION INDICATOR>
                  <region index>
                  <region indicator>
             | <GENERALIZED TEXT PATH MODE>
                  <text path mode enumerated>
             | <MITRE LIMIT>
```

```
                <non-negative-real>
             | <TRANSPARENT CELL COLOUR>
                <transparency indicator enumerated>
                <colour>

<off-on indicator enumerated> ::=   <OFF>
             | <ON>

<vdc precision> ::=     <VDC INTEGER PRECISION>
                <vdc integer precision value>
             | <VDC REAL PRECISION>
                <vdc real precision value>
              {these elements have encoding}
              {dependent parameters}

<clip mode enumerated>  ::=   <LOCUS>
             | <SHAPE>
             | <LOCUS THEN SHAPE>

<context name>::= <name>

<region index>::= <positive index>

<region indicator>      ::=   <positive index>

<text path mode enumerated>   ::=   <OFF>
             | <NON-TANGENTIAL>
             | <AXIS-TANGENTIAL>

<transparency indicator enumerated> ::=   <off-on indicator enumerated>
```

H.3.5 Graphical elements

```
<graphical element>     ::=   <polypoint element>
             | <polymarker element>
             | <text element>
             | <cell element>
             | <gdp element>
             | <rectangle element>
             | <circular element>
             | <elliptical element>
             | <curve element>
             | <symbol element>

<polymarker element>    ::=   <POLYMARKER>
                <point>
                <point list>

<polypoint element>     ::=   <polyline element>
             | <POLYGON>
                <point>(3)
                <point list>
             | <POLYGON SET>
                <point edge pair>(3)
                <point edge pair list>

<polyline element>      ::=   <POLYLINE>
                <point pair>
                <point list>
             | <DISJOINT POLYLINE>
                <point pair>
                <point pair list>
```

```
<point list> ::=<point>*

<point pair list> ::=   <point pair>*

<point pair> ::=<point>(2)

<point edge pair> ::=   <point><edge out flag>

<point edge pair list>  ::=   <point edge pair>*

<edge out flag> ::=     <INVISIBLE>
             | <VISIBLE>
             | <CLOSE INVISIBLE>
             | <CLOSE VISIBLE>

<text element>::= <TEXT>
                <point>
                <text tail>
             | <restricted text element>

<restricted text element>     ::=   <RESTRICTED TEXT>
                <extent>
                <point>
                <text tail>

<extent>    ::= <non-negative vdc value>(2)

<text tail> ::= <final character list>
             | <nonfinal character list>

<final character list>  ::=   <FINAL>
                <string>

<nonfinal character list>     ::=   <NOT FINAL>
                <string>
                <partial text attribute element>*
                <spanned text>

<spanned text>::= <APPEND TEXT>
                <text tail>

<cell element>::= <CELL ARRAY>
                <point>(3)
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)

              {this element has an encoding}
              {dependent parameter}

<local colour precision>      ::=   <colour precision value>
             | <colour index precision value>
             | <default colour precision indicator>

<gdp element> ::= <GDP>
                <gdp identifier>
                <point list>
                <data record>

<gdp identifier>  ::=   <integer>
```

```
<rectangle element>     ::=   <RECTANGLE>
                <point pair>

<circular element>      ::=   <CIRCLE>
                <point>
                <radius>
             | <CIRCULAR ARC 3 POINT CLOSE>
                <point>(3)
                <close type>
             | <CIRCULAR ARC CENTRE CLOSE>
                <point>
                <valid vdc vector>(2)
                <radius>
                <close type>
             | <circular arc element>

<radius>    ::= <non-negative vdc value>

<close type> ::= <PIE>
             | <CHORD>

<circular arc element>  ::=   <CIRCULAR ARC 3 POINT>
                <point>(3)
             | <CIRCULAR ARC CENTRE>
                <point>
                <valid vdc vector>(2)
                <radius>
             | <CIRCULAR ARC CENTRE REVERSED>
                <point>
                <valid vdc vector>(2)
                <radius>

<elliptical element>    ::=   <ELLIPSE>
                <point>(3)
             | <ELLIPTICAL ARC CLOSE>
                <point>(3)
                <valid vdc vector>(2)
                <close type>
                <elliptical arc element>

<elliptical arc element>      ::=   <ELLIPTICAL ARC>
                <point>(3)
                <valid vdc vector>(2)

<curve element> ::=     <HYPERBOLIC ARC>
                <point>(3)
                <valid vdc vector>(2)
             | <PARABOLIC ARC>
                <point>(3)
             | <NON-UNIFORM B-SPLINE>
                <spline order>
                <number of control points>
                <control points>
                <list of knots>
                <parameter start value>
                <parameter end value>
             | <NON-UNIFORM RATIONAL B-SPLINE>
                <spline order>
                <number of control points>
                <control points>
                <list of knots>
                <parameter start value>
```

```
                <parameter end value>
                <weights>
             | <POLYBEZIER>
                <positive index>
                <control point>(4)
                <control point list>

<spline order>::= <positive integer>  {m}

<number of control points>    ::=   <positive integer>  {n≥m}

<control points>  ::=   <point>(n)

<list of knots> ::=     <real>(n+m)

<parameter start value> ::=   <real>

<parameter end value>   ::=   <real>

<weights>   ::= <real>(n)

<control point> ::=     <point>

<control point list>    ::=   <<control point>(4)>*
             | <<control point>(3)>*

<symbol element>  ::=         <POLYSYMBOL>
                <positive index>
                <point>
                <point list>
```

H.3.6 Attribute elements

```
<primitive attribute element> ::=   <line attribute element>
             | <marker attribute element>
             | <text attribute element>
             | <filled-area attribute element>
             | <aspect source flags>
             | <pick identifier>
             | <symbol attribute element>

<line attribute element>      ::=   <LINE BUNDLE INDEX>
                <positive index>
             | <LINE TYPE>
                <index>
             | <LINE WIDTH>
                <size value>
             | <LINE COLOUR>
                <colour>
             | <LINE CAP>
                <positive index>
                <positive index>
             | <LINE JOIN>
                <positive index>
             | <LINE TYPE CONTINUATION>
                <positive index>
             | <LINE TYPE INITIAL OFFSET>
                <non-negative real>

<marker attribute element>    ::=   <MARKER BUNDLE INDEX>
                <positive index>
             | <MARKER TYPE>
```

```
                <index>
             | <MARKER SIZE>
                <size value>
             | <MARKER COLOUR>
                <colour>

<partial text attribute element>    ::=   <TEXT BUNDLE INDEX>
                <positive index>
             | <TEXT FONT INDEX>
                <positive index>
             | <TEXT PRECISION>
                <text precision enumerated>
             | <CHARACTER EXPANSION FACTOR>
                <non-negative real>
             | <CHARACTER SPACING>
                <real>
             | <TEXT COLOUR>
                <colour>
             | <CHARACTER HEIGHT>
                <non-negative vdc value>
             | <CHARACTER SET INDEX>
                <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                <positive index>
             | <TEXT SCORE TYPE>
                <type-indicator pair>+
             | <AUXILIARY COLOUR>
                <colour>
             | <TRANSPARENCY>
                <off-on indicator enumerated>
             | <escape element>

<text attribute element>      ::=   <TEXT BUNDLE INDEX>
                <positive index>
             | <TEXT FONT INDEX>
                <positive index>
             | <TEXT PRECISION>
                <text precision enumerated>
             | <CHARACTER EXPANSION FACTOR>
                <non-negative real>
             | <CHARACTER SPACING>
                <real>
             | <TEXT COLOUR>
                <colour>
             | <CHARACTER HEIGHT>
                <non-negative vdc value>
             | <CHARACTER ORIENTATION>
                <valid vdc vector>(2)
             | <TEXT PATH>
                <text path mode enumerated>
             | <TEXT ALIGNMENT>
                <horizontal alignment enumerated>
                <vertical alignment enumerated>
                <continuous alignment value> (2)
             | <CHARACTER SET INDEX>
                <positive index>
             | <ALTERNATE CHARACTER SET INDEX>
                <positive index>
             | <TEXT SCORE TYPE>
                <type-indicator pair>+
             | <RESTRICTED TEXT TYPE>
                <positive index>
```

```
<type-indicator pair>   ::=   <score type index>
              <off-on indicator enumerated>

<path enumerated> ::=   <RIGHT>
             | <LEFT>
             | <UP>
             | <DOWN>

<horizontal alignment
     enumerated>  ::=   <NORMAL HORIZONTAL>
             | <LEFT>
             | <CENTRE>
             | <RIGHT>
             | <CONTINUOUS HORIZONTAL>

<vertical alignment enumerated>     ::=   <NORMAL VERTICAL>
             | <TOP>
             | <CAP>
             | <HALF>
             | <BASE>
             | <BOTTOM>
             | <CONTINUOUS VERTICAL>

<score type index>      ::=   <positive index>

<continuous alignment value>  ::=   <real>

<filled-area attribute element>     ::=   <FILL BUNDLE INDEX>
                <positive index>
             | <INTERIOR STYLE>
                <interior style enumerated>
             | <FILL COLOUR>
                <colour>
             | <HATCH INDEX>
                <index>
             | <PATTERN INDEX>
                <positive index>
             | <FILL REFERENCE POINT>
                <point>
             | <PATTERN SIZE>
                <valid size vector>(2)
             | <INTERPOLATED INTERIOR>
                <positive index>
                <geometry definition>
                <number of stages>
                <stage designators>
                <reference colour list>
             | <edge attribute element>

<geometry definition>   ::=   <non-zero size value>(2n) {n>0}

<number of stages>      ::=   <positive integer>

<stage designators>     ::=   <real>(m)

<reference colour list> ::=   <colour>(k) {k≥0}
```

NOTE The type of this data is different in this Version 4 grammar, reflecting the new INTERIOR STYLE SPECIFICATION MODE (ISSM) which selects the units to be 'absolute', 'scaled', 'fractional', or 'mm'. In Version 1 and Version 2 it is <integer> or <real> (reflecting ‘vdc' in clause 7), which is effectively the same as ‘absolute' (the default ISSM).

```
<edge attribute element>      ::=   <EDGE BUNDLE INDEX>
                <positive index>
             | <EDGE TYPE>
                <index>
             | <EDGE WIDTH>
                <size value>
             | <EDGE COLOUR>
                <colour>
             | <EDGE VISIBILITY>
                <off-on indicator enumerated>
             | <EDGE CAP>
                <positive index>
                <positive index>
             | <EDGE JOIN>
                <positive index>
             | <EDGE TYPE CONTINUATION>
                <positive index>
             | <EDGE TYPE INITIAL OFFSET>
                <real>

<colour table element>  ::=   <COLOUR TABLE>
                <starting index>
                <colour direct>+

<starting index>  ::=   <colour index>

<pattern table element> ::=   <PATTERN TABLE>
                <positive index>
                <positive integer>(2)
                <local colour precision>
                <colour>(integer1 x integer2)

                {this element has an encoding}
                {dependent parameter}

<aspect source flags>   ::=   <ASPECT SOURCE FLAGS>
                <asf pair>+

<asf pair>  ::= <asf type enumerated>
              <asf enumerated>

<asf type enumerated>   ::=   <LINE TYPE ASF>
             | <LINE WIDTH ASF>
             | <LINE COLOUR ASF>
             | <MARKER TYPE ASF>
             | <MARKER SIZE ASF>
             | <MARKER COLOUR ASF>
             | <TEXT FONT ASF>
             | <TEXT PRECISION ASF>
             | <CHARACTER EXPANSION FACTOR ASF>
             | <CHARACTER SPACING ASF>
             | <TEXT COLOUR ASF>
             | <INTERIOR STYLE ASF>
             | <FILL COLOUR ASF>
             | <HATCH INDEX ASF>
             | <PATTERN INDEX ASF>
             | <EDGE TYPE ASF>
             | <EDGE WIDTH ASF>
             | <EDGE COLOUR ASF>

<asf enumerated>  ::=   <INDIVIDUAL>
             | <BUNDLED>
```

<pick identifier> ::= <PICK IDENTIFIER>
<name>

<symbol attribute element> ::= <SYMBOL LIBRARY INDEX>
<positive index>
<SYMBOL COLOUR>
<colour>
<SYMBOL SIZE>
<scale enumerated>
<positive vdc value>(2)
<SYMBOL ORIENTATION>
<valid vdc vector>(2)

<scale enumerated> ::= <HEIGHT>
| <WIDTH>
| <BOTH>

H.3.7 Escape elements

<escape element> ::= <ESCAPE>
<identifier>
<data record>

<identifier> ::= <integer>

H.3.8 External elements

<external element> ::= <MESSAGE>
<action flag enumerated>
<string fixed>
| <APPLICATION DATA>
<integer>
<data record>

<action flag enumerated> ::= <NO>
| <YES>

H.3.9 Segment elements

<segment control element> ::= <COPY SEGMENT>
<segment identifier>
<copy transformation matrix>
<segment transformation application>
| <inheritance element>

<inheritance element> ::= <INHERITANCE FILTER>
<filter selection list enumerated>*
<selection setting enumerated>
| <CLIP INHERITANCE>
<clip inheritance enumerated>

<segment attribute element> ::= <SEGMENT TRANSFORMATION>
<segment identifier>
<transformation matrix>
| <SEGMENT HIGHLIGHTING>
<segment identifier>
<highlighting enumerated>
| <SEGMENT DISPLAY PRIORITY>
<segment identifier>
<segment display priority>
| <SEGMENT PICK PRIORITY>

<segment identifier>
<segment pick priority>

<copy transformation matrix> ::= <transformation matrix>

<transformation matrix> ::= <2 x 2 matrix of reals>
<2 x 1 matrix of vdcs>

<segment transformation application> ::= <NO>
| <YES>

<filter selection list enumerated> ::= <attribute and control name enumerated>
| <attribute and control group enumerated>
| <asf name enumerated>
| <asf group enumerated>

<attribute and control
name enumerated> ::= <IH LINE BUNDLE INDEX>
| <IH LINE TYPE>
| <IH LINE WIDTH>
| <IH LINE COLOUR>
| <IH LINE CLIPPING MODE>
| <IH MARKER BUNDLE INDEX>
| <IH MARKER TYPE>
| <IH MARKER SIZE>
| <IH MARKER COLOUR>
| <IH MARKER CLIPPING MODE>
| <IH TEXT BUNDLE INDEX>
| <IH TEXT FONT INDEX>
| <IH TEXT PRECISION>
| <IH CHARACTER EXPANSION FACTOR>
| <IH CHARACTER SPACING>
| <IH TEXT COLOUR>
| <IH CHARACTER HEIGHT>
| <IH CHARACTER ORIENTATION>
| <IH TEXT PATH>
| <IH TEXT ALIGNMENT>
| <IH FILL BUNDLE INDEX>
| <IH INTERIOR STYLE>
| <IH FILL COLOUR>
| <IH HATCH INDEX>
| <IH PATTERN INDEX>
| <IH EDGE BUNDLE INDEX>
| <IH EDGE TYPE>
| <IH EDGE WIDTH>
| <IH EDGE COLOUR>
| <IH EDGE VISIBILITY>
| <IH EDGE CLIPPING MODE>
| <IH FILL REFERENCE POINT>
| <IH PATTERN SIZE>
| <IH AUXILIARY COLOUR>
| <IH TRANSPARENCY>
| <IH MITRE LIMIT>
| <IH LINE CAP>
| <IH LINE JOIN>
| <IH LINE TYPE CONTINUATION>
| <IH LINE TYPE INITIAL OFFSET>
| <IH TEXT SCORE TYPE>
| <IH RESTRICTED TEXT TYPE>
| <IH INTERPOLATED INTERIOR>
| <IH EDGE CAP>

```
             | <IH EDGE JOIN>
             | <IH EDGE TYPE CONTINUATION>
             | <IH EDGE TYPE INITIAL OFFSET>
             | <IH SYMBOL LIBRARY INDEX>
             | <IH SYMBOL COLOUR>
             | <IH SYMBOL SIZE>
             | <IH SYMBOL ORIENTATION>

<attribute and control
        group enumerated>     ::=   <LINE ATTRIBUTES>
             | <MARKER ATTRIBUTES>
             | <TEXT PRESENTATION AND PLACEMENT ATTRIBUTES>
             | <TEXT PLACEMENT AND ORIENTATION ATTRIBUTES>
             | <FILL ATTRIBUTES>
             | <EDGE ATTRIBUTES>
             | <PATTERN ATTRIBUTES>
             | <OUTPUT CONTROL>
             | <PICK IDENTIFIER>
             | <ALL ATTRIBUTES AND CONTROL>
             | <ALL>
             | <SYMBOL ATTRIBUTES>

<selection setting enumerated>      ::=   <STATE LIST>
             | &lt;SEGMENT&gt;

<asf name enumerated>   ::=   <IH LINE TYPE ASF>
             | <IH LINE WIDTH ASF>
             | <IH LINE COLOUR ASF>
             | <IH MARKER TYPE ASF>
             | <IH MARKER SIZE ASF>
             | <IH MARKER COLOUR ASF>
             | <IH TEXT FONT INDEX ASF>
             | <IH TEXT PRECISION ASF>
             | <IH CHARACTER EXPANSION FACTOR ASF>
             | <IH CHARACTER SPACING ASF>
             | <IH TEXT COLOUR ASF>
             | <IH INTERIOR STYLE ASF>
             | <IH FILL COLOUR ASF>
             | <IH HATCH INDEX ASF>
             | <IH PATTERN INDEX ASF>
             | <IH EDGE TYPE ASF>
             | <IH EDGE WIDTH ASF>
             | <IH EDGE COLOUR ASF>

<asf group enumerated>  ::=   <LINE ASFS>
             | <MARKER ASFS>
             | <TEXT ASFS>
             | <FILL ASFS>
             | <EDGE ASFS>
             | <ALL ASFS>

 clip inheritance enumerated> ::=   <STATE LIST>
             | <INTERSECTION>

<highlighting enumerated>     ::=   <NORMAL>
             | <HIGHLIGHTED>

&lt;segment display priority&gt;    ::=   <non-negative integer>

&lt;segment pick priority&gt; ::=   <non-negative integer>
```

H.3.10 Application structure descriptor elements

```
<application structure descriptor element>      ::=
           <APPLICATION STRUCTURE  ATTRIBUTE>
             <application structure attribute type>
             <data record>

<application structure attribute type>       ::=      <string fixed>

<data record> ::= <structured data record>
```

H.4 Terminal Symbols

The following are the terminals in this grammar. Their representation is dependent on the encoding scheme used. In Annex A of the subsequent parts of this Standard, these encoding-dependent symbols are further described.

<element name>
<integer>
<real>
<vdc value>
<string>
<string fixed>
<colour index>
<colour component>
<colour direct>
<integer precision value>
<real precision value>
<index precision value>
<colour precision value>
<colour index precision value>
<name precision value>
<default colour precision indicator>
<vdc integer precision value>
<vdc real precision value>
<data record>
<name>
<vc value>
<bitstream>
<2 x 2 matrix of reals>
<2 x 1 matrix of vdcs>
<3 x 3 matrix of reals>

The CGM extended opcodes are encoding dependent. A complete list of them can be found in the productions for <element name> below.

The enumerated types are:

<INTEGER>
<REAL>
<ON>
<OFF>
<INDEXED>
<DIRECT>
<ABSTRACT>
<METRIC>
<ABSOLUTE>
<SCALED>
<94 CHAR>
<96 CHAR>
<MULTI-BYTE 94 CHAR>

<MULTI-BYTE 96 CHAR>
<COMPLETE CODE>
<BASIC 7-BIT>
<BASIC 8-BIT>
<EXTENDED 7-BIT>
<EXTENDED 8-BIT>
<FRACTION OF DISPLAY SURFACE>
<MILLIMETRES WITH SCALE FACTOR>
<PHYSICAL DEVICE COORDINATES>
<NOT FORCED>
<FORCED>
<LEFT>
<RIGHT>
<CENTRE>
<BOTTOM>
<TOP>
<LOCUS>
<SHAPE>
<LOCUS THEN SHAPE>
<INVISIBLE>
<VISIBLE>
<CLOSE INVISIBLE>
<CLOSE VISIBLE>
<PIE>
<CHORD>
<FINAL>
<NOT FINAL>
<INDIVIDUAL>
<BUNDLED>
<HOLLOW>
<SOLID>
<PATTERN>
<HATCH>
<EMPTY>
<GEOMETRIC PATTERN>
<INTERPOLATED>
<STRING>
<CHARACTER>
<STROKE>
<UP>
<DOWN>
<NORMAL HORIZONTAL>
<CONTINUOUS HORIZONTAL>
<NORMAL VERTICAL>
<CAP>
<HALF>
<BASE>
<CONTINUOUS VERTICAL>
<YES>
<NO>
<LINE TYPE ASF>
<LINE WIDTH ASF>
<LINE COLOUR ASF>
<MARKER TYPE ASF>
<MARKER SIZE ASF>
<MARKER COLOUR ASF>
<TEXT FONT ASF>
<TEXT PRECISION ASF>
<CHARACTER EXPANSION FACTOR ASF>
<CHARACTER SPACING ASF>
<TEXT COLOUR ASF>

<INTERIOR STYLE ASF>
<HATCH INDEX ASF>
<PATTERN INDEX ASF>
<FILL COLOUR ASF>
<EDGE TYPE ASF>
<EDGE WIDTH ASF>
<EDGE COLOUR ASF>
<LINE ATTRIBUTES>
<MARKER ATTRIBUTES>
<TEXT PRESENTATION AND PLACEMENT ATTRIBUTES>
<TEXT PLACEMENT AND ORIENTATION ATTRIBUTES>
<FILL ATTRIBUTES>
<EDGE ATTRIBUTES>
<PATTERN ATTRIBUTES>
<OUTPUT CONTROL>
<PICK IDENTIFIER>
<ALL ATTRIBUTES AND CONTROL>
<ALL>
<LINE BUNDLE INDEX>
<LINE TYPE>
<LINE WIDTH>
<LINE COLOUR>
<LINE CLIPPING MODE>
<MARKER BUNDLE INDEX>
<MARKER TYPE>
<MARKER SIZE>
<MARKER COLOUR>
<MARKER CLIPPING MODE>
<TEXT BUNDLE INDEX>
<TEXT FONT INDEX>
<TEXT PRECISION>
<CHARACTER EXPANSION FACTOR>
<CHARACTER SPACING>
<TEXT COLOUR>
<CHARACTER HEIGHT>
<CHARACTER ORIENTATION>
<TEXT PATH>
<TEXT ALIGNMENT>
<FILL BUNDLE INDEX>
<INTERIOR STYLE>
<FILL COLOUR>
<HATCH INDEX>
<PATTERN INDEX>
<EDGE BUNDLE INDEX>
<EDGE TYPE>
<EDGE WIDTH>
<EDGE COLOUR>
<EDGE VISIBILITY>
<EDGE CLIPPING MODE>
<FILL REFERENCE POINT>
<PATTERN SIZE>
<AUXILIARY COLOUR>
<TRANSPARENCY>
<LINE CAP>
<LINE JOIN>
<LINE TYPE CONTINUATION>
<LINE TYPE INITIAL OFFSET>
<TEXT SCORE TYPE>
<RESTRICTED TEXT TYPE>
<INTERPOLATED INTERIOR>
<EDGE CAP>

<EDGE JOIN>
<EDGE TYPE CONTINUATION>
<EDGE TYPE INITIAL OFFSET>
<MITRE LIMIT>
<SYMBOL LIBRARY INDEX>
<SYMBOL COLOUR>
<SYMBOL SIZE>
<SYMBOL ORIENTATION>
<SYMBOL ATTRIBUTES>
<STATE LIST>
<INTERSECTION>
<SEGMENT>
<LINE ASFS>
<MARKER ASFS>
<TEXT ASFS>
<FILL ASFS>
<EDGE ASFS>
<ALL ASFS>
<NORMAL>
<HIGHLIGHTED>
<DRAWING SET>
<DRAWING PLUS CONTROL SET>
<VERSION 2 SET>
<EXTENDED PRIMITIVES SET>
<VERSION 2 GKSM SET>
<VERSION 3 SET>
<0 DEGREES>
<90 DEGREES>
<180 DEGREES>
<270 DEGREES>
<FRACTIONAL>
<MILLIMETRES>
<NON-TANGENTIAL>
<AXIS-TANGENTIAL>
<PARALLEL>
<CROSSHATCH>
<HEIGHT>
<WIDTH>
<BOTH>
<IH LINE BUNDLE INDEX>
<IH LINE TYPE>
<IH LINE WIDTH>
<IH LINE COLOUR>
<IH LINE CLIPPING MODE>
<IH MARKER BUNDLE INDEX>
<IH MARKER TYPE>
<IH MARKER SIZE>
<IH MARKER COLOUR>
<IH MARKER CLIPPING MODE>
<IH TEXT BUNDLE INDEX>
<IH TEXT FONT INDEX>
<IH TEXT PRECISION>
<IH CHARACTER EXPANSION FACTOR>
<IH CHARACTER SPACING>
<IH TEXT COLOUR>
<IH CHARACTER HEIGHT>
<IH CHARACTER ORIENTATION>
<IH TEXT PATH>
<IH TEXT ALIGNMENT>
<IH FILL BUNDLE INDEX>
<IH INTERIOR STYLE>

<IH FILL COLOUR>
<IH HATCH INDEX>
<IH PATTERN INDEX>
<IH EDGE BUNDLE INDEX>
<IH EDGE TYPE>
<IH EDGE WIDTH>
<IH EDGE COLOUR>
<IH EDGE VISIBILITY>
<IH EDGE CLIPPING MODE>
<IH FILL REFERENCE POINT>
<IH PATTERN SIZE>
<IH AUXILIARY COLOUR>
<IH TRANSPARENCY>
<IH MITRE LIMIT>
<IH LINE CAP>
<IH LINE JOIN>
<IH LINE TYPE CONTINUATION>
<IH LINE TYPE INITIAL OFFSET>
<IH TEXT SCORE TYPE>
<IH RESTRICTED TEXT TYPE>
<IH INTERPOLATED INTERIOR>
<IH EDGE CAP>
<IH EDGE JOIN>
<IH EDGE TYPE CONTINUATION>
<IH EDGE TYPE INITIAL OFFSET>
<IH SYMBOL LIBRARY INDEX>
<IH SYMBOL COLOUR>
<IH SYMBOL SIZE>
<IH SYMBOL ORIENTATION>
<IH LINE TYPE ASF>
<IH LINE WIDTH ASF>
<IH LINE COLOUR ASF>
<IH MARKER TYPE ASF>
<IH MARKER SIZE ASF>
<IH MARKER COLOUR ASF>
<IH TEXT FONT INDEX ASF>
<IH TEXT PRECISION ASF>
<IH CHARACTER EXPANSION FACTOR ASF>
<IH CHARACTER SPACING ASF>
<IH TEXT COLOUR ASF>
<IH INTERIOR STYLE ASF>
<IH FILL COLOUR ASF>
<IH HATCH INDEX ASF>
<IH PATTERN INDEX ASF>
<IH EDGE TYPE ASF>
<IH EDGE WIDTH ASF>
<IH EDGE COLOUR ASF>
<APPLICATION STRUCTURE>
<UI8>
<UI16>
<UI32>

```
element name>::= <BEGIN METAFILE>
            | <END METAFILE>
            | <BEGIN PICTURE>
            | <BEGIN PICTURE BODY>
            | <END PICTURE>
            | <BEGIN SEGMENT>
            | <END SEGMENT>
            | <BEGIN FIGURE>
```

```
<END FIGURE>
<BEGIN PROTECTION REGION>
<END PROTECTION REGION>
<BEGIN COMPOUND LINE>
<END COMPOUND LINE>
<BEGIN COMPOUND TEXT PATH>
<END COMPOUND TEXT PATH>
<BEGIN TILE ARRAY>
<END TILE ARRAY>
<BEGIN APPLICATION STRUCTURE>
<BEGIN APPLICATION STRUCTURE BODY>
<END APPLICATION STRUCTURE>
<METAFILE VERSION>
<METAFILE DESCRIPTION>
<VDC TYPE>
<INTEGER PRECISION>
<REAL PRECISION>
<INDEX PRECISION>
<COLOUR PRECISION>
<COLOUR INDEX PRECISION>
<NAME PRECISION>
<MAXIMUM COLOUR INDEX>
<COLOUR VALUE EXTENT>
<METAFILE ELEMENT LIST>
<METAFILE DEFAULTS REPLACEMENT>
<FONT LIST>
<CHARACTER SET LIST>
<CHARACTER CODING ANNOUNCER>
<MAXIMUM VDC EXTENT>
&lt;SEGMENT PRIORITY EXTENT&gt;
<COLOUR MODEL>
<COLOUR CALIBRATION>
<FONT PROPERTIES>
<GLYPH MAPPING>
<SYMBOL LIBRARY LIST>
<PICTURE DIRECTORY>
<SCALING MODE>
<COLOUR SELECTION MODE>
<LINE WIDTH SPECIFICATION MODE>
<MARKER SIZE SPECIFICATION MODE>
<EDGE WIDTH SPECIFICATION MODE>
<VDC EXTENT>
<BACKGROUND COLOUR>
<DEVICE VIEWPORT>
<DEVICE VIEWPORT SPECIFICATION MODE>
<DEVICE VIEWPORT MAPPING>
<LINE REPRESENTATION>
<MARKER REPRESENTATION>
<TEXT REPRESENTATION>
<FILL REPRESENTATION>
<EDGE REPRESENTATION>
<INTERIOR STYLE SPECIFICATION MODE>
<LINE AND EDGE TYPE DEFINITION>
<HATCH STYLE DEFINITION>
<GEOMETRIC PATTERN DEFINITION>
<PICTURE DIRECTORY>
<VDC INTEGER PRECISION>
<VDC REAL PRECISION>
<AUXILIARY COLOUR>
<TRANSPARENCY>
<CLIP RECTANGLE>
<CLIP INDICATOR>
```

```
| <LINE CLIPPING MODE>
| <MARKER CLIPPING MODE>
| <EDGE CLIPPING MODE>
| <NEW REGION>
| <SAVE PRIMITIVE CONTEXT>
| <RESTORE PRIMITIVE CONTEXT>
| <PROTECTION REGION INDICATOR>
| <GENERALIZED TEXT PATH MODE>
| <MITRE LIMIT>
| <TRANSPARENT CELL COLOUR>
| <POLYLINE>
| <DISJOINT POLYLINE>
| <POLYMARKER>
| <TEXT>
| <RESTRICTED TEXT>
| <APPEND TEXT>
| <POLYGON>
| <POLYGON SET>
| <CELL ARRAY>
| <GDP>
| <RECTANGLE>
| <CIRCLE>
| <CIRCULAR ARC 3 POINT>
| <CIRCULAR ARC 3 POINT CLOSE>
| <CIRCULAR ARC CENTRE>
| <CIRCULAR ARC CENTRE CLOSE>
| <CIRCULAR ARC CENTRE REVERSED>
| <ELLIPSE>
| <ELLIPTICAL ARC>
| <ELLIPTICAL ARC CLOSE>
| <CONNECTING EDGE>
| <HYPERBOLIC ARC>
| <PARABOLIC ARC>
| <NON-UNIFORM B-SPLINE>
| <NON-UNIFORM RATIONAL B-SPLINE>
| <POLYBEZIER>
| <POLYSYMBOL>
| <BITONAL TILE>
| <TILE>
| <LINE BUNDLE INDEX>
| <LINE TYPE>
| <LINE WIDTH>
| <LINE COLOUR>
| <MARKER BUNDLE INDEX>
| <MARKER TYPE>
| <MARKER SIZE>
| <MARKER COLOUR>
| <TEXT BUNDLE INDEX>
| <TEXT FONT INDEX>
| <TEXT PRECISION>
| <CHARACTER EXPANSION FACTOR>
| <CHARACTER SPACING>
| <TEXT COLOUR>
| <CHARACTER HEIGHT>
| <CHARACTER ORIENTATION>
| <TEXT PATH>
| <TEXT ALIGNMENT>
| <CHARACTER SET INDEX>
| <ALTERNATE CHARACTER SET INDEX>
| <FILL BUNDLE INDEX>
| <INTERIOR STYLE>
| <FILL COLOUR>
```

| <HATCH INDEX>
| <PATTERN INDEX>
| <EDGE BUNDLE INDEX>
| <EDGE TYPE>
| <EDGE WIDTH>
| <EDGE COLOUR>
| <EDGE VISIBILITY>
| <FILL REFERENCE POINT>
| <PATTERN TABLE>
| <PATTERN SIZE>
| <COLOUR TABLE>
| <ASPECT SOURCE FLAGS>
| <PICK IDENTIFIER>
| <COPY SEGMENT>
| <INHERITANCE FILTER>
| <CLIP INHERITANCE>
| <SEGMENT TRANSFORMATION>
| <SEGMENT HIGHLIGHTING>
| <SEGMENT DISPLAY PRIORITY>
| <SEGMENT PICK PRIORITY>
| <ESCAPE>
| <MESSAGE>
| <APPLICATION DATA>
| <LINE CAP>
| <LINE JOIN>
| <LINE TYPE CONTINUATION>
| <LINE TYPE INITIAL OFFSET>
| <TEXT SCORE TYPE>
| <RESTRICTED TEXT TYPE>
| <INTERPOLATED INTERIOR>
| <EDGE CAP>
| <EDGE JOIN>
| <EDGE TYPE CONTINUATION>
| <EDGE TYPE INITIAL OFFSET>
| <SYMBOL LIBRARY INDEX>
| <SYMBOL COLOUR>
| <SYMBOL SIZE>
| <SYMBOL ORIENTATION>

(Blank page)

Annex I
(normative)

Proforma tables and font metrics

I.1 Proforma tables

Table 13 — Metafile rules

Element	Specifications – PPF	Specifications - Model Profile
T.13.1	Same as Model Profile •	
Encodings	Select one or more encodings Binary: • Character • Clear text: •	Select one or more encodings Binary: ☑ Clear text: ☑
T.13.2	Same as Model Profile •	
Number of pictures	Number of pictures permitted in a metafile minimum (•0)? maximum (•0 or no limit)? Other:	Number of pictures permitted in a metafile minimum (•0)? 1. maximum (•0 or no limit)? *No limit* . Other: ***None.***
T.13.3	Same as Model Profile •	
Empty pictures	Are pictures allowed which have no graphical primitives ? (yes/no) Other:	Are pictures allowed which have no graphical primitives ? (yes/no) ***Yes.*** Other: ***None.***
T.13.4	Same as Model Profile •	
Metafile size	Any restrictions on metafile size ? (yes/no) Other:	Any restrictions on metafile size ? (yes/no) ***No.*** Other: ***None.***

Table 14 — Multi-element rules

Functionality	Specifications - PPF	Specifications - Model Profile
T.14.1	Same as Model Profile •	
Colour References: 9.5.4.1	Select which rule applies to each metafile (choose 1): Either all colours or None. shall be defined. • All colours shall be defined. • No colours shall be defined. • Are colour indexes allowed to be redefined within a picture or metafile? (yes/no) Any restrictions on the number of distinct colours used within a picture or metafile? (Monochrome metafiles shall use at most two distinct colours.) Are conformance categories defined? (yes/no) If yes, specify. Other:	Select which rule applies to each metafile (choose 1): Either all colours or None. shall be defined. ☑ All colours shall be defined. • No colours shall be defined. • Are colour indexes allowed to be redefined within a picture or metafile? (yes/no) ***No.*** Any restrictions on the number of distinct colours used within a picture or metafile? (Monochrome metafiles shall use at most two distinct colours.) ***None.*** Are conformance categories defined? (yes/no) ***Yes.*** If yes, specify. ***3 categories: monochrome, greyscale, and colour.*** Other: ***None.***
T.14.2	Same as Model Profile •	
Line primitives - geometric degeneracies References: 9.5.4.3	Geometric degeneracies are: Permitted • Prohibited • If permitted, graphical meaning of the degeneracy: Other:	Geometric degeneracies are:Permitted ☑ Prohibited • If permitted, graphical meaning of the degeneracy: ***A line primitive element, whose entire locus is a single point, denotes a graphical dot which is a filled circle, with diameter equal to the current line width and colour equal to the current line colour.*** Other: ***None.***

Table 14 — Multi-element rules (Continued)

Functionality	Specifications - PPF	Specifications - Model Profile
T.14.3	Same as Model Profile •	
Filled area primitives - geometric degeneracies References: 9.5.4.4	Geometric degeneracies are: Permitted • Prohibited • If permitted, graphical meaning of the degeneracy: Other:	Geometric degeneracies are:Permitted ☑ Prohibited • If permitted, graphical meaning of the degeneracy: ***A filled-area primitive element, whose entire locus is either a single point or a line has the following meaning:*** - ***If the locus of a filled-area primitive is a single point, then the meaning is a dot (which is a filled circle).*** - ***If the locus of a filled-area primitive is a non-degenerate line segment, then the meaning is a line.*** ***The dot or line is displayed with the fill colour if EDGE VISIBILITY is 'off', unless INTERIOR STYLE is 'empty', in which case it is not rendered. If EDGE VISIBILITY is 'on', the interior treatment is the dot or line displayed in the fill colour, and then a dot or line superimposed with the current edge attributes.*** Other: ***None.***
T.14.4	Same as Model Profile •	
Graphical text strings References: 9.5.4.5	Minimum string length (bytes): Maximum string length (bytes): Any restrictions on the use of ISO/IEC 2022 switching controls? CO control codes (except NUL and ISO/IEC 2022 switching) are prohibited. Other:	Minimum string length (bytes): ***0.*** Maximum string length (bytes): ***254.*** Any restrictions on the use of ISO/IEC 2022 switching controls? CO control codes (except NUL and ISO/IEC 2022 switching) are prohibited. ***Any character set used in the metafile which is accessed by ISO/IEC 2022 switching techniques shall be in the Character Set List (defined in this profile).*** Other: ***None.***

Table 14 — Multi-element rules (Continued)

Functionality	**Specifications – PPF**	**Specifications - Model Profile**
T.14.5	Same as Model Profile •	
Non-graphical text strings References: 9.5.4.6	Maximum string length (bytes): for type SF: for type SF within type D: Format effectors and ESC: Permitted • Prohibited • Other C0 control codes (except NUL and ISO/IEC 2022 switching) are prohibited. Any limits on the set of acceptable character sets? Any restrictions on the use of ISO/IEC 2022 switching controls? Other:	Maximum string length (bytes): for type SF: ***254.*** . for type SF within type D: ***1024.*** Format effectors and ESC: Permitted ☑ Prohibited • Other C0 control codes (except NUL and ISO/IEC 2022 switching) are prohibited. Any limits on the set of acceptable character sets? ***The permitted character sets are ISO 8859-1 LHS No.1 and ISO 8859-1 RHS No.1.*** Any restrictions on the use of ISO/IEC 2022 switching controls? ***Any character set used in the metafile which is accessed by ISO/IEC 2022 switching techniques shall be in the character set list (defined in this profile).*** Other: ***None.***
T.14.6	Same as Model Profile •	
Data record strings References: 9.5.4.7	Maximum string length (bytes) or state "no limit": SDR-coding techniques must be used (see annex C.2.2). Other:	Maximum string length (bytes) or state "no limit": ***32767.*** SDR-coding techniques must be used (see annex C.2.2). Other: ***None.***

Table 15 —Delimiter elements

Element	Specifications - PPF	Specifications - Model Profile
T.15.0	Same as Model Profile •	
no-op [v1] References: Part 3, 8.2	Element is: Required • Permitted • The parameter value of this element is encoding dependent. This element is applicable only to binary encoding. It shall be included in the profile only if binary encoding is permitted or required. If binary encoding is permitted, is the element Required • Permitted • If permitted, are there any restrictions on the Parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. This element is applicable only to binary encoding. It shall be included in the profile only if binary encoding is permitted or required. If binary encoding is permitted, is the element Required • Permitted ☑ If permitted, are there any restrictions on the Parameter value? ***None.*** Other: ***None.***
T.15.1	Same as Model Profile •	
BEGIN METAFILE END METAFILE [v1] References: 7.2.1 7.2.2 9.5.4.6 T.14.5	Element is: Required ☑ The *metafile identifier* parameter shall follow the rules for non-graphical text, clause 9.5.4.6 and T.14.5. Other:	Element is: Required ☑ The *metafile identifier* parameter shall follow the rules for non-graphical text, clause 9.5.4.6 and T.14.5. Other: ***None.***

Table 15 —Delimiter elements (Continued)

Element	**Specifications - PPF**	**Specifications - Model Profile**
T.15.2	Same as Model Profile •	
BEGIN PICTURE BEGIN PICTURE BODY END PICTURE [v1] References: 7.2.3 7.2.4 7.2.5 9.5.4.6 T.14.5	Element is: Required • Permitted • Prohibited • The *picture identifier* shall follow the rules for non-graphical text, clause 9.5.4.6 and T.14.5. Number of occurrences of these elements allowed in the metafile: Other:	Element is: Required • Permitted ☑ Prohibited • The *picture identifier* shall follow the rules for non-graphical text, clause 9.5.4.6 and T.14.5. Number of occurrences of these elements allowed in the metafile: *No limit.* Other: ***None.***
T.15.3	Same as Model Profile •	
BEGIN SEGMENT END SEGMENT [v2] References: 7.2.6 7.2.7	Element is: Required • Permitted • Prohibited • Maximum number of simultaneously defined segments (both global and local) at any point in the metafile: Any limits on the number of elements or restrictions on which elements compose a segment? Is there any meaning given to the *segment identifier* parameter? (yes/no) If yes, specify. (Meaning shall have no graphical effect.) Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneously defined segments (both global and local) at any point in the metafile: ***1024.*** Any limits on the number of elements or restrictions on which elements compose a segment? ***None.*** Is there any meaning given to the *segment identifier* parameter? (yes/no) ***No.*** If yes, specify. (Meaning shall have no graphical effect). Other: ***When global segments are specified in the Metafile Descriptor, all global segment definitions shall follow all other Metafile Descriptor elements. When segments are specified in the Picture Descriptor, all such segment definitions shall follow all other Picture Descriptor elements.***

Table 15 —Delimiter elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.15.4	Same as Model Profile •	
BEGIN FIGURE END FIGURE [v2] References: 7.2.8 7.2.9	Element is: Required • Permitted • Prohibited • Limits on the number of elements or restrictions on which elements comprise a figure definition: Other:	Element is: Required • Permitted ☑ Prohibited • Limits on the number of elements or restrictions on which elements comprise a figure definition: ***Maximum number of elements = 128. No restrictions on which eligible elements may be included.*** Other: ***None.***
T.15.5	Same as Model Profile •	
BEGIN PROTECTION REGION END PROTECTION REGION [v3] References: 7.2.10 7.2.11	Element is: Required • Permitted • Prohibited • Maximum number of simultaneously defined protection regions: Maximum number of elements within each protection region: Is there any meaning to the *region index* parameter other than as a unique identifier for each protection region? (yes/no) If yes, specify. (Meaning shall have no graphical effect). Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneously defined protection regions: ***32.*** Maximum number of elements within each protection region: ***128.*** Is there any meaning to the *region index* parameter other than as a unique identifier for each protection region? (yes/no) ***No.*** If yes, specify. (Meaning shall have no graphical effect). Other: ***None.***

Table 15 —Delimiter elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.15.6	Same as Model Profile •	
BEGIN COMPOUND LINE END COMPOUND LINE [v3] References: 7.2.12 7.2.13	Element is: Required • Permitted • Prohibited • Limits on the number of elements and identity of elements comprising a path definition: Other:	Element is: Required • Permitted ✓ Prohibited • Limits on the number of elements and identity of elements comprising a path definition: ***Maximum number of elements is 128. No restrictions on which eligible elements may be included***. Other: ***None.***
T.15.7	Same as Model Profile •	
BEGIN COMPOUND TEXT PATH END COMPOUND TEXT PATH [v3] References: 7.2.14 7.2.15	Element is: Required • Permitted • Prohibited • Limits on the number and identity of elements comprising a path definition: Other:	Element is: Required • Permitted ✓ Prohibited • Limits on the number and identity of elements comprising a path definition: ***Maximum number of elements is 128. No restrictions on which eligible elements may be included.*** Other: ***None.***

Table 15 —Delimiter elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.15.8	Same as Model Profile •	
BEGIN TILE ARRAY END TILE ARRAY [v3] References: 7.2.16 7.2.17	Element is: Required • Permitted • Prohibited • Maximum number of tiles in path direction: Maximum number of tiles in line direction: Maximum number of cells/tile in path direction: Maximum number of cells/tile in line direction: Limits on pel path: Limits on line progression: Limits on image offset: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of tiles in path direction: ***16.*** Maximum number of tiles in line direction: ***16.*** Maximum number of cells/tile in path direction: ***1024.*** Maximum number of cells/tile in line direction: ***1024.*** Limits on pel path: ***None.*** Limits on line progression: ***None.*** Limits on image offset: ***None.*** Other: ***None.***
T.15.9	Same as Model Profile •	
BEGIN APPLICATION STRUCTURE BEGIN APPLICATION STRUCTURE BODY END APPLICATION STRUCTURE [V4] References : 7.2.18 7.2.19 7.2.20	Element is: Required • Permitted • Prohibited • Limits on the maximum number of defined structures within a picture: Limits on the number and identity of elements comprizing a structure: Is there any meaning to the *application structure identifier* parameter? yes/no If yes, specify. Is the inhreritance flag parameter restricted? yes/no Other:	Element is: Required • Permitted ☑ Prohibited •• Limits on the maximum number of defined structures within a picture: ***None.*** Limits on the number and identity of elements comprizing a structure: ***None.*** Is there any meaning to the *application structure identifier* parameter? yes/no ***No. No assigned meaning beyond being a unique identifier for the application structure.*** If yes, specify. Is the inhreritance flag parameter restricted? yes/no **No.** Other: ***None.***

Table 16 — Metafile descriptor elements

Element	Specifications – PPF	Specifications - Model Profile
T.16.1	Same as Model Profile •	
METAFILE VERSION [v1] References: 7.3.1	Element is: Required ☑ Metafile versions permitted by this profile: Other:	Element is: Required ☑ Metafile versions permitted by this profile: *1, 2, 3, 4.* Other: ***None.***
T.16.2	Same as Model Profile •	
METAFILE DESCRIPTION [v1] References: 7.3.2 9.5.2.1 9.5.2.2 9.5.4.6 T.14.1 T.14.5	Element is: Required ☑ The *description* parameter shall follow the rules for non-graphical text, clause 9.5.4.6 and T.14.5. The substring within the SF parameter shall be of the form: "keyword:item", where the double quotes are part of the substring. Maximum number of occurrences of this element? Profile identification (use keyword, "ProfileId:"): Profile edition (use keyword, "ProfileEd:"): If the profile edition is not given, then the edition defaults to 1.	Element is: Required ☑ The *description* parameter shall follow the rules for non-graphical text, clause 9.5.4.6 and T.14.5. The substring within the SF parameter shall be of the form: "keyword:item", where the double quotes are part of the substring. Maximum number of occurrences of this element? ***Unlimited.*** Profile identification (use keyword, "ProfileId:"): ***"ProfileId:Model-Profile".*** Profile edition (use keyword, "ProfileEd:"): ***"ProfileEd:2".*** If the profile edition is not given, then the edition defaults to 1.

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.2 continued METAFILE DESCRIPTION	Additional information content: Metafile colour conformance class, source, and date items shall be encoded as substrings of the *description* parameter using the keywords: "ColourClass:", "Source:", and "Date:", respectively. ColourClass: Required • Permitted • Content: Source: Required • Permitted • Content: Date: Required • Permitted • Content shall be date of metafile generation. Other:	Additional information content: Metafile colour conformance class, source, and date items shall be encoded as substrings of the *description* parameter using the keywords: "ColourClass:", "Source:", and "Date:", respectively. ColourClass: Required ☑ Permitted • Content: (One of: colour, greyscale, or monochrome.) Source: Required ☑ Permitted • Content: (Vendor, product, and version). Date: Required ☑ Permitted • Content shall be date of metafile generation***. The form and content shall be in accordance with ISO 8601:1988.*** Other: ***None.***
T.16.3	Same as Model Profile •	
VDC TYPE [v1] References: 7.3.3	Element is: Required • Permitted • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.16.4	Same as Model Profile •	
INTEGER PRECISION [v1] References: 7.3.4 Part 3, 8.3 Part 4, 7.2	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted are there any restrictions on the parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the parameter value? ***8, 16, or 32.*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***[-256,255], [-32767,32767], [-32768,32767] or [-2147483648, 2147483647]*** Other: ***None.***
T.16.5	Same as Model Profile •	
REAL PRECISION [v1] References: 7.3.5 Part 3, 8.3 Part 4, 7.2	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? ***(1, 16, 16) or (0, 9, 23)*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***-32767, +32767, 4; or*** ***-32768, +32767, 10; or*** ***-3.4028235E38, +3.4028235E38, 8*** ***Note: The latter two values are the closest approximation, in base 10 clear text, to the REAL PRECISION values allowed in binary encoded CGMs.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.6	Same as Model Profile •	
INDEX PRECISION [v1] References: 7.3.6	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted, if permitted, are there any restrictions on the parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, If permitted, are there any restrictions on the parameter value? ***8, 16, or 32.*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***[0, 127], [-256, 255], [-32767, 32767], [-32768, 32767], or [-2147483648, 2147483647]*** Other: ***None.***
T.16.7	Same as Model Profile •	
COLOUR PRECISION [v1] References: 7.3.7	Element is: Required ☐ Permitted ☐ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the parameter value? ***8 or 16.*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***255 or 65535.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.8	Same as Model Profile •	
COLOUR INDEX PRECISION [v1] References: 7.3.8	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the parameter value? ***8 or 16.*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***127, 255, 32767.*** Other: ***None.***
T.16.9	Same as Model Profile •	
MAXIMUM COLOUR INDEX [v1] References: 7.3.9	Element is: Required • Permitted • Prohibited • Is this element required to be a least upper bound? (yes/no) Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • Is this element required to be a least upper bound? (yes/no) ***No.*** Any restrictions on the parameter values? ***0-1*** ***for monochrome metafiles.*** ***0-63*** ***for greyscale metafiles.*** ***0-255*** ***for colour metafiles.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.16.10	Same as Model Profile •	
COLOUR VALUE EXTENT [v1] References: 7.3.10	Element is: Required • Permitted • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.16.11	Same as Model Profile •	
METAFILE ELEMENT LIST [v1] References: 7.3.11	Element is: Required ☑ Other:	Element is: Required ☑ Other: ***None.***
T.16.12	Same as Model Profile •	
METAFILE DEFAULTS REPLACEMENT [v1] References: 7.3.12	Element is: Required • Permitted • Prohibited • Is each occurrence of the MDR restricted to defining just one default? (yes/no) Additional restrictions may be specified in parts 3, and 4 of ISO/IEC 8632. NOTE — Profile specifications regarding use of MDR shall be consistent with other profile specifications. For example, if a profile restricts metafiles to a single picture, then it makes little sense for the profile to require the MDR element in metafiles. Other:	Element is: Required • Permitted ☑ Prohibited • Is each occurrence of the MDR restricted to defining just one default? (yes/no) ***No.*** Additional restrictions may be specified in parts 3, and 4 of ISO/IEC 8632. Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.13	Same as Model Profile •	
FONT LIST [v1] References: 7.3.13 I.2	Element is: Required • Permitted • Prohibited • This element is required for all metafiles containing graphical text. Maximum number of fonts in the list: All font indexes referenced in the metafile, including the default (nominally index 1) shall be defined in the FONT LIST element, with font name construction consistent with the rules of ISO/IEC 9541. List of permitted fonts: Other:	Element is: Required • Permitted ☑ Prohibited • This element is required for all metafiles containing graphical text. Maximum number of fonts in the list: ***64.*** All font indexes referenced in the metafile, including the default (nominally index 1) shall be defined in the FONT LIST element, with font name construction consistent with the rules of ISO/IEC 9541. List of permitted fonts: ***Times-Roman*** ***Helvetica-BoldOblique*** ***Times-Bold*** ***Courier*** ***Times-Italic*** ***Courier-Bold*** ***Times-BoldItalic*** ***Courier-Oblique*** ***Helvetica*** ***Courier-BoldOblique*** ***Helvetica-Bold*** ***Symbol*** ***Helvetica-Oblique*** ***NOTE — These font names are trademarked and some are proprietary and copyrighted. Times and Helvetica are registered trademarks of Allied Corporation, the owner of the copyright on the fonts of those names. Metric equivalents of the named fonts may be substituted by interpreters. Times is a serif font. Helvetica is a sans-serif font. Courier is a monospaced, serif font. The association of character code to glyph which shall be used for each of the fonts and the metrics of the named fonts are contained in clause I.2, annex I.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.14 CHARACTER SET LIST [v1] References: 7.3.14	Same as Model Profile •	
	Element is: Required • Permitted • Prohibited • This element is required for all metafiles containing graphical text. Maximum limit for the number of character sets in the character set list: Character sets shall be selected from the ISO Registry of Character Sets. This list may be extended by adding profile-defined character sets. List character sets: If any of these character sets is of type "complete code", specify the content of the complete code and its associated sequence tail: Other:	Element is: Required • Permitted ☑ Prohibited • This element is required for all metafiles containing graphical text. Maximum limit for the number of character sets in the character set list: ***4.*** Character sets shall be selected from the ISO Registry of Character Sets. This list may be extended by adding profile-defined character sets. List character sets: ***"94-character G-set", 4/2 (ISO 8859-1 LH);*** ***"96-character G-set", 4/1 (ISO 8859-1 RH);*** ***"94-character G-set", 2/10 3/10 (Symbol LH);*** ***"94-character G-set", 2/6 3/10 (Symbol RH).*** If any of these character sets is of type "complete code", specify the content of the complete code and its associated sequence tail: ***Not applicable.*** Other: ***None.***
T.16.15 CHARACTER CODING ANNOUNCER [v1] References: 7.3.15	Same as Model Profile •	
	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter values? ***Values shall be 'basic 7-bit' and 'basic 8-bit'.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.16.16	Same as Model Profile •	
NAME PRECISION [v2] References: 7.3.16 Part 3, 8.3 Part 4, 7.2	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? ***16 or 32.*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***[-256, 255], [-32767, 32767], [-32768, 32767], or [-2147483648, 2147483647]*** Other: ***None.***
T.16.17	Same as Model Profile •	
MAXIMUM VDC EXTENT [v2] References: 7.3.17	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter values? ***None.*** Other: ***None.***
T.16.18	Same as Model Profile •	
SEGMENT PRIORITY EXTENT [v2] References: 7.3.18	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter values? ***None.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.19	Same as Model Profile •	
COLOUR MODEL [v3] References: 7.3.19	Element is: Required • Permitted • Prohibited • Any restrictions on the set of colour models? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the set of colour models? ***None.*** Other: ***None.*** ***NOTE — Three colour models have been registered (as of date of publication) in the ISO Register of Graphical Items: RGB-alpha, sRGB, sRGB-alpha***
T.16.20	Same as Model Profile •	
COLOUR CALIBRATION [v3] References: 7.3.20	Element is: Required • Permitted • Prohibited • Calibration selection values permitted in accordance with the permitted model(s): If CYMK is permitted, minimum number of grid locations: Any restrictions on the number of colour lookup table entries, n? Any restrictions on the number of grid locations, m? If CYMK is permitted, algorithms for interpolation between grid locations? Other:	Element is: Required • Permitted ☑ Prohibited • Calibration selection values permitted in accordance with the permitted model(s): Values ***1..6, 9.*** If CYMK is permitted, minimum number of grid locations: ***3.*** Any restrictions on the number of colour lookup table entries, n? ***None.*** Any restrictions on the number of grid locations, m? ***None.*** If CYMK is permitted, algorithms for interpolation between grid locations? ***None.*** Other: ***None.***
T.16.21	Same as Model Profile •	
FONT PROPERTIES [v3] References: 7.3.21	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter values? ***All defined index and enumerated values of all parameters shall be permitted.*** Other: ***None.***

Table 16 — Metafile descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.16.22	Same as Model Profile •	
GLYPH MAPPING [v3] References: 7.3.22	Element is: Required • Permitted • Prohibited • Subset of AFII registered glyphs which may be referenced: Maximum number of glyphs which may be defined: Other:	Element is: Required • Permitted ✓ Prohibited • Subset of AFII registered glyphs which may be referenced: ***None.*** Maximum number of glyphs which may be defined: ***8192.*** Other: ***None.***
T.16.23	Same as Model Profile •	
SYMBOL LIBRARY LIST [v3] References: 7.3.23	Element is: Required • Permitted • Prohibited • Libraries which may be accessed and their encoding rules: Maximum number of libraries which may be accessed: Other:	Element is: Required • Permitted • Prohibited ✓ Libraries which may be accessed and their encoding rules: Maximum number of libraries which may be accessed: Other: ***NOTE — There are currently no registered symbol libraries.***
T.16.24	Same as Model Profile •	
PICTURE DIRECTORY [V4] References : 7.3.24 9.5.4.6 T.14.5	Element is: Required • Permitted • Prohibited • Follows rules for non-graphical text strings for picture identifier parameter, clause 9.5.4.6 and T.14.5. If present, shall PICTURE DIRECTORY elements be complete, i.e., have an entry for every picture in the metafile? (yes/no) If "no", describe any special meaning associated with those entries which appear in PICTURE DIRECTORY elements which are incomplete. Other:	Element is: Required • Permitted ✓ Prohibited • Follows rules for non-graphical text strings for picture identifier parameter, clause 9.5.4.6 and T.14.5. If present, shall PICTURE DIRECTORY elements be complete, i.e., have an entry for every picture in the metafile? (yes/no) ***Yes.*** If "no", describe any special meaning associated with those entries which appear in PICTURE DIRECTORY elements which are incomplete. Other: ***None.***

Table 17 — Picture descriptor elements

Functionality	Specifications – PPF	Specifications - Model Profile
T.17.1	Same as Model Profile •	
SCALING MODE [v1] References: 7.4.1	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter values? ***If SCALING MODE is metric then the 'metric scale factor' shall be positive.*** Other: ***None.***
T.17.2	Same as Model Profile •	
COLOUR SELECTION MODE [v1][v2] References: 7.4.2	Element is: Required • Permitted • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Any restrictions on the parameter values? ***None.*** Other: ***None.***
T.17.3	Same as Model Profile •	
LINE WIDTH SPECIFICATION MODE [v1][v2] References: 7.4.3	Element is: Required • Permitted • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Any restrictions on the parameter values? ***None.*** Other: ***None.***

Table 17 — Picture descriptor elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.17.4	Same as Model Profile •	
MARKER SIZE SPECIFICATION MODE [v1][v2] References: 7.4.4	Element is: Required • Permitted • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Any restrictions on the parameter values? ***None.*** Other: ***None.***
T.17.5	Same as Model Profile •	
EDGE WIDTH SPECIFICATION MODE [v1][v2] References: 7.4.5	Element is: Required • Permitted • Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Any restrictions on the parameter values? ***None.*** Other: ***None.***
T.17.6	Same as Model Profile •	
VDC EXTENT [v1] References: 7.4.6	Element is: Required • Permitted • Limits on the sense and orientation of the VDC space: Is zero-area VDC extent permitted? (yes/no) If yes, specify its meaning. Other:	Element is: Required • Permitted ☑ Limits on the sense and orientation of the VDC space: ***None.*** Is zero-area VDC extent permitted? (yes/no) ***No.*** If yes, specify its meaning. Other: ***None.***

Table 17 — Picture descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.17.7	Same as Model Profile •	
BACKGROUND COLOUR [v1] References: 7.4.7 9.5.4.1 T.14.1	Element is: Required • Permitted • The colour value parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Other:	Element is: Required • Permitted ☑ The colour value parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Other: ***None.***
T.17.8	Same as Model Profile •	
DEVICE VIEWPORT [v2] References: 7.4.8	Element is: Required • Permitted • Interaction of this element with environmental presentation directives: Meaning of this element if the specified value is inconsistent with the presentation device: Other:	Element is: Required • Permitted • Prohibited ☑ Interaction of this element with environmental presentation directives: Meaning of this element if the specified value is inconsistent with the presentation device: Other: ***NOTE — This element is prohibited due to its device dependence.***
T.17.9	Same as Model Profile •	
DEVICE VIEWPORT SPECIFICATION MODE [v2] References: 7.4.9	Element is: Required • Permitted • Prohibited • Set of legal values: Other:	Element is: Required • Permitted • Prohibited ☑ Set of legal values: Other: ***NOTE — This element is prohibited due to its device dependence.***

Table 17 — Picture descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.17.10	Same as Model Profile •	
DEVICE VIEWPORT MAPPING [v2] References: 7.4.10	Element is: Required • Permitted • Prohibited • Set of legal values: Other:	Element is: Required • Permitted • Prohibited ☑ Set of legal values: Other: ***NOTE — This element is prohibited due to its device dependence.***
T.17.11	Same as Model Profile •	
LINE REPRESENTATION [v2] References: 7.4.11 9.5.2.6 9.5.4.2 T.20.1	Element is: Required • Permitted • Prohibited • Maximum number of simultaneous bundle definitions: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneous bundle definitions: ***20.*** Other: ***None.***
T.17.12	Same as Model Profile •	
MARKER REP-RESENTATION [v2] References: 7.4.12 9.5.2.6 9.5.4.2 T.20.5	Element is: Required • Permitted • Prohibited • Maximum number of simultaneous bundle definitions: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneous bundle definitions: ***20.*** Other: ***None.***

Table 17 — Picture descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.17.13	Same as Model Profile •	
TEXT REPRESENTATION [v2] References: 7.4.13 9.5.2.6 9.5.4.2 T.20.9	Element is: Required • Permitted • Prohibited • Maximum number of simultaneous bundle definitions: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneous bundle definitions: ***20.*** Other: ***None.***
T.17.14	Same as Model Profile •	
FILL REPRESENTATION [v2] References: 7.4.14 9.5.2.6 9.5.4.2 T.20.21	Element is: Required • Permitted • Prohibited • Maximum number of simultaneous bundle definitions: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneous bundle definitions: ***20.*** Other: ***None.***
T.17.15	Same as Model Profile •	
EDGE REPRESENTATION [v2] References: 7.4.15 9.5.2.6 9.5.4.2 T.20.26	Element is: Required • Permitted • Prohibited • Maximum number of simultaneous bundle definitions: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneous bundle definitions: ***20.*** Other: ***None.***

Table 17 — Picture descriptor elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.17.16	Same as Model Profile •	
INTERIOR STYLE SPECIFICATION MODE [v3] References: 7.4.16	Element is: Required • Permitted • Any restriction on the parameter value? Other:	Element is: Required • Permitted ☑ Any restriction on the parameter value? ***None.*** Other: ***None.***
T.17.17	Same as Model Profile •	
LINE AND EDGE TYPE DEFINITION [v3] References: 7.4.17	Element is: Required • Permitted • Prohibited • Any limits on the number of definitions? Any limits on the number of elements in a given definition? Any restrictions on the dash cycle repeat length? Any restrictions on complexity of definition to prevent degeneracies? Other:	Element is: Required • Permitted ☑ Prohibited • Any limits on the number of definitions? ***Maximum of 32 line types shall be specified simultaneously.*** Any limits on the number of elements in a given definition? ***Number of values in the dash gap list shall not exceed 8.*** Any restrictions on the dash cycle repeat length? ***None.*** Any restrictions on complexity of definition to prevent degeneracies? ***None.*** Other: ***None.***

Table 17 — Picture descriptor elements (Continued)

	Specifications - PPF	Specifications - Model Profile
T.17.18	Same as Model Profile •	
HATCH STYLE DEFINITION [v3] References: 7.4.18	Element is: Required • Permitted • Prohibited • Limit on the number of hatch styles? Limit on the number of gaps in a given definition? Any limits on duty cycle length? Any restrictions on complexity of definition to prevent degeneracies? Any restrictions on the style indicator: Other:	Element is: Required • Permitted ☑ Prohibited • Limit on the number of hatch styles? ***Maximum of 32 hatch styles shall be specified simultaneously.*** Limit on the number of gaps in a given definition? ***Number of entries in the gap width list shall not exceed 8.*** Any limits on duty cycle length? ***None.*** Any restrictions on complexity of definition to prevent degeneracies? ***None.*** Any restrictions on the style indicator? ***None.*** Other: ***None.***
T.17.19	Same as Model Profile •	
GEOMETRIC PATTERN DEFINITION [v3] References: 7.4.19	Element is: Required • Permitted • Prohibited • Any limits on the number of geometric patterns defined? NOTE — The number of geometric patterns cannot exceed the number of segments. Any limits on the classes of primitives? Other:	Element is: Required • Permitted ☑ Prohibited • Any limits on the number of geometric patterns defined? ***The maximum number of geometric patterns is 64.*** NOTE — The number of geometric patterns cannot exceed the number of segments. Any limits on the classes of primitives? ***None.*** Other: **None.**

Table 17 — Picture descriptor elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.17.20	Same as Model Profile •	
APPLICATION STRUCTURE DIRECTORY [V4] References : 7.4.20 9.5.4.6 T.14.5	Element is: Required • Permitted • Prohibited • Follows rules for non-graphical text strings for *application structure identifier* parameter, clause 9.5.4.6 and T.14.20. If present, shall APPLICATION STRUCTURE DIRECTORY elements be complete, i.e., have an entry for every application structure in the picture? (yes/no) If "no", describe any special meaning associated with those entries which appear in APPLICATION STRUCTURE DIRECTORY elements which are incomplete. Other:	Element is: Required • Permitted ☑ Prohibited • Follows rules for non-graphical text strings for *application structure identifier* parameter, clause 9.5.4.6 and T.14.20. If present, shall APPLICATION STRUCTURE DIRECTORY elements be complete, i.e., have an entry for every application structure in the picture? (yes/no) ***Yes.*** If "no", describe any special meaning associated with those entries which appear in APPLICATION STRUCTURE DIRECTORY elements which are incomplete. Other: ***None.***

Table 18 — Control elements

Element	Specifications – PPF	Specifications - Model Profile
T.18.1	Same as Model Profile	
VDC INTEGER PRECISION [v1] References: 7.5.1 Part 3, 8.5 Part 4, 7.4	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? ***16 or 32***. Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***[-32767, 32767], [-32768, 32767], or [-2147483648, 2147483647]***. Other: ***None.***
T.18.2	Same as Model Profile	
VDC REAL PRECISION [v1] References: 7.5.2 Part 3, 8.5 Part 4, 7.4	Element is: Required • Permitted • The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the parameter value? Other: If clear text encoding is permitted, are there any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ The parameter value of this element is encoding dependent. If binary encoding is permitted, are there any restrictions on the Parameter value? ***(1, 16, 16) or (0, 9, 32)*** Other: ***None.*** If clear text encoding is permitted, are there any restrictions on the parameter value? ***0.0, 1.0 , 4; or*** ***-32767, 32767, 4; or*** ***-32768, 32767, 10; or*** ***-3.4028235E38, +3.4028235E38, 8*** ***Note: The latter two values are the closest approximation, in base 10 clear text, to the REAL PRECISION values allowed in binary encoded CGMs.*** Other: ***None.***

Element	Specifications – PPF	Specifications - Model Profile
T.18.3 AUXILIARY COLOUR [v1] References: 7.5.3 9.5.4.1 T.14.1 D.4.4.1	Same as Model Profile Element is: Required • Permitted • Prohibited • The *auxiliary colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Other:	 Element is: Required • Permitted ☑ Prohibited • The *auxiliary colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Other: ***None.*.**
T.18.4 TRANSPARENCY [v1] References: 7.5.4 9.5.7.9 T.14.1 T.26.11	Same as Model Profile • Element is: Required • Permitted • Prohibited • Any restriction on the parameter value? Other:	 Element is: Required • Permitted ☑ Prohibited • Any restriction on the parameter value? ***None.*.** Other: ***None.***
T.18.5 CLIP RECTANGLE [v1] References: 7.5.5 D.4.4.2	Same as Model Profile • Element is: Required • Permitted • Prohibited • Meaning of boundary cases for: zero-area: area greater than VDC extent: additional cases? NOTE — Because objects "inside and on the boundary are drawn", then zero-area does not have the sometimes claimed effect of hiding subsequent primitives - there will be a visible effect, a dot or a line, if the object intersects the boundary of the degenerate area. Other:	 Element is: Required • Permitted ☑ Prohibited • Meaning of boundary cases for: zero-area: ***Prohibited.*** area greater than VDC extent: ***Clipping shall be done to the intersection of CLIP RECTANGLE and VDC EXTENT***. additional cases: ***None.*** Other: ***None.***

Table 18 — Control elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.18.6	Same as Model Profile •	
CLIP INDICATOR [v1] References: 7.5.6	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.18.7	Same as Model Profile •	
LINE CLIPPING MODE [v2] References: 7.5.7 D.4.4.3	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter value? ***None.*** Other: **None.**
T.18.8	Same as Model Profile •	
MARKER CLIPPING MODE [v2] References: 7.5.8 D.4.4.3	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.18.9	Same as Model Profile •	
EDGE CLIPPING MODE [v2] References: 7.5.9 D.4.4.3	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 18 — Control elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.18.10	Same as Model Profile •	
NEW REGION [v2] References: 7.5.10	Element is: Required • Permitted • Prohibited • This element shall be permitted only if BEGIN FIGURE is permitted. Any restrictions on the number of occurrences? Other:	Element is: Required • Permitted ☑ Prohibited • This element shall be permitted only if BEGIN FIGURE is permitted. Any restrictions on the number of occurrences? ***None.*** Other: ***None.***
T.18.11	Same as Model Profile •	
SAVE PRIMITIVE CONTEXT [v2] References: 7.5.11	Element is: Required • Permitted • Prohibited • Maximum number of simultaneously saved contexts: Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of simultaneously saved contexts: *1024.* Other: ***None.***
T.18.12	Same as Model Profile •	
RESTORE PRIMITIVE CONTEXT [v2] References: 7.5.12	Element is: Required • Permitted • Prohibited • This element is permitted only if SAVE PRIMITIVE CONTEXT is permitted. Other:	Element is: Required • Permitted ☑ Prohibited • This element is permitted only if SAVE PRIMITIVE CONTEXT is permitted. Other: ***None.***

Table 18 — Control elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.18.13	Same as Model Profile ☐	
PROTECTION REGION INDICATOR [v3] References: 7.5.13	Element is: Required ☐ Permitted ☐ Prohibited ☐ This element shall be permitted only if BEGIN PROTECTION REGION is permitted. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ This element shall be permitted only if BEGIN PROTECTION REGION is permitted. Other: ***None.***
T.18.14	Same as Model Profile ☐	
GENERALIZED TEXT PATH MODE [v3] References: 7.5.14	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.18.15	Same as Model Profile •	
MITRE LIMIT [v3] References: 7.5.15	Element is: Required • Permitted • Prohibited • Any restrictions on the parameter value? Other:	Element is: Required • Permitted ☑ Prohibited • Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.18.16	Same as Model Profile •	
TRANSPARENT CELL COLOUR [v3] References: 7.5.16 9.5.4.1 T.14.1	Element is: Required • Permitted • Prohibited • The *transparent cell colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter values? Other:	Element is: Required • Permitted ☑ Prohibited • The *transparent cell colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter values? ***None.*** Other: ***None.***

Table 19 — Graphical primitive elements

Element	Specifications - PPF	Specifications - Model Profile
T.19.1	Same as Model Profile •	
POLYLINE [v1] References: 7.6.1 T.14.2 D.2.21	Element is: Required • Permitted • Prohibited • Maximum number of points or state "no limit": Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of points or state "no limit": ***4096.*** Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.2	Same as Model Profile •	
DISJOINT POLYLINE [v1] References: 7.6.2 T.14.2 D.2.2.1	Element is: Required • Permitted • Prohibited • Maximum number of points or state "no limit": Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of points or state "no limit": ***4096.*** Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.3	Same as Model Profile •	
POLYMARKER [v1] References: 7.6.3	Element is: Required • Permitted • Prohibited • Maximum number of points or state "no limit": Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of points or state "no limit": ***4096.*** Other: ***None.***
T.19.4	Same as Model Profile •	
TEXT [v1] References: 7.6.4 9.5.4.5	Element is: Required • Permitted • Prohibited • The *string* parameter shall follow the rules for graphical text, clause 9.5.4.5. Is the 'not final' flag allowed: (yes/no) Other:	Element is: Required • Permitted ☑ Prohibited • The *string* parameter shall follow the rules for graphical text, clause 9.5.4.5. Is the 'not final' flag allowed: (yes/no) ***Yes.*** Other: ***None.***

Table 19 — Graphical primitive elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.19.5	Same as Model Profile •	
RESTRICTED TEXT [v1] References: 7.6.5 9.5.4.5 T.26.7 D.4.5.2	Element is: Required • Permitted • Prohibited • The *string* parameter shall follow the rules for graphical text, clause 9.5.4.5. Is the 'not final' flag allowed: (yes/no) For [v1/2] metafiles, is the realization of RESTRICTED TEXT according to one of the standard or registered values for RESTRICTED TEXT TYPE? (yes/no) If yes, specify. For [v3] and [v4] metafiles, RESTRICTED TEXT TYPE shall be used if this element is used. Other:	Element is: Required • Permitted ☑ Prohibited • The *string* parameter shall follow the rules for graphical text, clause 9.5.4.5. Is the 'not final' flag allowed: (yes/no) ***Yes.*** For [v1/2] metafiles, is the realization of RESTRICTED TEXT according to one of the standard or registered values for RESTRICTED TEXT TYPE? (yes/no) ***Yes.*** If yes, specify. Boxed-cap, also see T.26.7 For [v3] and [v4] metafiles, RESTRICTED TEXT TYPE shall be used if this element is used. Other: ***None.***
T.19.6	Same as Model Profile •	
APPEND TEXT [v1] References: 7.6.6 9.5.4.5 D.4.5.1	Element is: Required • Permitted • Prohibited • The *string* parameter shall follow the rules for graphical text, clause 9.5.4.5. Other:	Element is: Required • Permitted ☑ Prohibited • The *string* parameter shall follow the rules for graphical text, clause 9.5.4.5. Other: ***None.***
T.19.7	Same as Model Profile •	
POLYGON [v1] References: 7.6.7 T.14.3 D.2.2.2	Element is: Required • Permitted • Prohibited • Maximum number of points: Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	Element is: Required • Permitted ☑ Prohibited • Maximum number of points: ***4096.*** Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***None.***

Table 19 — Graphical primitive elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.19.8	Same as Model Profile •	
POLYGON SET [v1] References: 7.6.8 T.14.3 D.2.2.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ Maximum number of points: Number of polygons in a set? Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Maximum number of points: ***4096.*** Number of polygons in a set? ***No limit.*** Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***Each individual polygon within a set shall have at least 3 points***.
T.19.9	Same as Model Profile ☐	
CELL ARRAY [v1] References: 7.6.9 D.4.5.3	Element is: Required • Permitted • Prohibited • Limit for nx: Limit for ny: Limit for nx·ny: Are rotated and skewed cell arrays allowed? (yes/no) If yes, specify the graphical meaning. Other:	Element is: Required • Permitted ☑ Prohibited • Limit for nx: ***2048.*** Limit for ny: ***2048.*** Limit for nx·ny: ***4194304.*** Are rotated and skewed cell arrays allowed? (yes/no) ***No.*** If yes, specify the graphical meaning. Other: ***Zero-area cell arrays are prohibited***.
T.19.10	Same as Model Profile •	
GENERALIZED DRAWING PRIMITIVE [v1] References: 7.6.10	Element is: Required • Permitted • Prohibited • List all registered GDPs that are allowed: List all profile-defined GDPs that are allowed and attach complete description: NOTE — Only registered GDPs and profile-defined GDPs shall be allowed in profiles. Other:	Element is: Required • Permitted • Prohibited ☑ List all the registered GDPs that are allowed: List all profile-defined GDPs that are allowed and attach complete description: Other:

Table 19 — Graphical primitive elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.19.11	Same as Model Profile •	
RECTANGLE [v1] References: 7.6.11 T.14.3 D.2.2.2	Element is: Required • Permitted • Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***None.***
T.19.12	Same as Model Profile •	
CIRCLE [v1] References: 7.6.12 T.14.3 D.2.2.2	Element is: Required • Permitted • Prohibited • Zero-area degeneracies shall be as defined in T.14.3. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***None.***
T.19.13	Same as Model Profile •	
CIRCULAR ARC 3 POINT [v1] References: 7.6.13 T.14.2 D.2.2.2 D.4.5.4	Element is: Required • Permitted • Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***

Table 19 — Graphical primitive elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.19.14	Same as Model Profile •	
CIRCULAR ARC 3 POINT CLOSE [v1] References: 7.6.14 T.14.3 D.2.2.2 D.4.5.5	Element is: Required • Permitted • Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***None.***
T.19.15	Same as Model Profile •	
CIRCULAR ARC CENTRE [v1] References: 7.6.15 T.14.2 D.2.2.2 D.4.5.6	Element is: Required • Permitted • Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.16	Same as Model Profile ☐	
CIRCULAR ARC CENTRE CLOSE [v1] References: 7.6.16 T.14.3 9.5.4.4 D.2.2.2 D.4.5.7	Element is: Required • Permitted • Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***None.***

Table 19 — Graphical primitive elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.19.17 ELLIPSE [v1] References: 7.6.17 D.2.2.2 D.4.5.9 D.4.5.10	Same as Model Profile • Element is: Required • Permitted • Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	 Element is: Required • Permitted ☑ Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other: **None.**
T.19.18 ELLIPTICAL ARC [v1] References: 7.6.18 T.14.2 D.2.2.1 D.4.5.11	Same as Model Profile • Element is: Required • Permitted • Prohibited • Zero-length geometric degeneracies shall be as defined in clause T.14.2. Other:	 Element is: Required • Permitted ☑ Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.19 ELLIPTICAL ARC CLOSE [v1] References: 7.6.19 T.14.3 D.2.2.2 D.4.5.12	Same as Model Profile • Element is: Required • Permitted • Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other:	 Element is: Required • Permitted ☑ Prohibited • Zero-area geometric degeneracies shall be as defined in T.14.3. Other: ***None.***

Table 19 — Graphical primitive elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.19.20	Same as Model Profile •	
CIRCULAR ARC CENTRE REVERSED [v2] References: 7.6.20 T.14.2 D.2.2.1 D.4.5.8	Element is: Required ☐ Permitted ☐ Prohibited ☐ Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.21	Same as Model Profile •	
CONNECTING EDGE [v2] References: 7.6.21 T.14.2 D.2.2.1	Element is: Required • Permitted • Prohibited • This element shall be permitted only if BEGIN/END FIGURE is permitted. Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • This element shall be permitted only if BEGIN/END FIGURE is permitted. Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.22	Same as Model Profile •	
HYPERBOLIC ARC [v3] References: 7.6.22 9.5.4.3 T.14.2 T.14.2	Element is: Required • Permitted • Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***

Table 19 — Graphical primitive elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.19.23	Same as Model Profile •	
PARABOLIC ARC [v3] References: 7.6.23 T.14.2 D.2.2.1	Element is: Required • Permitted • Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Zero-length geometric degeneracies shall be as defined in T.14.2 Other: ***None.***
T.19.24	Same as Model Profile •	
NON-UNIFORM B-SPLINE [v3] References: 7.6.24 T.14.2 D.2.2.1	Element is: Required • Permitted • Prohibited • Set of spline orders: Maximum number of control points: Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Set of spline orders: ***cubic spline (order=4).*** Maximum number of control points: ***4096.*** Zero-length geometric degeneracies shall be as defined in T.14.2. Other: The spline shall be clamped form, i.e., the first 4 knots shall be identical and the last 4 knots shall be identical**.**
T.19.25	Same as Model Profile •	
NON-UNIFORM RATIONAL B-SPLINE [v3] References: 7.6.25 T.14.2 D.2.2.1	Element is: Required • Permitted • Prohibited • Set of spline orders: Maximum number of control points: Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required • Permitted ☑ Prohibited • Set of spline orders: ***cubic spline (order=4).*** Maximum number of control points: ***4096.*** Zero-length geometric degeneracies shall be as defined in T.14.2. Other: The spline shall be clamped form, i.e., the first 4 knots shall be identical and the last 4 knots shall be identical**.**

Table 19 — Graphical primitive elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.19.26	Same as Model Profile ☐	
POLYBEZIER [v3] References: 7.6.26 T.14.2 D.2.2.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ Maximum number of points: Any restrictions on the continuity indicator? Zero-length geometric degeneracies shall be as defined in T.14.2. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Maximum number of points: ***4096.*** Any restrictions on the continuity indicator? ***None.*** Zero-length geometric degeneracies shall be as defined in T.14.2. Other: ***None.***
T.19.27	Same as Model Profile ☐	
POLYSYMBOL [v3] References: 7.6.27 D.2.2.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ Point list: Effect of a reference to a symbol index parameter which is not in the symbol library: Other:	Element is: Required ☐ Permitted ☐ Prohibited ☑ Point list: Effect of a reference to a symbol index parameter which is not in the symbol library. Other: ***NOTE — This element is prohibited because SYMBOL LIBRARY LIST is prohibited.***
T.19.28	Same as Model Profile ☐	
BITONAL TILE [v3] References: 7.6.28 D.2.2.1 D.4.5.13	Element is: Required ☐ Permitted ☐ Prohibited ☐ List allowable compression types: Requirements on row padding: Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ List allowable compression types: *Values* ***0..6.*** Requirements on row padding: ***None.*** Other: ***CCITT compression methods (T6 and T4) shall be used with 1 bit cell colour precision and indexed colour.*** NOTE — Several compression types have been registered (as of date of publication) in the ISO Register of Graphical Items, specifically: JPEG, LZW, and PNG.

Table 19 — Graphical primitive elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.19.29	Same as Model Profile ☐	
TILE [v3] References: 7.6.29 D.2.2.1 D.4.5.13	Element is: Required ☐ Permitted ☐ Prohibited ☐ List allowable compression types: Requirements on row padding: Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ List allowable compression types: *Values **0..6.*** Requirements on row padding? ***None.*** Other: ***CCITT compression methods (T6 and T4) shall be used with 1 bit cell colour precision and indexed colour.*** NOTE — Several compression types have been registered (as of date of publication) in the ISO Register of Graphical Items, specifically: JPEG, LZW, and PNG.

Table 20 — Attribute elements

Element	Specifications – PPF	Specifications - Model Profile
T.20.1	Same as Model Profile ☐	
LINE BUNDLE INDEX [v1] References: 7.7.1 9.5.4.2 D.4.6.1 T.17.11	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *line bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: For [v2], [v3], and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *line bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: ***1..5.*** ***Index*** ***1*** ***2*** ***3*** ***4*** ***5*** ***line type*** ***1*** ***2*** ***3*** ***4*** ***5*** ***line width*** ***1.0*** ***1.0*** ***1.0*** ***1.0*** ***1.0*** ***line colour*** ***1*** ***1*** ***1*** ***1*** ***1*** For [v2], [v3], and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other: ***None.***
T.20.2	Same as Model Profile ☐	
LINE TYPE [v1] References: 7.7.2 7.4.17 D.4.6.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ Select 1 or more of the following: ☐ values 1..5; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). For [v3] and [v4] metafiles, ☐ negative values assigned by the LINE AND EDGE TYPE DEFINITION element. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Select 1 or more of the following: ☑ values 1..5; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). For [v3] and [v4] metafiles, ☑ negative values assigned by the LINE AND EDGE TYPE DEFINITION element. Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.3	Same as Model Profile ☐	
LINE WIDTH [v1] References: 7.7.3 D.4.6.3	Element is: Required ☐ Permitted ☐ Prohibited ☐ Is value zero allowed? (yes/no) If yes, specify its meaning. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Is value zero allowed? (yes/no) ***Yes.*** If yes, specify its meaning. **Minimum available line width.** Any restrictions on the parameter value? ***None.*** Other: ***None.*** .
T.20.4	Same as Model Profile ☐	
LINE COLOUR [v1] References: 7.7.4 9.5.4.1 T.14.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *line colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *line colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.5	Same as Model Profile ☐	
MARKER BUNDLE INDEX [v1] References: 7.7.5 9.5.4.2 T.17.12 D.4.6.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *marker bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *marker bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: ***1..5.*** ***index*** ***1*** ***2*** ***3*** ***4*** ***5*** ***marker type*** ***1*** ***2*** ***3*** ***4*** ***5*** ***marker width*** ***1.0*** ***1.0*** ***1.0*** ***1.0*** ***1.0*** ***marker colour*** ***1*** ***1*** ***1*** ***1*** ***1*** For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.6	Same as Model Profile ☐	
MARKER TYPE [v1] References: 7.7.6 D.4.6.4	Element is: Required ☐ Permitted ☐ Prohibited ☐ Indicate one or more of the following restrictions: ☐ values 1-5; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Indicate one or more of the following restrictions: ☑ values 1..5; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). Other: ***None.***
T.20.7	Same as Model Profile ☐	
MARKER SIZE [v1] References: 7.7.7 D.4.6.5	Element is: Required ☐ Permitted ☐ Prohibited ☐ Is value zero allowed? (yes/no) If yes, specify its meaning. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Is value zero allowed? (yes/no) ***Yes.*** If yes, specify its meaning. ***Minimum available size.*** Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.8	Same as Model Profile ☐	
MARKER COLOUR [v1] References: 7.7.8 9.5.4.1 T.14.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *marker colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *marker colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.9	Same as Model Profile ☐	
TEXT BUNDLE INDEX [v1] References: 7.7.9 9.5.4.2 T.17.13 D.4.6.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The text bundle index parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The text bundle index parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values: *1..2.* ***index*** ***1*** ***2*** ***font index*** ***1*** ***1*** ***text precision*** ***stroke*** ***stroke*** ***character expansion factor*** ***1.0*** ***0.7*** ***character spacing*** ***0.0*** ***0.0*** ***text colour*** ***1*** ***1*** For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other: ***None.***
T.20.10	Same as Model Profile ☐	
TEXT FONT INDEX [v1] References: 7.7.10 9.5.4.2 T.16.13	Element is: Required ☐ Permitted ☐ Prohibited ☐ Every referenced index shall refer to an entry in the FONT LIST (see T.16.13) Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Every referenced index shall refer to an entry in the FONT LIST (see T.16.13). Other: ***None.***
T.20.11	Same as Model Profile ☐	
TEXT PRECISION [v1] References: 7.7.11	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.12	Same as Model Profile ☐	
CHARACTER EXPANSION FACTOR [v1] References: 7.7.12 D.4.6.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Is value zero allowed? (yes/no) If yes, state the meaning. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Is value zero allowed? (yes/no) ***No.*** If yes, state the meaning. Any restrictions on the parameter value? ***Values shall be restricted to the range 0.1..10.0*** Other: ***None.***
T.20.13	Same as Model Profile ☐	
CHARACTER SPACING [v1] References: 7.7.13 D.4.6.8	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***Values shall be restricted to the range of -1.0..5.0.*** Other: ***None.***
T.20.14	Same as Model Profile ☐	
TEXT COLOUR [v1] References: 7.7.14 9.5.4.1 T.14.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *text colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *text colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.15	Same as Model Profile ☐	
CHARACTER HEIGHT [v1] References: 7.7.15 D.4.6.9	Element is: Required ☐ Permitted ☐ Prohibited ☐ Is zero height allowed? (yes/no) If yes, state its meaning. Any restrictions on the parameter? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Is zero height allowed: (Yes/no) ***Yes.*** If yes, state its meaning: ***Minimum available height.*** Any restrictions on the parameter? ***None.*** Other: ***None.***
T.20.16	Same as Model Profile ☐	
CHARACTER ORIENTATION [v1] References: 7.7.16 D.4.6.10	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the following distortion aspects? rotation? skewing? mirroring? aspect ratio? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the following distortion aspects? rotation? ***None.*** skewing? ***None.*** mirroring? ***None.*** aspect ratio? ***None.*** Other: ***None.***
T.20.17	Same as Model Profile ☐	
TEXT PATH [v1] References: 7.7.17 D.4.6.11	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.18	Same as Model Profile ☐	
TEXT ALIGNMENT [v1] References: 7.7.18 D.4.6.12	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the horizontal and vertical alignment values? Any restrictions on the continuous horizontal and vertical alignment values? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the horizontal and vertical alignment values? ***None.*** Any restrictions on the continuous horizontal and vertical alignment values? ***None.*** Other: ***None.***
T.20.19	Same as Model Profile ☐	
CHARACTER SET INDEX [v1] References: 7.7.19 9.5.4.2 T.16.14 T.16.22 D.4.6.13	Element is: Required ☐ Permitted ☐ Prohibited ☐ Every referenced index shall refer to an entry in the CHARACTER SET LIST or GLYPH MAPPING. This includes implicit reference to the default index value. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Every referenced index shall refer to an entry in the CHARACTER SET LIST or GLYPH MAPPING. This includes implicit reference to the default index value. Other: ***None.***
T.20.20	Same as Model Profile ☐	
ALTERNATE CHARACTER SET INDEX [v1] References: 7.7.20 9.5.4.2 T.16.14 T.16.22 D.4.6.13	Element is: Required ☐ Permitted ☐ Prohibited ☐ Every referenced index shall refer to an entry in the CHARACTER SET LIST or GLYPH MAPPING. This includes implicit reference to the default index value. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Every referenced index shall refer to an entry in the CHARACTER SET LIST or GLYPH MAPPING. This includes implicit reference to the default index value. Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.21	Same as Model Profile ☐	
FILL BUNDLE INDEX [v1] References: 7.7.21 9.5.4.2 T.17.14 D.4.6.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *fill bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *fill bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: ***1..5.*** ***Index*** ***1*** ***2*** ***3*** ***4*** ***5*** ***interior style*** ***hatch*** ***hatch*** ***hatch*** ***hatch*** ***hatch*** ***fill colour*** ***1*** ***1*** ***1*** ***1*** ***1*** ***hatch index*** ***1*** ***2*** ***3*** ***4*** ***5*** ***pattern index*** ***1*** ***1*** ***1*** ***1*** ***1*** For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other: ***None.***
T.20.22	Same as Model Profile ☐	
INTERIOR STYLE [v1] References: 7.7.22 D.4.6.15	Element is: Required ☐ Permitted ☐ Prohibited ☐ For ‘hollow’ interior style, line type and width of the bounding line: Any restrictions on the parameter value?: Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ For ‘hollow’ interior style, line type and width of the bounding line: ***Solid line type and default line width.*** Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.23	Same as Model Profile ☐	
FILL COLOUR [v1] References: 7.7.23 9.5.4.1 T.14.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *fill colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *fill colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.24	Same as Model Profile ☐	
HATCH INDEX [v1] References: 7.4.18 7.7.24 5.7.4.2 D.4.6.16	Element is: Required ☐ Permitted ☐ Prohibited ☐ Select 1 or more of the following: ☐ values 1..6; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). For [v3] and [v4] metafiles, ☑ negative values assigned by the HATCH STYLE DEFINITION element. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Select 1 or more of the following: ☑ values 1..6; ☐ subset of registered values (attach list); ☐ profile defined values (attach complete description). For [v3] and [v4] metafiles, ☑ negative values assigned by the HATCH STYLE DEFINITION element. Other: ***None.***
T.20.25	Same as Model Profile ☐	
PATTERN INDEX [v1] References: 7.7.25 9.5.4.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *pattern index* parameter shall follow the rules for indexes, clause 9.5.4.2. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *pattern index* parameter shall follow the rules for indexes, clause 9.5.4.2. Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.26	Same as Model Profile ☐	
EDGE BUNDLE INDEX [v1] References: 7.7.26 9.5.4.2 T.17.15 D.4.6.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *edge bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *edge bundle index* parameter shall follow the rules for indexes, clause 9.5.4.2. For [v1] metafiles, allowable index values and meanings of values: ***1..5***. ***index*** ***1*** ***2*** ***3*** ***4*** ***5*** ***edge type*** ***1*** ***2*** ***3*** ***4*** ***5*** ***edge width*** ***1.0*** ***1.0*** ***1.0*** ***1.0*** ***1.0*** ***edge colour*** ***1*** ***1*** ***1*** ***1*** ***1*** For [v2], [v3] and [v4] metafiles, any referenced bundle shall have an explicit representation definition. Other: ***None.***
T.20.27	Same as Model Profile ☐	
EDGE TYPE [v1] References: 7.4.17 7.7.27 D.4.6.17	Element is: Required ☐ Permitted ☐ Prohibited ☐ Select 1 or more of the following: ☐ values 1..5; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). For [v3] and [v4] metafiles, negative values assigned by the LINE AND EDGE TYPE DEFINITION element. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Select 1 or more of the following: ☑ values 1..5; ☐ subset of registered values (attach list); ☐ profile-defined values (attach complete description). For [v3] and [v4] metafiles, ☑ negative values assigned by the LINE AND EDGE TYPE DEFINITION element. Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.28	Same as Model Profile ☐	
EDGE WIDTH [v1] References: 7.7.28 D.4.6.18	Element is: Required ☐ Permitted ☐ Prohibited ☐ Is value zero allowed? (Yes/no) If yes, specify its meaning. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Is value zero allowed? (Yes/no) ***Yes.*** If yes, specify its meaning. ***Minimum available edge width.*** Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.29	Same as Model Profile ☐	
EDGE COLOUR [v1] References: 7.7.29 9.5.4.1 T.14.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *edge colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ The *edge colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.30	Same as Model Profile ☐	
EDGE VISIBILITY [v1] References: 7.7.30	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.31	Same as Model Profile ☐	
FILL REFERENCE POINT [v1] References: 7.7.31	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.20.32	Same as Model Profile ☐	
PATTERN TABLE [v1] References: 7.7.32	Element is: Required ☐ Permitted ☐ Prohibited ☐ Maximum size for nx: Allowable values for nx: Maximum size for ny: Allowable values for ny: Any restrictions on the number of pattern definitions? Any restrictions on allowable combinations of nx and ny? Any restrictions on the number of colours? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Maximum size for nx: ***32.*** Allowable values for nx: ***8, 16, or 32.*** Maximum size for ny: ***32.*** Allowable values for ny: ***8, 16, or 32.*** Any restrictions on the number of pattern definitions? ***64.*** Any restrictions on allowable combinations of nx and ny? ***None.*** Any restrictions on the number of colours? ***None.*** Other: ***None.***
T.20.33	Same as Model Profile ☐	
PATTERN SIZE [v1] References: 7.7.33 D.4.6.19	Element is: Required ☐ Permitted ☐ Prohibited ☐ Must pattern vectors be parallel to coordinate axes? (Yes/no) If no, state the meaning of skewed or non-aligned patterns. Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Must pattern vectors be parallel to coordinate axes? (yes/no) ***Yes.*** If no, state the meaning of skewed or non-aligned patterns. Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.34	Same as Model Profile ☐	
COLOUR TABLE [v1] References: 7.7.34 9.5.4.1 T.14.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any limits on the length of colour list? Any restrictions on the index values? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any limits on the length of colour list? **Monochrome: 2, Greyscale: 64, Colour: 256.** Any restrictions on the index values? ***Index values shall not exceed the maximum colour index.*** Other: ***None.***
T.20.35	Same as Model Profile ☐	
ASPECT SOURCE FLAGS [v1] References: 7.7.35 D.4.6.20	Element is: Required ☐ Permitted ☐ Prohibited ☐ Are all ASF values to be the same: for the metafile? (yes/no) within each class (line, marker, text, fill, edge) of primitive? (yes/no) Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Are all ASF values to be the same: for the metafile? (yes/no) ***No.*** within each class (line, marker, text, fill, edge) of primitive? (yes/no) ***Yes.*** Other: ***None.***
T.20.36	Same as Model Profile ☐	
PICK IDENTIFIER [v2] References: 7.7.36	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.37	Same as Model Profile ☐	
LINE CAP [v3] References: 7.7.37 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values for the line cap indicator? (choose 1 or both) ☐ values 1..5; ☐ subset of registered values (attach list). Any restrictions on the set of values for the dash cap indicator? (choose 1 or both) ☐ values 1..3; ☐ subset of registered values (attach list). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values for the line cap indicator? (choose 1 or both) ☑ values 1..5; ☐ subset of registered values (attach list). Any restrictions on the set of values for the dash cap indicator? (choose 1 or both) ☑ values 1..3; ☐ subset of registered values (attach list). Other: ***None.***
T.20.38	Same as Model Profile ☐	
LINE JOIN [v3] References: 7.7.38 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☐ values 1...4; ☐ subset of registered values (attach list). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☑ values 1...4; ☐ subset of registered values (attach list). Other: ***None.***
T.20.39	Same as Model Profile ☐	
LINE TYPE CONTINUATION [v3] References: 7.7.39 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values? ***1..4.*** Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.20.40	Same as Model Profile ☐	
LINE TYPE INITIAL OFFSET [v3] References: 7.7.40	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.41	Same as Model Profile ☐	
TEXT SCORE TYPE [v3] References: 7.7.41	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☐ Values 1..4; ☐ Subset of registered values (attach list). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☑ Values 1..4; ☐ Subset of registered values (attach list). Other: ***None.***
T.20.42	Same as Model Profile ☐	
RESTRICTED TEXT TYPE [v3] References: 7.7.42 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☐ Values 1..6; ☐ Subset of registered values (attach list). Algorithms for achieving restriction type? (attach) Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☑ values 1..6; ☐ Subset of registered values (attach list). Algorithms for achieving restriction type? (attach) **Not specified**. Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.20.43	Same as Model Profile ☐	
INTERPOLATED INTERIOR [v3] References: 7.7.43	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any limits on the number of stages? Any restrictions on the set of values? (choose 1 or both) ☐ Values 1..3; ☐ Subset of registered values (attach list). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any limits on the number of stages? ***Maximum number of stages is 8.*** Any restrictions on the set of values? (choose 1 or both) ☑ Values 1..3; ☐ Subset of registered values (attach list). Other: ***None.***
T.20.44	Same as Model Profile ☐	
EDGE CAP [v3] References: 7.7.44 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values for the edge cap indicator? (choose 1 or both) ☐ values 1..5; ☐ subset of registered values (attach list). Any restrictions on the set of values for the dash cap indicator? (choose 1 or both) ☐ values 1..3; ☐ subset of registered values (attach list). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values for the edge cap indicator? (choose 1 or both) ☑ values 1..5; ☐ subset of registered values (attach list). Any restrictions on the set of values for the dash cap indicator? (choose 1 or both) ☑ values 1..3; ☐ subset of registered values (attach list). Other: ***None.***
T.20.45	Same as Model Profile ☐	
EDGE JOIN [v3] References: 7.7.45 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☐ values 1...4; ☐ subset of registered values (attach list). Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values? (choose 1 or both) ☑ values 1..4; ☐ subset of registered values (attach list). Other: ***None.***

Table 20— Attribute elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.20.46	Same as Model Profile ☐	
EDGE TYPE CONTINUATION [v3] References: 7.7.46 9.5.7.5 T.26.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the set of values? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the set of values? ***1..4.*** Other: ***None.***
T.20.47	Same as Model Profile ☐	
EDGE TYPE INITIAL OFFSET [v3] References: 7.7.47	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter value? ***None.*** Other: ***None.***
T.20.48	Same as Model Profile ☐	
SYMBOL LIBRARY INDEX [v3] References: 7.7.48 9.5.4.2 T.16.23	Element is: Required ☐ Permitted ☐ Prohibited ☐ Every referenced index shall refer to an entry in the SYMBOL LIBRARY LIST (see T.16.23). Other:	Element is: Required ☐ Permitted ☐ Prohibited ☑ Every referenced index shall refer to an entry in the SYMBOL LIBRARY LIST (see T.16.23). Other: ***This element is prohibited because SYMBOL LIBRARY LIST is prohibited.***

Table 20— Attribute elements (Continued)

Element	Specifications – PPF	Specifications - Model Profile
T.20.49	Same as Model Profile ☐	
SYMBOL COLOUR [v3] References: 7.7.49 9.5.4.1 T.14.1 T.16.23 D.4.6.21	Element is: Required ☐ Permitted ☐ Prohibited ☐ The *symbol colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☐ Prohibited ☑ The *symbol colour specifier* parameter shall follow the rules for colour, clause 9.5.4.1 and T.14.1. Any restrictions on the parameter value? Other: ***This element is prohibited because SYMBOL LIBRARY LIST is prohibited.***
T.20.50	Same as Model Profile ☐	
SYMBOL SIZE [v3] References: 7.7.50 T.16.23	Element is: Required ☐ Permitted ☐ Prohibited ☐ Is value zero allowed: (yes/no) If yes, specify its meaning. Any restrictions on the parameter value? Other:	Element is: Required ☐ Permitted ☐ Prohibited ☑ Is value zero allowed: (yes/no) If yes, specify its meaning. Any restrictions on the parameter value? Other: ***This element is prohibited because SYMBOL LIBRARY LIST is prohibited.***
T.20.51	Same as Model Profile ☐	
SYMBOL ORIENTATION [v3] References: 7.7.51 T.16.23 D.4.6	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on rotation? Any restrictions on skewing? Any restrictions on mirroring? Any restrictions on distortion of aspect ratio? Other:	Element is: Required ☐ Permitted ☐ Prohibited ☑ Any restrictions on rotation? Any restrictions on skewing? Any restrictions on mirroring? Any restrictions on distortion of aspect ratio? Other: ***This element is prohibited because SYMBOL LIBRARY LIST is prohibited.***

Table 21 — Escape elements

Element	Specifications – PPF	Specifications - Model Profile
T.21.1	Same as Model Profile ☐	
ESCAPE [v1] References: 7.8.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ List all registered ESCAPEs that are allowed: List all profile-defined ESCAPEs that are allowed and attach complete description: Other: NOTE — Only registered ESCAPEs and profile-defined ESCAPEs are allowed in profiles.	Element: Required ☐ Permitted ☑ Prohibited ☐ List all registered ESCAPEs that are allowed: ***ESCAPE 22, Transparent Cell Colour [v1/v2 metafiles only].*** List all profile-defined ESCAPEs that are allowed and attach complete description: ***None.*** Other: ***None.***

Table 22 — External elements

Element	Specifications – PPF	Specifications - Model Profile
T.22.1	Same as Model Profile ☐	
MESSAGE [v1] References: 7.9.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ Values of the *action required flag* parameter: 'action' Permitted ☐ Prohibited ☐ (if permitted, specify the messages and actions taken.) 'no action' Permitted ☐ Prohibited ☐ Any restrictions on the length of the message string, other than those for type SF parameter? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Values of the *action required flag* parameter: 'action' Permitted ☐ Prohibited ☑ (if permitted, specify the messages and actions taken.) 'no action' Permitted ☑ Prohibited ☐ Any restrictions on the length of the message string, other than those for SF parameter? ***None.*** Other: ***None.***
T.22.2	Same as Model Profile ☐	
APPLICATION DATA [v1] References: 7.9.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ Attach a syntactic and semantic description of all application data elements associated with this profile. Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ The use of this element shall not be restricted. Attach a syntactic and semantic description of all application data elements associated with this profile. Other: ***None.***

Table 23 — Segment elements

Element	Specifications – PPF	Specifications - Model Profile
T.23.1	Same as Model Profile ☐	
COPY SEGMENT [v2] References: 7.10.1 D.4.9.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ Every segment identifier shall refer to a defined segment. Any limits on the segment transformation application value? Any restrictions on the nature of the transformation (e.g., permitting only isotropic transformations)? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Every segment identifier shall refer to a defined segment. Any limits on the segment transformation application value? ***None.*** Any restrictions on the nature of the transformation (e.g., permitting only isotropic transformations)? ***Non-singular.*** Other: ***None.***
T.23.2	Same as Model Profile ☐	
INHERITANCE FILTER [v2] References: 7.10.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any limits on the filter selection list? Any limits on the selection setting? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Any limits on the filter selection list? ***None.*** An limits on the selection setting? ***None.*** Other: ***None.***
T.23.3	Same as Model Profile ☐	
CLIP INHERITANCE [v2] References: 7.10.3 D.4.9.2	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any limits on the parameter? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Any limits on the parameter? ***None.*** Other: ***None.***

Table 23 — Segment elements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.23.4	Same as Model Profile ☐	
SEGMENT TRANS-FORMATION [v2] References: 7.10.4	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the nature of the transformation (e.g., permitting only isotropic transformations)? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the nature of the transformation (e.g., permitting only isotropic transformations)? ***Non-singular.*** Other: ***None.***
T.23.5	Same as Model Profile ☐	
SEGMENT HIGHLIGHTING [v2] References: 7.10.5	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter values? Other:	Element: Required Permitted ☑ Prohibited Any restrictions on the parameter values? ***None.*** Other: ***None.***
T.23.6	Same as Model Profile ☐	
SEGMENT DISPLAY PRIORITY [v2] References: 7.10.6	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter values? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter values? ***None.*** Other: ***None.***
T.23.7	Same as Model Profile ☐	
SEGMENT PICK PRIORITY [v2] References: 7.10.7	Element is: Required ☐ Permitted ☐ Prohibited ☐ Any restrictions on the parameter values? Other:	Element: Required ☐ Permitted ☑ Prohibited ☐ Any restrictions on the parameter values? ***None.*** Other: ***None.***

Table 24 — Application structure descriptor elements

Element	Specifications – PPF	Specifications - Model Profile
T.24.1	Same as Model Profile ☐	
APPLICATION STRUCTURE ATTRIBUTE [V4] References: 6.9 6.13.5 7.9.2 7.11.1	Element is: Required ☐ Permitted ☐ Prohibited ☐ Define the set of structure attribute elements for use within application structures, and attach complete syntactic and semantic description: Other:	Element is: Required ☐ Permitted ☑ Prohibited ☐ Define the set of structure attribute elements for use within application structures, and attach complete syntactic and semantic description: ***None.*** Other: ***None.***

Table 25 — Generator implementation requirements

Functionality	Specifications - PPF	Specifications - Model Profile
T.25.1	Same as Model Profile ☐	
Colour requirements References: 9.5.4.1 9.5.6.2.1	Colour mapping is: Permitted ☐ Prohibited ☐ Reduction of the number of colours? Definition of mapping algorithms, metrics, and colour space? For [v1/2] metafiles, implicit colour calibration specifications? Other:	Colour mapping is: Permitted ☑ Prohibited ☐ Reduction of the number of colours? ***Not specified.*** ***NOTE — If mapping of application colours to metafile colour specifications is required, it is recommended that colour distance in the mapping be computed by the Euclidean metric in CIEXYZ space.*** Definition of mapping algorithms, metrics, and colour space? ***No specific colour mapping techniques or selection of metafile colour sets are defined.*** For [v1/2] metafiles, implicit colour calibration specifications? ***No specifications are defined.*** Other: ***None.***
T.25.2	Same as Model Profile	
Geometric accuracy and latitude References: 9.5.6.2.2	Accuracy and latitude for mapping application graphics to CGM graphical primitive elements:	Accuracy and latitude for mapping application graphics to CGM graphical primitive elements: Accuracy and latitude for mapping application graphics to CGM graphical primitive elements: ***Generators shall produce a metafile whose graphical primitive elements match the application graphical primitives accurately to within ±0.1% of relative position within the VDC Extent box or ±½ pixel of the intended size, whichever is greater. Generators shall produce geometric size aspects of the primitives (e.g., text size, line width, and edge width) to within 1% of the intended size or ±½ pixel of the intended size, whichever is greater.*** ***This requirement shall apply to all graphical primitive elements, unless superseded by specific element requirements in this clause.***

Table 25 — Generator implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.25.3	Same as Model Profile ☐	
Text accuracy and latitude References: 9.5.6.2.3	Is text accuracy and latitude addressed? (yes/no) If yes, specify.	Is text accuracy and latitude addressed? (yes/no) ***Yes.*** If yes, specify. ***Metafile text specifications shall match the text of the application picture to within ±1% of relative to the intended size or ±½ pixel of the intended size, whichever is greater, for the placement and overall extent of each text string.***
T.25.4	Same as Model Profile ☐	
Font substitution References: 9.5.6.2.4 I.2	Font substitution is: Permitted ☐ Prohibited ☐ Similarity of font visual characteristics? Font metrics? Individual glyph metrics? Other:	Font substitution is: Permitted ☑ Prohibited ☐ Similarity of font visual characteristics? ***Substituted fonts shall have similar visual characteristics (e.g., posture, weight, proportionate width).*** Font metrics? ***Specified in clause I.2***. Individual glyph metrics? ***Specified in clause I.2.*** Other: ***None.***
T.25.5	Same as Model Profile	
Preservation of primitives References: 9.5.6.3	Is preservation of graphical primitive elements addressed? (yes/no) If yes, specify allowable substitutions.	Is preservation of graphical primitive elements addressed? (yes/no) ***No.*** If yes, specify allowable substitutions.

Table 25 — Generator implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.25.6	Same as Model Profile ☐	
Semantic latitude References: 9.5.6.4	Drawing priority and mode: Clipping: Edge centreing: Meaning of predefined line types and edge types: Meaning of predefined hatch styles: Other:	Drawing priority and mode: Priority shall correspond to the metafile order (i.e., primitives occurring later in the file shall overlay primitives occurring earliest in the file). Mode shall be "replacement" mode. Clipping: ***Clipping shall be to the intersection of the clip rectangle, the VDC EXTENT, the device viewport, and the device view surface limits.*** Edge centreing: ***Edges shall be centred on the ideal mathematically-defined edge of the area.*** Meaning of predefined line types and edge types: ***The exact on-off definitions for the predefined line types and edge types are not specified.*** Meaning of predefined hatch styles: The inter-line spacing is not specified. Use the latitudes of annex D.4.6.16 for the angular directions. Other: ***None.***
T.25.7	Same as Model Profile ☐	
Error processing References: 9.5.6.5	Is error processing addressed? (yes/no) If yes, specify the action taken. Classification of error severity? Requirements for error recovery? Requirements for error reporting? Additional areas? Other:	Is error processing addressed? (yes/no) ***No.*** If yes, specify the action taken. Classification of error severity? Requirements for error recovery? Requirements for error reporting? Additional areas? Other: ***None.***

Table 25 — Generator implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.25.8	Same as Model Profile ☐	
Reporting References: 9.5.6.6	Is reporting required? (yes/no) If yes, specify the action taken. Method and format of the reporting? Requirement to report substitution, error, fallback behaviour, mappings, or other behaviours? Additional areas? Other:	Is reporting required? (yes/no) ***No.*** If yes, specify the action taken. Method and format of the reporting? Requirement to report substitution, error, fallback behaviour, mappings, or other behaviours? Additional areas? Other: ***None.***
T.25.9	Same as Model Profile ☐	
Degeneracies References: 9.5.6.7 9.5.4.4 D.2 D.4	Is the generation of degenerate primitives addressed? (yes/no) If yes, attach specifications. Other:	Is the generation of degenerate primitives addressed? (yes/no) ***No. The generation of degenerate primitives is not restricted.*** If yes, attach specifications. Other: ***None.***

Table 26 — Interpreter implementation requirements

Functionality	Specifications - PPF	Specifications - Model Profile
T.26.1	Same as Model Profile ☐	
Number of pictures References: 9.5.7.2 T.13.2	If 0 pictures are permitted (see T.13.2), describe the interpreter behaviour:	If 0 pictures are permitted (see T.13.2), describe the interpreter behaviour: ***Prohibited by T.13.2.***
T.26.2	Same as Model Profile •	
Empty pictures References: 9.5.7.3 T.13.3	If permitted (see T.13.3), interpreter behavior:	If permitted (see T.13.3), interpreter behavior: The graphical effect shall be one picture in the background colour.
T.26.3	Same as Model Profile ☐	
Colour requirements References: 9.5.4.1 9.5.4.5 9.5.7.4.2	Interpreters shall be classified as either monochrome, greyscale, or colour interpreters (depending on the colour capability of the interpreter), and shall meet the criteria in attachment 26.3 ***Conversions between different colour models shall be according to the conversions in annex G.*** Mapping of metafile colour to device components? For [v1/2] metafiles, implicit colour calibration specifications? Other:	Interpreters shall be classified as either monochrome, greyscale, or colour interpreters (depending on the colour capability of the interpreter), and shall meet the criteria in attachment 26.3 ***Conversions between different colour models shall be according to the conversions in annex G.*** Mapping of metafile colour to device components? ***If mapping (to fewer colour, or greyscale, or monochrome) is required for RGB metafiles, the recommendations of annex D.3.2 shall be used.*** For [v1/2] metafiles, implicit colour calibration specifications? ***No* specifications are defined.** Other: ***None.***

Table 26 — Interpreter implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.26.4	Same as Model Profile ☐	
Geometric accuracy and latitude References: 9.5.7.4.2	Accuracy and latitude for placement and realization of geometric aspects when geometric primitive elements are rendered.	Accuracy and latitude for placement and realization of geometric aspects when geometric primitive elements are rendered. Interpreters shall render graphical primitive elements accurately to within ± 0.1% of relative position within the VDC Extent box or ±½ of the pixel resolution of the output device, whichever is greater. Interpreters shall render the geometric size aspect of primitives (e.g., text size, line width, and edge width) to within 1% of the intended size or ±½ pixel of resolution of the output device, whichever is greater. This requirement shall apply to all graphical primitive elements, unless superseded by specific element requirements in this clause.
T.26.5	Same as Model Profile ☐	
Text rendering References: 9.5.7.4.3	Is text accuracy and latitude addressed? (yes/no) If yes, specify. Is precision of text rendering is addressed? (yes/no) If yes, specify interpreter action.	Is text accuracy and latitude addressed? (yes/no) ***Yes.*** If yes, specify. ***Interpreter-rendered text shall match the text specification of the metafile to within 1% relative to the intended size or ±½ pixel of resolution of the output device, whichever is greater, for the placement and overall extent of each text string.*** Is precision of text rendering is addressed? (yes/no) ***Yes.*** If yes, specify interpreter action. ***Interpreters shall render text using 'stroke' precision, regardless of the actual value of the TEXT PRECISION of the metafile.***

Table 26 — Interpreter implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.26.6	Same as Model Profile ☐	
Font substitution References: T.16.13 9.5.7.4.4 annex I.2	Font substitution is: Permitted ☐ Prohibited ☐ If prohibited, use the font as specified in the FONT LIST. If permitted, include a reference set of font and glyph metrics which correspond to the canonical instances of the substitutable font. Are substitution methods, latitudes, constraints addressed? (yes/no) If yes, specify: Similarity of font visual characteristics? Font metrics? Individual glyph metrics? Additional areas? Other:	Font substitution is: Permitted ☑ Prohibited ☐ If prohibited, use the font as specified in the FONT LIST. If permitted, include a reference set of font and glyph metrics which correspond to the canonical instances of the substitutable font. ***See the FONT LIST element and I.2.*** Are substitution methods, latitudes, constraints addressed? (yes/no) ***Yes***. If yes, specify: Similarity of font visual characteristics? ***Substituted fonts shall have similar visual characteristics to the fonts specified in the metafile.*** Font metrics? ***Substituted fonts shall have similar metrics to the fonts specified in the metafile.*** Individual glyph metrics? ***As specified in annex I.2.*** Additional areas? ***None.*** Other: ***None.***
T.26.7	Same as Model Profile ☐	
Semantic latitude References: 9.5.7.5 T.20.37 T.20.38 T.20.39 T.20.42 T.20.44 T.20.45 T.20.46	Drawing priority and mode: View surface clearing at picture start: Clipping:	Drawing priority and mode: ***Priority shall correspond to the metafile order (i.e., primitives occurring later in the file shall overlay primitives occurring earliest in the file. Mode shall be "replacement" mode.*** View surface clearing at picture start: ***Surface will be cleared upon the occurrence of BEGIN PICTURE BODY.*** Clipping: ***When CLIP INDICATOR is 'off', clipping shall be to the intersection of the device viewport and the device view surface limits. When CLIP INDICATOR is 'on', clipping shall be to the intersection of the clip rectangle, the VDC EXTENT, the device viewport, and the device view surface limits.***

Table 26 — Interpreter implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.26.7 continued Semantic latitude	Edge centreing: Meaning of predefined line types and edge types: Meaning of predefined hatch styles: In the absence of a LINE/TEXT/MARKER CLIPPING MODE element, the interpreter treatment of LINE/MARKER/TEXT CLIPPING MODE shall be (check one): ☐ in the style of one specific parameter value, from the set of standardized values. Specify which one: ☐ In the style of any of the specific parameter values, from the set of standardized values. For [v1/v2] metafiles, text restriction method for RESTRICTED TEXT elements, chosen from the set of standard and registered styles of the RESTRICTED TEXT TYPE element: For [v1/2] metafiles, interpreter treatment of the 2 aspects of line cap shall be either: ☐ in the style of one specific parameter value pair from the set of standard and registered values (excluding values 1) of the LINE CAP element. Values = ? ☐ in the style of any parameter value pair from the set of standard and registered values (excluding values 1) of the LINE CAP element. For [v1/2] metafiles, interpreter treatment of the 2 aspects of edge cap shall be either: ☐ in the style of one specific parameter value pair, from the set of standard and registered values (excluding values 1) of the EDGE CAP element. Values = ? ☐ in the style of any parameter value pair, from the set of standard and registered values (excluding values 1) of the EDGE CAP element. For [v1/2] metafiles, interpreter treatment of line join shall be either: ☐ in the style of one specific parameter value, from the set of	Edge centering: ***Edges shall be centred on the ideal mathematically-defined edge of the area.*** Meaning of predefined line types and edge types: ***The exact on-off definitions for the predefined line types and edge types are not specified.*** Meaning of predefined hatch styles: ***The inter-line spacing is not specified. Use the latitudes of annex D.4.6.16 for the angular directions***. In the absence of a LINE/TEXT/MARKER CLIPPING MODE element, the interpreter treatment of LINE/MARKER/TEXT CLIPPING MODE shall be (check one): ☐ in the style of one specific parameter value, from the set of standardized values. Specify which one: ☑ In the style of any of the specific parameter values, from the set of standardized values. For [v1/v2] metafiles, text restriction method for RESTRICTED TEXT elements, chosen from the set of standard and registered styles of the RESTRICTED TEXT TYPE element: ***Value 2***. For [v1/2] metafiles, interpreter treatment of the 2 aspects of line cap shall be either: ☐ in the style of one specific parameter value pair from the set of standard and registered values (excluding values 1) of the LINE CAP element. Values = ? ☑ in the style of any parameter value pair from the set of standard and registered values (excluding values 1) of the LINE CAP element. For [v1/2] metafiles, interpreter treatment of the 2 aspects of edge cap shall be either: ☐ in the style of one specific parameter value pair, from the set of standard and registered values (excluding values 1) of the EDGE CAP element. Values = ? ☑ in the style of any parameter value pair, from the set of standard and registered values (excluding values 1) of the EDGE CAP element. For [v1/2] metafiles, interpreter treatment of line join shall be either: ☐ in the style of one specific parameter value, from the set of

Table 26 — Interpreter implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.26.7 continued Semantic latitude	standard and registered values (excluding value 1) of the LINE JOIN element. Value = ? ☐ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the LINE JOIN element. For [v1/2] metafiles, interpreter treatment of edge join shall be either: ☐ in the style of one specific parameter value, from the set of standard and registered values (excluding value 1) of the EDGE JOIN element. Value = ? ☐ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the EDGE JOIN element For [v1/2] metafiles, interpreter treatment of line type continuation shall be either: ☐ in the style of one specific parameter value, from the set of standard and registered values (excluding value 1) of the LINE TYPE CONTINUATION element. Value = ? ☐ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the LINE TYPE CONTINUATION element. For [v1/2] metafiles, interpreter treatment of edge type continuation shall be either: ☐ in the style of one specific parameter value, from the set of standard and registered values (excluding value 1) of the EDGE TYPE CONTINUATION element. Value = ? ☐ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the EDGE TYPE CONTINUATION element. Other:	standard and registered values (excluding value 1) of the LINE JOIN element. Value = ? ☑ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the LINE JOIN element. For [v1/2] metafiles, interpreter treatment of edge join shall be either: ☐ in the style of one specific parameter value, from the set of standard and registered values (excluding value 1) of the EDGE JOIN element. Value = ? ☑ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the EDGE JOIN element. For [v1/2] metafiles, interpreter treatment of line type continuation shall be either ☐ in the style of one specific parameter value, from the set of standard and registered values (excluding value 1) of the LINE TYPE CONTINUATION element. Value = ? ☑ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the LINE TYPE CONTINUATION element. For [v1/2] metafiles, interpreter treatment of edge type continuation shall be either: ☐ in the style of one specific parameter value, from the set of standard and registered values (excluding value 1) of the EDGE TYPE CONTINUATION element. Value = ? ☑ in the style of any parameter value, from the set of standard and registered values (excluding value 1) of the EDGE TYPE CONTINUATION element. Other: ***None.***

Table 26 — Interpreter implementation requirements (Continued)

Element	Specifications - PPF	Specifications - Model Profile
T.26.8	Same as Model Profile ☐	
Error processing References: 9.5.7.6	Is error processing addressed? (yes/no) If yes, specify the action taken. Classification of error severity? Requirements for error recovery? Requirements for error reporting? Additional areas? Other:	Is error processing addressed? (yes/no) ***No.*** If yes, specify the action taken. Classification of error severity? Requirements for error recovery? Requirements for error reporting? Additional areas? Other: ***None.***
T.26.9	Same as Model Profile ☐	
Reporting References: 9.5.7.7	Is reporting required? (yes/no) If yes, specify the action taken. Method and format of the reporting? Requirement to report any substitution, error, fallback behaviour, mappings, or other behaviors? Additional areas? Other:	Is reporting required? (yes/no) ***No.*** If yes, specify the action taken. Method and format of the reporting? Requirement to report any substitution, error, fallback behaviour, mappings, or other behaviors? Additional areas? Other: ***None.***
T.26.10	**Same as Model Profile** ☐	
Degeneracies References: 9.5.7.8 9.5.4.4 D.2 D.4	Is the interpretation of degenerate primitives addressed? (yes/no) If yes, for each primitive, specify the degeneracy including its source (i.e., intrinsic or computational). Other:	Is the interpretation of degenerate primitives addressed? (yes/no) ***Yes.*** If yes, for each primitive, specify the degeneracy including its source (i.e., intrinsic or computational). ***Intrinsically degenerate primitives shall be rendered as specified in annex D subsections: D.2.2, D.2.3, D.4.5.4 through D.4.5.8, D.4.5.11, and D.4.5.12. Interpreters are not required to detect computational degeneracy. If interpreters do detect computational degeneracies, they shall be rendered as specified in annex D subsections: D.2.2, D.2.3, D.4.5.4 through D.4.5.8, D.4.5.11, and D.4.5.12*** Other: ***None.***
T.26.11	**Same as Model Profile** ☐	
Transparency References: 7.5.3	If Transparency permitted specify interpreter behavior:	If Transparency permitted specify interpreter behavior: Interpreters shall implement the AUXILIARY COLOUR and TRANSPARENCY elements as described in the 2nd and 3rd paragraphs of the description in 7.5.4.

7.5.4 T.18.4		
T.26.12	**Same as Model Profile** ☐	
INTERPRETATION OF STRUCTURES AND DIRECTORIES [v4] References: 7.2.18 7.3.24 7.4.20 9.5.7.10	Any requirements on the interpretation of the application structures ? Is application meaning associated with application structures ? yes/no If yes, specify the interpreter action or actions for each type of structure. Other:	Any requirements on the interpretation of the application structures ? ***Interpreters shall produce the correct graphical results.*** Is application meaning associated with application structures ? yes/no ***No***. If yes, specify the interpreter action or actions for each type of structure. Other: ***None.***

Table 26 — Interpreter implementation requirements (Continued)

Attachment 26.3		
Colour requirements, Model Profile:		
The colour mapping step (CMS) and colour rendering step (CRS) for each class of interpreters is as follows:		
monochrome:	CMS	all foreground information is mapped to one colour, background information to another colour.
	CRS	all foreground information is mapped to one colour, background information to another colour.
greyscale:	CMS	32 grey levels, the recommendations of annex D.3.2 is used to map colour to grey.
	CRS	a unique representation of each of the levels of grey.
full colour:	CMS	5R,9G,5B grid of RGB colour cube, plus a 32 grey levels (0-1), some of which are already on the grid.
	CRS	a unique representation of the 254 (255) "colours".

I.2 Font character codes and metrics

I.2.1 Introduction

This clause defines the association of numeric character code value to glyph for the fonts in the Model Profile FONT LIST. Details regarding font substitution by generators and/or interpreters can be found in the implementation requirements (see 9.5.6 and 9.5.7).

Clause I.2.3 presents the font metrics for the fonts in the Model Profile FONT LIST. Each glyph is defined on a notional 1000x1000 grid. For each font the CapHeight is specified and for each glyph the width is specified.

These fonts use a different method to specify character size to that used within CGM. Within the font definition the character size corresponds to the height of the notional grid, 1000. This includes some white space above the actual glyph. Within CGM the character height corresponds to the distance between the baseline and the capline, the CapHeight. Thus, if a character height of 10 units in VDC is specified, then this size corresponds to the CapHeight of the selected font, not to the grid size of 1000. The width of a glyph in VDC units is then:

character height * character expansion factor * (width/CapHeight).

I.2.2 Association of character code to glyph

The ISO Latin1 Encoding character codes to glyphs (see I.2.2.1) shall be used for each of the fonts except Symbol, which shall use the Symbol character codes to glyph (see I.2.2.2)

NOTE The character names ("Char Name") in I.2.2.1 and I.2.2.2 are abbreviated names. The full name of these characters are in ISO/IEC 10646-1:1993, tables 1 and 2.

I.2.2.1 ISO Latin 1 Encoding

ISOLatin1 Encoding

Char Number	Standard Char	Char Name	Char Number	Standard Char	Char Name	Char Number	Standard Char	Char Name	Char Number	Standard Char	Char Name
0			64	@	at	128			192	À	Agrave
1			65	A	A	129			193	Á	Aacute
2			66	B	B	130			194	Â	Acircumflex
3			67	C	C	131			195	Ã	Atilde
4			68	D	D	132			196	Ä	Adieresis
5			69	E	E	133			197	Å	Aring
6			70	F	F	134			198	Æ	AE
7			71	G	G	135			199	Ç	Ccedilla
8			72	H	H	136			200	È	Egrave
9			73	I	I	137			201	É	Eacute
10			74	J	J	138			202	Ê	Ecircumflex
11			75	K	K	139			203	Ë	Edieresis
12			76	L	L	140			204	Ì	Igrave
13			77	M	M	141			205	Í	Iacute
14			78	N	N	142			206	Î	Icircumflex
15			79	O	O	143			207	Ï	Idieresis
16			80	P	P	144			208	Ð	Eth
17			81	Q	Q	145			209	Ñ	Ntilde
18			82	R	R	146			210	Ò	Ograve
19			83	S	S	147			211	Ó	Oacute
20			84	T	T	148			212	Ô	Ocircumflex
21			85	U	U	149			213	Õ	Otilde
22			86	V	V	150			214	Ö	Odieresis
23			87	W	W	151			215	×	multiply
24			88	X	X	152			216	Ø	Oslash
25			89	Y	Y	153			217	Ù	Ugrave
26			90	Z	Z	154			218	Ú	Uacute
27			91	[	bracketleft	155			219	Û	Ucircumflex
28			92	\	backslash	156			220	Ü	Udieresis
29			93	]	bracketright	157			221	Ý	Yacute
30			94	^	asciicircum	158			222	Þ	Thorn
31			95	_	underscore	159			223	ß	germandbls
32		space	96	`	grave	160		space	224	à	agrave
33	!	exclam	97	a	a	161	¡	exclamdown	225	á	aacute
34	"	quotedbl	98	b	b	162	¢	cent	226	â	acircumflex
35	#	numbersign	99	c	c	163	£	sterling	227	ã	atilde
36	$	dollar	100	d	d	164	¤	currency	228	ä	adieresis
37	%	percent	101	e	e	165	¥	yen	229	å	aring
38	&	ampersand	102	f	f	166	¦	brokenbar	230	æ	ae
39	'	apostrophe	103	g	g	167	§	section	231	ç	ccedilla
40	(	parenleft	104	h	h	168	¨	dieresis	232	è	egrave
41	)	parenright	105	i	i	169	©	copyright	233	é	eacute
42	*	asterisk	106	j	j	170	ª	ordfeminine	234	ê	ecircumflex
43	+	plus	107	k	k	171	«	guillemotleft	235	ë	edieresis
44	,	comma	108	l	l	172	¬	logicalnot	236	ì	igrave
45	-	minus	109	m	m	173	-	hyphen	237	í	iacute
46	.	period	110	n	n	174	®	registered	238	î	icircumflex
47	/	slash	111	o	o	175	¯	macron	239	ï	idieresis
48	0	zero	112	p	p	176	°	degree	240	ð	eth
49	1	one	113	q	q	177	±	plusminus	241	ñ	ntilde
50	2	two	114	r	r	178	²	twosuperior	242	ò	ograve
51	3	three	115	s	s	179	³	threesuperior	243	ó	oacute
52	4	four	116	t	t	180	´	acute	244	ô	ocircumflex
53	5	five	117	u	u	181	µ	mu	245	õ	otilde
54	6	six	118	v	v	182	¶	paragraph	246	ö	odieresis
55	7	seven	119	w	w	183	·	periodcentered	247	÷	divide
56	8	eight	120	x	x	184	¸	cedilla	248	ø	oslash
57	9	nine	121	y	y	185	¹	onesuperior	249	ù	ugrave
58	:	colon	122	z	z	186	º	ordmasculine	250	ú	uacute
59	;	semicolon	123	{	braceleft	187	»	guillemotright	251	û	ucircumflex
60	<	less	124	|	bar	188	¼	onequarter	252	ü	udieresis
61	=	equal	125	}	braceright	189	½	onehalf	253	ý	yacute
62	>	greater	126	~	asciitilde	190	¾	threequarters	254	þ	thorn
63	?	question	127			191	¿	questiondown	255	ÿ	ydieresis

I.2.2.2 Symbol Encoding

Symbol Encoding

Char Number	Standard Char	Char Name	Char Number	Standard Char	Char Name	Char Number	Standard Char	Char Name	Char Number	Standard Char	Char Name
0			64	≅	congruent	128			192	ℵ	aleph
1			65	Α	Alpha	129			193	ℑ	Ifraktur
2			66	Β	Beta	130			194	ℜ	Rfraktur
3			67	Χ	Chi	131			195	℘	weierstrass
4			68	Δ	Delta	132			196	⊗	circlemultiply
5			69	Ε	Epsilon	133			197	⊕	circleplus
6			70	Φ	Phi	134			198	∅	emptyset
7			71	Γ	Gamma	135			199	∩	intersection
8			72	Η	Eta	136			200	∪	union
9			73	Ι	Iota	137			201	⊃	propersuperset
10			74	ϑ	theta1	138			202	⊇	reflexsuperset
11			75	Κ	Kappa	139			203	⊄	notsubset
12			76	Λ	Lambda	140			204	⊂	propersubset
13			77	Μ	Mu	141			205	⊆	reflexsubset
14			78	Ν	Nu	142			206	∈	element
15			79	Ο	Omicron	143			207	∉	notelement
16			80	Π	Pi	144			208	∠	angle
17			81	Θ	Theta	145			209	∇	gradient
18			82	Ρ	Rho	146			210	®	registerserif
19			83	Σ	Sigma	147			211	©	copyrightserif
20			84	Τ	Tau	148			212	™	trademarkserif
21			85	Υ	Upsilon	149			213	∏	product
22			86	ς	sigma1	150			214	√	radical
23			87	Ω	Omega	151			215	⋅	dotmath
24			88	Ξ	Xi	152			216	¬	logicalnot
25			89	Ψ	Psi	153			217	∧	logicaland
26			90	Ζ	Zeta	154			218	∨	logicalor
27			91	[	bracketleft	155			219	⇔	arrowdblboth
28			92	∴	therefore	156			220	⇐	arrowdblleft
29			93	]	bracketright	157			221	⇑	arrowdblup
30			94	⊥	perpendicular	158			222	⇒	arrowdblright
31			95	_	underscore	159			223	⇓	arrowdbldown
32		space	96		radicalex	160			224	◊	lozenge
33	!	exclam	97	α	alpha	161	ϒ	Upsilon1	225	〈	angleleft
34	∀	universal	98	β	beta	162	′	minute	226	®	registersans
35	#	numbersign	99	χ	chi	163	≤	lessequal	227	©	copyrightsans
36	∃	existential	100	δ	delta	164	⁄	fraction	228	™	trademarksans
37	%	percent	101	ε	epsilon	165	∞	infinity	229	∑	summation
38	&	ampersand	102	ϕ	phi	166	ƒ	florin	230	⎛	parenlefttp
39	∍	suchthat	103	γ	gamma	167	♣	club	231	⎜	parenleftex
40	(	parenleft	104	η	eta	168	♦	diamond	232	⎝	parenleftbt
41	)	parenright	105	ι	iota	169	♥	heart	233	⎡	bracketlefttp
42	∗	asteriskmath	106	φ	phi1	170	♠	spade	234	⎢	bracketleftex
43	+	plus	107	κ	kappa	171	↔	arrowboth	235	⎣	bracketleftbt
44	,	comma	108	λ	lambda	172	←	arrowleft	236	⎧	bracelefttp
45	−	minus	109	μ	mu	173	↑	arrowup	237	⎨	braceleftmid
46	.	period	110	ν	nu	174	→	arrowright	238	⎩	braceleftbt
47	/	slash	111	ο	omicron	175	↓	arrowdown	239	⎪	braceex
48	0	zero	112	π	pi	176	°	degree	240		
49	1	one	113	θ	theta	177	±	plusminus	241	〉	angleright
50	2	two	114	ρ	rho	178	″	second	242	∫	integral
51	3	three	115	σ	sigma	179	≥	greaterequal	243	⌠	integraltp
52	4	four	116	τ	tau	180	×	multiply	244	⎮	integralex
53	5	five	117	υ	upsilon	181	∝	proportional	245	⌡	integralbt
54	6	six	118	ϖ	omega1	182	∂	partialdiff	246	⎞	parenrighttp
55	7	seven	119	ω	omega	183	•	bullet	247	⎟	parenrightex
56	8	eight	120	ξ	xi	184	÷	divide	248	⎠	parenrightbt
57	9	nine	121	ψ	psi	185	≠	notequal	249	⎤	bracketrighttp
58	:	colon	122	ζ	zeta	186	≡	equivalence	250	⎥	bracketrightex
59	;	semicolon	123	{	braceleft	187	≈	approxequal	251	⎦	bracketrightbt
60	<	less	124	|	bar	188	…	ellipsis	252	⎫	bracerighttp
61	=	equal	125	}	braceright	189	⏐	arrowvertex	253	⎬	bracerightmid
62	>	greater	126	∼	similar	190	⎯	arrowhorizex	254	⎭	bracerightbt
63		question	127			191	↵	carriagereturn	255		

I.2.3 Font metric tables

I.2.3.1 Times-Roman

Times-Roman

CapHeight = 662

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	921	128			192	À	722
1			65	A	722	129			193	Á	722
2			66	B	667	130			194	Â	722
3			67	C	667	131			195	Ã	722
4			68	D	722	132			196	Ä	722
5			69	E	611	133			197	Å	722
6			70	F	556	134			198	Æ	889
7			71	G	722	135			199	Ç	667
8			72	H	722	136			200	È	611
9			73	I	333	137			201	É	611
10			74	J	389	138			202	Ê	611
11			75	K	722	139			203	Ë	611
12			76	L	611	140			204	Ì	333
13			77	M	889	141			205	Í	333
14			78	N	722	142			206	Î	333
15			79	O	722	143			207	Ï	333
16			80	P	556	144			208	Ð	722
17			81	Q	722	145			209	Ñ	722
18			82	R	667	146			210	Ò	722
19			83	S	556	147			211	Ó	722
20			84	T	611	148			212	Ô	722
21			85	U	722	149			213	Õ	722
22			86	V	722	150			214	Ö	722
23			87	W	944	151			215	×	564
24			88	X	722	152			216	Ø	722
25			89	Y	722	153			217	Ù	722
26			90	Z	611	154			218	Ú	722
27			91	[	333	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	333	157			221	Ý	722
30			94	^	469	158			222	Þ	556
31			95	_	500	159			223	ß	500
32		250	96	`	333	160		250	224	à	444
33	!	333	97	a	444	161	¡	333	225	á	444
34	"	408	98	b	500	162	¢	500	226	â	444
35	#	500	99	c	444	163	£	500	227	ã	444
36	$	500	100	d	500	164	¤	500	228	ä	444
37	%	833	101	e	444	165	¥	500	229	å	444
38	&	778	102	f	333	166	¦	200	230	æ	667
39	'	333	103	g	500	167	§	500	231	ç	444
40	(	333	104	h	500	168	¨	333	232	è	444
41	)	333	105	i	278	169	©	760	233	é	444
42	*	500	106	j	278	170	ª	276	234	ê	444
43	+	564	107	k	500	171	«	500	235	ë	444
44	,	250	108	l	278	172	¬	564	236	ì	278
45	-	564	109	m	778	173	-	333	237	í	278
46	.	250	110	n	500	174	®	760	238	î	278
47	/	278	111	o	500	175	¯	333	239	ï	278
48	0	500	112	p	500	176	°	400	240	ð	500
49	1	500	113	q	500	177	±	564	241	ñ	500
50	2	500	114	r	333	178	²	300	242	ò	500
51	3	500	115	s	389	179	³	300	243	ó	500
52	4	500	116	t	278	180	´	333	244	ô	500
53	5	500	117	u	500	181	µ	500	245	õ	500
54	6	500	118	v	500	182	¶	453	246	ö	500
55	7	500	119	w	722	183	·	250	247	÷	564
56	8	500	120	x	500	184	¸	333	248	ø	500
57	9	500	121	y	500	185	¹	300	249	ù	500
58	:	278	122	z	444	186	º	310	250	ú	500
59	;	278	123	{	480	187	»	500	251	û	500
60	<	564	124	|	200	188	¼	750	252	ü	500
61	=	564	125	}	480	189	½	750	253	ý	500
62	>	564	126	~	541	190	¾	750	254	þ	500
63	?	444	127			191	¿	444	255	ÿ	500

I.2.3.2 Times-Bold

Times-Bold

CapHeight = 676

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	930	128			192	À	722
1			65	A	722	129			193	Á	722
2			66	B	667	130			194	Â	722
3			67	C	722	131			195	Ã	722
4			68	D	722	132			196	Ä	722
5			69	E	667	133			197	Å	722
6			70	F	611	134			198	Æ	1000
7			71	G	778	135			199	Ç	722
8			72	H	778	136			200	È	667
9			73	I	389	137			201	É	667
10			74	J	500	138			202	Ê	667
11			75	K	778	139			203	Ë	667
12			76	L	667	140			204	Ì	389
13			77	M	944	141			205	Í	389
14			78	N	722	142			206	Î	389
15			79	O	778	143			207	Ï	389
16			80	P	611	144			208	Ð	722
17			81	Q	778	145			209	Ñ	722
18			82	R	722	146			210	Ò	778
19			83	S	556	147			211	Ó	778
20			84	T	667	148			212	Ô	778
21			85	U	722	149			213	Õ	778
22			86	V	722	150			214	Ö	778
23			87	W	1000	151			215	×	570
24			88	X	722	152			216	Ø	778
25			89	Y	722	153			217	Ù	722
26			90	Z	667	154			218	Ú	722
27			91	[	333	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	333	157			221	Ý	722
30			94	^	581	158			222	Þ	611
31			95	_	500	159			223	ß	556
32		250	96	`	333	160		250	224	à	500
33	!	333	97	a	500	161	¡	333	225	á	500
34	"	555	98	b	556	162	¢	500	226	â	500
35	#	500	99	c	444	163	£	500	227	ã	500
36	$	500	100	d	556	164	¤	500	228	ä	500
37	%	1000	101	e	444	165	¥	500	229	å	500
38	&	833	102	f	333	166	¦	220	230	æ	722
39	'	333	103	g	500	167	§	500	231	ç	444
40	(	333	104	h	556	168	¨	333	232	è	444
41	)	333	105	i	278	169	©	747	233	é	444
42	*	500	106	j	333	170	ª	300	234	ê	444
43	+	570	107	k	556	171	«	500	235	ë	444
44	,	250	108	l	278	172	¬	570	236	ì	278
45	-	570	109	m	833	173	-	333	237	í	278
46	.	250	110	n	556	174	®	747	238	î	278
47	/	278	111	o	500	175	¯	333	239	ï	278
48	0	500	112	p	556	176	°	400	240	ð	500
49	1	500	113	q	556	177	±	570	241	ñ	556
50	2	500	114	r	444	178	²	300	242	ò	500
51	3	500	115	s	389	179	³	300	243	ó	500
52	4	500	116	t	333	180	´	333	244	ô	500
53	5	500	117	u	556	181	µ	556	245	õ	500
54	6	500	118	v	500	182	¶	540	246	ö	500
55	7	500	119	w	722	183	·	250	247	÷	570
56	8	500	120	x	500	184	¸	333	248	ø	500
57	9	500	121	y	500	185	¹	300	249	ù	556
58	:	333	122	z	444	186	º	330	250	ú	556
59	;	333	123	{	394	187	»	500	251	û	556
60	<	570	124	|	220	188	¼	750	252	ü	556
61	=	570	125	}	394	189	½	750	253	ý	500
62	>	570	126	~	520	190	¾	750	254	þ	556
63	?	500	127			191	¿	500	255	ÿ	500

I.2.3.3 Times-Italic

Times-Italic

CapHeight = 653

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	920	128			192	À	611
1			65	A	611	129			193	Á	611
2			66	B	611	130			194	Â	611
3			67	C	667	131			195	Ã	611
4			68	D	722	132			196	Ä	611
5			69	E	611	133			197	Å	611
6			70	F	611	134			198	Æ	889
7			71	G	722	135			199	Ç	667
8			72	H	722	136			200	È	611
9			73	I	333	137			201	É	611
10			74	J	444	138			202	Ê	611
11			75	K	667	139			203	Ë	611
12			76	L	556	140			204	Ì	333
13			77	M	833	141			205	Í	333
14			78	N	667	142			206	Î	333
15			79	O	722	143			207	Ï	333
16			80	P	611	144			208	Ð	722
17			81	Q	722	145			209	Ñ	667
18			82	R	611	146			210	Ò	722
19			83	S	500	147			211	Ó	722
20			84	T	556	148			212	Ô	722
21			85	U	722	149			213	Õ	722
22			86	V	611	150			214	Ö	722
23			87	W	833	151			215	×	675
24			88	X	611	152			216	Ø	722
25			89	Y	556	153			217	Ù	722
26			90	Z	556	154			218	Ú	722
27			91	[	389	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	389	157			221	Ý	556
30			94	^	422	158			222	Þ	611
31			95	_	500	159			223	ß	500
32		250	96	`	333	160		250	224	à	500
33	!	333	97	a	500	161	¡	389	225	á	500
34	"	420	98	b	500	162	¢	500	226	â	500
35	#	500	99	c	444	163	£	500	227	ã	500
36	$	500	100	d	500	164	¤	500	228	ä	500
37	%	833	101	e	444	165	¥	500	229	å	500
38	&	778	102	f	278	166	¦	275	230	æ	667
39	'	333	103	g	500	167	§	500	231	ç	444
40	(	333	104	h	500	168	¨	333	232	è	444
41	)	333	105	i	278	169	©	760	233	é	444
42	*	500	106	j	278	170	ª	276	234	ê	444
43	+	675	107	k	444	171	«	500	235	ë	444
44	,	250	108	l	278	172	¬	675	236	ì	278
45	-	675	109	m	722	173	-	333	237	í	278
46	.	250	110	n	500	174	®	760	238	î	278
47	/	278	111	o	500	175	¯	333	239	ï	278
48	0	500	112	p	500	176	°	400	240	ð	500
49	1	500	113	q	500	177	±	675	241	ñ	500
50	2	500	114	r	389	178	²	300	242	ò	500
51	3	500	115	s	389	179	³	300	243	ó	500
52	4	500	116	t	278	180	´	333	244	ô	500
53	5	500	117	u	500	181	µ	500	245	õ	500
54	6	500	118	v	444	182	¶	523	246	ö	500
55	7	500	119	w	667	183	·	250	247	÷	675
56	8	500	120	x	444	184	¸	333	248	ø	500
57	9	500	121	y	444	185	¹	300	249	ù	500
58	:	333	122	z	389	186	º	310	250	ú	500
59	;	333	123	{	400	187	»	500	251	û	500
60	<	675	124	|	275	188	¼	750	252	ü	500
61	=	675	125	}	400	189	½	750	253	ý	444
62	>	675	126	~	541	190	¾	750	254	þ	500
63	?	500	127			191	¿	500	255	ÿ	444

I.2.3.4 Times-BoldItalic

Times-BoldItalic

CapHeight = 669

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	832	128			192	À	667
1			65	A	667	129			193	Á	667
2			66	B	667	130			194	Â	667
3			67	C	667	131			195	Ã	667
4			68	D	722	132			196	Ä	667
5			69	E	667	133			197	Å	667
6			70	F	667	134			198	Æ	944
7			71	G	722	135			199	Ç	667
8			72	H	778	136			200	È	667
9			73	I	389	137			201	É	667
10			74	J	500	138			202	Ê	667
11			75	K	667	139			203	Ë	667
12			76	L	611	140			204	Ì	389
13			77	M	889	141			205	Í	389
14			78	N	722	142			206	Î	389
15			79	O	722	143			207	Ï	389
16			80	P	611	144			208	Ð	722
17			81	Q	722	145			209	Ñ	722
18			82	R	667	146			210	Ò	722
19			83	S	556	147			211	Ó	722
20			84	T	611	148			212	Ô	722
21			85	U	722	149			213	Õ	722
22			86	V	667	150			214	Ö	722
23			87	W	889	151			215	×	570
24			88	X	667	152			216	Ø	722
25			89	Y	611	153			217	Ù	722
26			90	Z	611	154			218	Ú	722
27			91	[	333	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	333	157			221	Ý	611
30			94	^	570	158			222	Þ	611
31			95	_	500	159			223	ß	500
32		250	96	`	333	160		250	224	à	500
33	!	389	97	a	500	161	¡	389	225	á	500
34	"	555	98	b	500	162	¢	500	226	â	500
35	#	500	99	c	444	163	£	500	227	ã	500
36	$	500	100	d	500	164	¤	500	228	ä	500
37	%	833	101	e	444	165	¥	500	229	å	500
38	&	778	102	f	333	166	¦	220	230	æ	722
39	'	333	103	g	500	167	§	500	231	ç	444
40	(	333	104	h	556	168	¨	333	232	è	444
41	)	333	105	i	278	169	©	747	233	é	444
42	*	500	106	j	278	170	ª	266	234	ê	444
43	+	570	107	k	500	171	«	500	235	ë	444
44	,	250	108	l	278	172	¬	606	236	ì	278
45	-	606	109	m	778	173	-	333	237	í	278
46	.	250	110	n	556	174	®	747	238	î	278
47	/	278	111	o	500	175	¯	333	239	ï	278
48	0	500	112	p	500	176	°	400	240	ð	500
49	1	500	113	q	500	177	±	570	241	ñ	556
50	2	500	114	r	389	178	²	300	242	ò	500
51	3	500	115	s	389	179	³	300	243	ó	500
52	4	500	116	t	278	180	´	333	244	ô	500
53	5	500	117	u	556	181	µ	576	245	õ	500
54	6	500	118	v	444	182	¶	500	246	ö	500
55	7	500	119	w	667	183	·	250	247	÷	570
56	8	500	120	x	500	184	¸	333	248	ø	500
57	9	500	121	y	444	185	¹	300	249	ù	556
58	:	333	122	z	389	186	º	300	250	ú	556
59	;	333	123	{	348	187	»	500	251	û	556
60	<	570	124	|	220	188	¼	750	252	ü	556
61	=	570	125	}	348	189	½	750	253	ý	444
62	>	570	126	~	570	190	¾	750	254	þ	500
63	?	500	127			191	¿	500	255	ÿ	444

I.2.3.5 Helvetica

Helvetica

CapHeight = 718

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	1015	128			192	À	667
1			65	A	667	129			193	Á	667
2			66	B	667	130			194	Â	667
3			67	C	722	131			195	Ã	667
4			68	D	722	132			196	Ä	667
5			69	E	667	133			197	Å	667
6			70	F	611	134			198	Æ	1000
7			71	G	778	135			199	Ç	722
8			72	H	722	136			200	È	667
9			73	I	278	137			201	É	667
10			74	J	500	138			202	Ê	667
11			75	K	667	139			203	Ë	667
12			76	L	556	140			204	Ì	278
13			77	M	833	141			205	Í	278
14			78	N	722	142			206	Î	278
15			79	O	778	143			207	Ï	278
16			80	P	667	144			208	Ð	722
17			81	Q	778	145			209	Ñ	722
18			82	R	722	146			210	Ò	778
19			83	S	667	147			211	Ó	778
20			84	T	611	148			212	Ô	778
21			85	U	722	149			213	Õ	778
22			86	V	667	150			214	Ö	778
23			87	W	944	151			215	×	584
24			88	X	667	152			216	Ø	778
25			89	Y	667	153			217	Ù	722
26			90	Z	611	154			218	Ú	722
27			91	[	278	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	278	157			221	Ý	667
30			94	^	469	158			222	Þ	667
31			95	_	556	159			223	ß	611
32		278	96	‘	222	160		278	224	à	556
33	!	278	97	a	556	161	¡	333	225	á	556
34	"	355	98	b	556	162	¢	556	226	â	556
35	#	556	99	c	500	163	£	556	227	ã	556
36	$	556	100	d	556	164	¤	556	228	ä	556
37	%	889	101	e	556	165	¥	556	229	å	556
38	&	667	102	f	278	166	¦	260	230	æ	889
39	'	222	103	g	556	167	§	556	231	ç	500
40	(	333	104	h	556	168	¨	333	232	è	556
41	)	333	105	i	222	169	©	737	233	é	556
42	*	389	106	j	222	170	ª	370	234	ê	556
43	+	584	107	k	500	171	«	556	235	ë	556
44	,	278	108	l	222	172	¬	584	236	ì	278
45	-	584	109	m	833	173	-	333	237	í	278
46	.	278	110	n	556	174	®	737	238	î	278
47	/	278	111	o	556	175	¯	333	239	ï	278
48	0	556	112	p	556	176	°	400	240	ð	556
49	1	556	113	q	556	177	±	584	241	ñ	556
50	2	556	114	r	333	178	²	333	242	ò	556
51	3	556	115	s	500	179	³	333	243	ó	556
52	4	556	116	t	278	180	´	333	244	ô	556
53	5	556	117	u	556	181	µ	556	245	õ	556
54	6	556	118	v	500	182	¶	537	246	ö	556
55	7	556	119	w	722	183	·	278	247	÷	584
56	8	556	120	x	500	184	¸	333	248	ø	611
57	9	556	121	y	500	185	¹	333	249	ù	556
58	:	278	122	z	500	186	º	365	250	ú	556
59	;	278	123	{	334	187	»	556	251	û	556
60	<	584	124	|	260	188	¼	834	252	ü	556
61	=	584	125	}	334	189	½	834	253	ý	500
62	>	584	126	~	584	190	¾	834	254	þ	556
63	?	556	127			191	¿	611	255	ÿ	500

I.2.3.6 Helvetica-Bold

Helvetica-Bold

CapHeight = 718

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	975	128			192	À	722
1			65	A	722	129			193	Á	722
2			66	B	722	130			194	Â	722
3			67	C	722	131			195	Ã	722
4			68	D	722	132			196	Ä	722
5			69	E	667	133			197	Å	722
6			70	F	611	134			198	Æ	1000
7			71	G	778	135			199	Ç	722
8			72	H	722	136			200	È	667
9			73	I	278	137			201	É	667
10			74	J	556	138			202	Ê	667
11			75	K	722	139			203	Ë	667
12			76	L	611	140			204	Ì	278
13			77	M	833	141			205	Í	278
14			78	N	722	142			206	Î	278
15			79	O	778	143			207	Ï	278
16			80	P	667	144			208	Ð	722
17			81	Q	778	145			209	Ñ	722
18			82	R	722	146			210	Ò	778
19			83	S	667	147			211	Ó	778
20			84	T	611	148			212	Ô	778
21			85	U	722	149			213	Õ	778
22			86	V	667	150			214	Ö	778
23			87	W	944	151			215	×	584
24			88	X	667	152			216	Ø	778
25			89	Y	667	153			217	Ù	722
26			90	Z	611	154			218	Ú	722
27			91	[	333	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	333	157			221	Ý	667
30			94	^	584	158			222	Þ	667
31			95	_	556	159			223	ß	611
32		278	96	`	278	160		278	224	à	556
33	!	333	97	a	556	161	¡	333	225	á	556
34	"	474	98	b	611	162	¢	556	226	â	556
35	#	556	99	c	556	163	£	556	227	ã	556
36	$	556	100	d	611	164	¤	556	228	ä	556
37	%	889	101	e	556	165	¥	556	229	å	556
38	&	722	102	f	333	166	¦	280	230	æ	889
39	'	278	103	g	611	167	§	556	231	ç	556
40	(	333	104	h	611	168	¨	333	232	è	556
41	)	333	105	i	278	169	©	737	233	é	556
42	*	389	106	j	278	170	ª	370	234	ê	556
43	+	584	107	k	556	171	«	556	235	ë	556
44	,	278	108	l	278	172	¬	584	236	ì	278
45	?	584	109	m	889	173	-	333	237	í	278
46	.	278	110	n	611	174	®	737	238	î	278
47	/	278	111	o	611	175	¯	333	239	ï	278
48	0	556	112	p	611	176	°	400	240	ð	611
49	1	556	113	q	611	177	±	584	241	ñ	611
50	2	556	114	r	389	178	²	333	242	ò	611
51	3	556	115	s	556	179	³	333	243	ó	611
52	4	556	116	t	333	180	´	333	244	ô	611
53	5	556	117	u	611	181	µ	611	245	õ	611
54	6	556	118	v	556	182	¶	556	246	ö	611
55	7	556	119	w	778	183	·	278	247	÷	584
56	8	556	120	x	556	184	¸	333	248	ø	611
57	9	556	121	y	556	185	¹	333	249	ù	611
58	:	333	122	z	500	186	º	365	250	ú	611
59	;	333	123	{	389	187	»	556	251	û	611
60	<	584	124	|	280	188	¼	834	252	ü	611
61	=	584	125	}	389	189	½	834	253	ý	556
62	>	584	126	~	584	190	¾	834	254	þ	611
63	?	611	127			191	¿	611	255	ÿ	556

I.2.3.7 Helvetica-Oblique

Helvetica-Oblique

CapHeight = 718

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	1015	128			192	À	667
1			65	A	667	129			193	Á	667
2			66	B	667	130			194	Â	667
3			67	C	722	131			195	Ã	667
4			68	D	722	132			196	Ä	667
5			69	E	667	133			197	Å	667
6			70	F	611	134			198	Æ	1000
7			71	G	778	135			199	Ç	722
8			72	H	722	136			200	È	667
9			73	I	278	137			201	É	667
10			74	J	500	138			202	Ê	667
11			75	K	667	139			203	Ë	667
12			76	L	556	140			204	Ì	278
13			77	M	833	141			205	Í	278
14			78	N	722	142			206	Î	278
15			79	O	778	143			207	Ï	278
16			80	P	667	144			208	Ð	722
17			81	Q	778	145			209	Ñ	722
18			82	R	722	146			210	Ò	778
19			83	S	667	147			211	Ó	778
20			84	T	611	148			212	Ô	778
21			85	U	722	149			213	Õ	778
22			86	V	667	150			214	Ö	778
23			87	W	944	151			215	×	584
24			88	X	667	152			216	Ø	778
25			89	Y	667	153			217	Ù	722
26			90	Z	611	154			218	Ú	722
27			91	[	278	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	278	157			221	Ý	667
30			94	^	469	158			222	Þ	667
31			95	_	556	159			223	ß	611
32		278	96	‘	222	160		278	224	à	556
33	!	278	97	a	556	161	¡	333	225	á	556
34	"	355	98	b	556	162	¢	556	226	â	556
35	#	556	99	c	500	163	£	556	227	ã	556
36	$	556	100	d	556	164	¤	556	228	ä	556
37	%	889	101	e	556	165	¥	556	229	å	556
38	&	667	102	f	278	166	¦	260	230	æ	889
39	'	222	103	g	556	167	§	556	231	ç	500
40	(	333	104	h	556	168	¨	333	232	è	556
41	)	333	105	i	222	169	©	737	233	é	556
42	*	389	106	j	222	170	ª	370	234	ê	556
43	+	584	107	k	500	171	«	556	235	ë	556
44	,	278	108	l	222	172	¬	584	236	ì	278
45	-	584	109	m	833	173	-	333	237	í	278
46	.	278	110	n	556	174	®	737	238	î	278
47	/	278	111	o	556	175	¯	333	239	ï	278
48	0	556	112	p	556	176	°	400	240	ð	556
49	1	556	113	q	556	177	±	584	241	ñ	556
50	2	556	114	r	333	178	²	333	242	ò	556
51	3	556	115	s	500	179	³	333	243	ó	556
52	4	556	116	t	278	180	´	333	244	ô	556
53	5	556	117	u	556	181	µ	556	245	õ	556
54	6	556	118	v	500	182	¶	537	246	ö	556
55	7	556	119	w	722	183	·	278	247	÷	584
56	8	556	120	x	500	184	¸	333	248	ø	611
57	9	556	121	y	500	185	¹	333	249	ù	556
58	:	278	122	z	500	186	º	365	250	ú	556
59	;	278	123	{	334	187	»	556	251	û	556
60	<	584	124	|	260	188	¼	834	252	ü	556
61	=	584	125	}	334	189	½	834	253	ý	500
62	>	584	126	~	584	190	¾	834	254	þ	556
63	?	556	127			191	¿	611	255	ÿ	500

I.2.3.8 Helvetica-BoldOblique

Helvetica-BoldOblique

CapHeight = 718

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	975	128			192	À	722
1			65	A	722	129			193	Á	722
2			66	B	722	130			194	Â	722
3			67	C	722	131			195	Ã	722
4			68	D	722	132			196	Ä	722
5			69	E	667	133			197	Å	722
6			70	F	611	134			198	Æ	1000
7			71	G	778	135			199	Ç	722
8			72	H	722	136			200	È	667
9			73	I	278	137			201	É	667
10			74	J	556	138			202	Ê	667
11			75	K	722	139			203	Ë	667
12			76	L	611	140			204	Ì	278
13			77	M	833	141			205	Í	278
14			78	N	722	142			206	Î	278
15			79	O	778	143			207	Ï	278
16			80	P	667	144			208	Ð	722
17			81	Q	778	145			209	Ñ	722
18			82	R	722	146			210	Ò	778
19			83	S	667	147			211	Ó	778
20			84	T	611	148			212	Ô	778
21			85	U	722	149			213	Õ	778
22			86	V	667	150			214	Ö	778
23			87	W	944	151			215	×	584
24			88	X	667	152			216	Ø	778
25			89	Y	667	153			217	Ù	722
26			90	Z	611	154			218	Ú	722
27			91	[	333	155			219	Û	722
28			92	\	278	156			220	Ü	722
29			93	]	333	157			221	Ý	667
30			94	^	584	158			222	Þ	667
31			95	_	556	159			223	ß	611
32		278	96	`	278	160		278	224	à	556
33	!	333	97	a	556	161	¡	333	225	á	556
34	"	474	98	b	611	162	¢	556	226	â	556
35	#	556	99	c	556	163	£	556	227	ã	556
36	$	556	100	d	611	164	¤	556	228	ä	556
37	%	889	101	e	556	165	¥	556	229	å	556
38	&	722	102	f	333	166	¦	280	230	æ	889
39	'	278	103	g	611	167	§	556	231	ç	556
40	(	333	104	h	611	168	¨	333	232	è	556
41	)	333	105	i	278	169	©	737	233	é	556
42	*	389	106	j	278	170	ª	370	234	ê	556
43	+	584	107	k	556	171	«	556	235	ë	556
44	,	278	108	l	278	172	¬	584	236	ì	278
45	-	584	109	m	889	173	-	333	237	í	278
46	.	278	110	n	611	174	®	737	238	î	278
47	/	278	111	o	611	175	¯	333	239	ï	278
48	0	556	112	p	611	176	°	400	240	ð	611
49	1	556	113	q	611	177	±	584	241	ñ	611
50	2	556	114	r	389	178	^{2}	333	242	ò	611
51	3	556	115	s	556	179	^{3}	333	243	ó	611
52	4	556	116	t	333	180	´	333	244	ô	611
53	5	556	117	u	611	181	µ	611	245	õ	611
54	6	556	118	v	556	182	¶	556	246	ö	611
55	7	556	119	w	778	183	·	278	247	÷	584
56	8	556	120	x	556	184	¸	333	248	ø	611
57	9	556	121	y	556	185	^{1}	333	249	ù	611
58	:	333	122	z	500	186	º	365	250	ú	611
59	;	333	123	{	389	187	»	556	251	û	611
60	<	584	124	|	280	188	¼	834	252	ü	611
61	=	584	125	}	389	189	½	834	253	ý	556
62	>	584	126	~	584	190	¾	834	254	þ	611
63	?	611	127			191	¿	611	255	ÿ	556

I.2.3.9 Courier

Courier

CapHeight = 562

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	600	128			192	À	600
1			65	A	600	129			193	Á	600
2			66	B	600	130			194	Â	600
3			67	C	600	131			195	Ã	600
4			68	D	600	132			196	Ä	600
5			69	E	600	133			197	Å	600
6			70	F	600	134			198	Æ	600
7			71	G	600	135			199	Ç	600
8			72	H	600	136			200	È	600
9			73	I	600	137			201	É	600
10			74	J	600	138			202	Ê	600
11			75	K	600	139			203	Ë	600
12			76	L	600	140			204	Ì	600
13			77	M	600	141			205	Í	600
14			78	N	600	142			206	Î	600
15			79	O	600	143			207	Ï	600
16			80	P	600	144			208	Ð	600
17			81	Q	600	145			209	Ñ	600
18			82	R	600	146			210	Ò	600
19			83	S	600	147			211	Ó	600
20			84	T	600	148			212	Ô	600
21			85	U	600	149			213	Õ	600
22			86	V	600	150			214	Ö	600
23			87	W	600	151			215	×	600
24			88	X	600	152			216	Ø	600
25			89	Y	600	153			217	Ù	600
26			90	Z	600	154			218	Ú	600
27			91	[	600	155			219	Û	600
28			92	\	600	156			220	Ü	600
29			93	]	600	157			221	Ý	600
30			94	^	600	158			222	Þ	600
31			95	_	600	159			223	ß	600
32		600	96	`	600	160		600	224	à	600
33	!	600	97	a	600	161	¡	600	225	á	600
34	"	600	98	b	600	162	¢	600	226	â	600
35	#	600	99	c	600	163	£	600	227	ã	600
36	$	600	100	d	600	164	¤	600	228	ä	600
37	%	600	101	e	600	165	¥	600	229	å	600
38	&	600	102	f	600	166	¦	600	230	æ	600
39	'	600	103	g	600	167	§	600	231	ç	600
40	(	600	104	h	600	168	¨	600	232	è	600
41	)	600	105	i	600	169	©	600	233	é	600
42	*	600	106	j	600	170	ª	600	234	ê	600
43	+	600	107	k	600	171	«	600	235	ë	600
44	,	600	108	l	600	172	¬	600	236	ì	600
45	-	600	109	m	600	173	-	600	237	í	600
46	.	600	110	n	600	174	®	600	238	î	600
47	/	600	111	o	600	175	¯	600	239	ï	600
48	0	600	112	p	600	176	°	600	240	ð	600
49	1	600	113	q	600	177	±	600	241	ñ	600
50	2	600	114	r	600	178	²	600	242	ò	600
51	3	600	115	s	600	179	³	600	243	ó	600
52	4	600	116	t	600	180	´	600	244	ô	600
53	5	600	117	u	600	181	µ	600	245	õ	600
54	6	600	118	v	600	182	¶	600	246	ö	600
55	7	600	119	w	600	183	·	600	247	÷	600
56	8	600	120	x	600	184	¸	600	248	ø	600
57	9	600	121	y	600	185	¹	600	249	ù	600
58	:	600	122	z	600	186	º	600	250	ú	600
59	;	600	123	{	600	187	»	600	251	û	600
60	<	600	124	|	600	188	¼	600	252	ü	600
61	=	600	125	}	600	189	½	600	253	ý	600
62	>	600	126	~	600	190	¾	600	254	þ	600
63	?	600	127			191	¿	600	255	ÿ	600

I.2.3.10 Courier-Bold

Courier-Bold

CapHeight = 562

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	600	128			192	À	600
1			65	A	600	129			193	Á	600
2			66	B	600	130			194	Â	600
3			67	C	600	131			195	Ã	600
4			68	D	600	132			196	Ä	600
5			69	E	600	133			197	Å	600
6			70	F	600	134			198	Æ	600
7			71	G	600	135			199	Ç	600
8			72	H	600	136			200	È	600
9			73	I	600	137			201	É	600
10			74	J	600	138			202	Ê	600
11			75	K	600	139			203	Ë	600
12			76	L	600	140			204	Ì	600
13			77	M	600	141			205	Í	600
14			78	N	600	142			206	Î	600
15			79	O	600	143			207	Ï	600
16			80	P	600	144			208	Ð	600
17			81	Q	600	145			209	Ñ	600
18			82	R	600	146			210	Ò	600
19			83	S	600	147			211	Ó	600
20			84	T	600	148			212	Ô	600
21			85	U	600	149			213	Õ	600
22			86	V	600	150			214	Ö	600
23			87	W	600	151			215	×	600
24			88	X	600	152			216	Ø	600
25			89	Y	600	153			217	Ù	600
26			90	Z	600	154			218	Ú	600
27			91	[	600	155			219	Û	600
28			92	\	600	156			220	Ü	600
29			93	]	600	157			221	Ý	600
30			94	^	600	158			222	Þ	600
31			95	_	600	159			223	ß	600
32		600	96	`	600	160		600	224	à	600
33	!	600	97	a	600	161	¡	600	225	á	600
34	"	600	98	b	600	162	¢	600	226	â	600
35	#	600	99	c	600	163	£	600	227	ã	600
36	$	600	100	d	600	164	¤	600	228	ä	600
37	%	600	101	e	600	165	¥	600	229	å	600
38	&	600	102	f	600	166	¦	600	230	æ	600
39	'	600	103	g	600	167	§	600	231	ç	600
40	(	600	104	h	600	168	¨	600	232	è	600
41	)	600	105	i	600	169	©	600	233	é	600
42	*	600	106	j	600	170	ª	600	234	ê	600
43	+	600	107	k	600	171	«	600	235	ë	600
44	,	600	108	l	600	172	¬	600	236	ì	600
45	-	600	109	m	600	173	-	600	237	í	600
46	.	600	110	n	600	174	®	600	238	î	600
47	/	600	111	o	600	175	¯	600	239	ï	600
48	0	600	112	p	600	176	°	600	240	ð	600
49	1	600	113	q	600	177	±	600	241	ñ	600
50	2	600	114	r	600	178	²	600	242	ò	600
51	3	600	115	s	600	179	³	600	243	ó	600
52	4	600	116	t	600	180	´	600	244	ô	600
53	5	600	117	u	600	181	µ	600	245	õ	600
54	6	600	118	v	600	182	¶	600	246	ö	600
55	7	600	119	w	600	183	·	600	247	÷	600
56	8	600	120	x	600	184	¸	600	248	ø	600
57	9	600	121	y	600	185	¹	600	249	ù	600
58	:	600	122	z	600	186	º	600	250	ú	600
59	;	600	123	{	600	187	»	600	251	û	600
60	<	600	124	|	600	188	¼	600	252	ü	600
61	=	600	125	}	600	189	½	600	253	ý	600
62	>	600	126	~	600	190	¾	600	254	þ	600
63	?	600	127			191	¿	600	255	ÿ	600

I.2.3.11 Courier-Oblique

Courier-Oblique

CapHeight = 562

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	600	128			192	À	600
1			65	A	600	129			193	Á	600
2			66	B	600	130			194	Â	600
3			67	C	600	131			195	Ã	600
4			68	D	600	132			196	Ä	600
5			69	E	600	133			197	Å	600
6			70	F	600	134			198	Æ	600
7			71	G	600	135			199	Ç	600
8			72	H	600	136			200	È	600
9			73	I	600	137			201	É	600
10			74	J	600	138			202	Ê	600
11			75	K	600	139			203	Ë	600
12			76	L	600	140			204	Ì	600
13			77	M	600	141			205	Í	600
14			78	N	600	142			206	Î	600
15			79	O	600	143			207	Ï	600
16			80	P	600	144			208	Ð	600
17			81	Q	600	145			209	Ñ	600
18			82	R	600	146			210	Ò	600
19			83	S	600	147			211	Ó	600
20			84	T	600	148			212	Ô	600
21			85	U	600	149			213	Õ	600
22			86	V	600	150			214	Ö	600
23			87	W	600	151			215	×	600
24			88	X	600	152			216	Ø	600
25			89	Y	600	153			217	Ù	600
26			90	Z	600	154			218	Ú	600
27			91	[	600	155			219	Û	600
28			92	\	600	156			220	Ü	600
29			93	]	600	157			221	Ý	600
30			94	^	600	158			222	Þ	600
31			95	_	600	159			223	ß	600
32		600	96	`	600	160		600	224	à	600
33	!	600	97	a	600	161	¡	600	225	á	600
34	"	600	98	b	600	162	¢	600	226	â	600
35	#	600	99	c	600	163	£	600	227	ã	600
36	$	600	100	d	600	164	¤	600	228	ä	600
37	%	600	101	e	600	165	¥	600	229	å	600
38	&	600	102	f	600	166	¦	600	230	æ	600
39	'	600	103	g	600	167	§	600	231	ç	600
40	(	600	104	h	600	168	¨	600	232	è	600
41	)	600	105	i	600	169	©	600	233	é	600
42	*	600	106	j	600	170	ª	600	234	ê	600
43	+	600	107	k	600	171	«	600	235	ë	600
44	,	600	108	l	600	172	¬	600	236	ì	600
45	-	600	109	m	600	173	-	600	237	í	600
46	.	600	110	n	600	174	®	600	238	î	600
47	/	600	111	o	600	175	¯	600	239	ï	600
48	0	600	112	p	600	176	°	600	240	ð	600
49	1	600	113	q	600	177	±	600	241	ñ	600
50	2	600	114	r	600	178	²	600	242	ò	600
51	3	600	115	s	600	179	³	600	243	ó	600
52	4	600	116	t	600	180	´	600	244	ô	600
53	5	600	117	u	600	181	µ	600	245	õ	600
54	6	600	118	v	600	182	¶	600	246	ö	600
55	7	600	119	w	600	183	·	600	247	÷	600
56	8	600	120	x	600	184	¸	600	248	ø	600
57	9	600	121	y	600	185	¹	600	249	ù	600
58	:	600	122	z	600	186	º	600	250	ú	600
59	;	600	123	{	600	187	»	600	251	û	600
60	<	600	124	|	600	188	¼	600	252	ü	600
61	=	600	125	}	600	189	½	600	253	ý	600
62	>	600	126	~	600	190	¾	600	254	þ	600
63	?	600	127			191	¿	600	255	ÿ	600

I.2.3.12 Courier-BoldOblique

Courier-BoldOblique

CapHeight = 562

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	@	600	128			192	À	600
1			65	A	600	129			193	Á	600
2			66	B	600	130			194	Â	600
3			67	C	600	131			195	Ã	600
4			68	D	600	132			196	Ä	600
5			69	E	600	133			197	Å	600
6			70	F	600	134			198	Æ	600
7			71	G	600	135			199	Ç	600
8			72	H	600	136			200	È	600
9			73	I	600	137			201	É	600
10			74	J	600	138			202	Ê	600
11			75	K	600	139			203	Ë	600
12			76	L	600	140			204	Ì	600
13			77	M	600	141			205	Í	600
14			78	N	600	142			206	Î	600
15			79	O	600	143			207	Ï	600
16			80	P	600	144			208	Ð	600
17			81	Q	600	145			209	Ñ	600
18			82	R	600	146			210	Ò	600
19			83	S	600	147			211	Ó	600
20			84	T	600	148			212	Ô	600
21			85	U	600	149			213	Õ	600
22			86	V	600	150			214	Ö	600
23			87	W	600	151			215	×	600
24			88	X	600	152			216	Ø	600
25			89	Y	600	153			217	Ù	600
26			90	Z	600	154			218	Ú	600
27			91	[	600	155			219	Û	600
28			92	\	600	156			220	Ü	600
29			93	]	600	157			221	Ý	600
30			94	^	600	158			222	Þ	600
31			95	_	600	159			223	ß	600
32		600	96	`	600	160		600	224	à	600
33	!	600	97	a	600	161	¡	600	225	á	600
34	"	600	98	b	600	162	¢	600	226	â	600
35	#	600	99	c	600	163	£	600	227	ã	600
36	$	600	100	d	600	164	¤	600	228	ä	600
37	%	600	101	e	600	165	¥	600	229	å	600
38	&	600	102	f	600	166	¦	600	230	æ	600
39	'	600	103	g	600	167	§	600	231	ç	600
40	(	600	104	h	600	168	¨	600	232	è	600
41	)	600	105	i	600	169	©	600	233	é	600
42	*	600	106	j	600	170	ª	600	234	ê	600
43	+	600	107	k	600	171	«	600	235	ë	600
44	,	600	108	l	600	172	¬	600	236	ì	600
45	-	600	109	m	600	173	-	600	237	í	600
46	.	600	110	n	600	174	®	600	238	î	600
47	/	600	111	o	600	175	¯	600	239	ï	600
48	0	600	112	p	600	176	°	600	240	ð	600
49	1	600	113	q	600	177	±	600	241	ñ	600
50	2	600	114	r	600	178	²	600	242	ò	600
51	3	600	115	s	600	179	³	600	243	ó	600
52	4	600	116	t	600	180	´	600	244	ô	600
53	5	600	117	u	600	181	µ	600	245	õ	600
54	6	600	118	v	600	182	¶	600	246	ö	600
55	7	600	119	w	600	183	·	600	247	÷	600
56	8	600	120	x	600	184	¸	600	248	ø	600
57	9	600	121	y	600	185	¹	600	249	ù	600
58	:	600	122	z	600	186	º	600	250	ú	600
59	;	600	123	{	600	187	»	600	251	û	600
60	<	600	124	|	600	188	¼	600	252	ü	600
61	=	600	125	}	600	189	½	600	253	ý	600
62	>	600	126	~	600	190	¾	600	254	þ	600
63	?	600	127			191	¿	600	255	ÿ	600

I.2.3.13 Symbol

Symbol

CapHeight = 673

Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width	Char Number	Standard Char	Char Width
0			64	≅	549	128			192	ℵ	823
1			65	Α	722	129			193	ℑ	686
2			66	Β	667	130			194	ℜ	795
3			67	Χ	722	131			195	℘	987
4			68	Δ	612	132			196	⊗	768
5			69	Ε	611	133			197	⊕	768
6			70	Φ	763	134			198	∅	823
7			71	Γ	603	135			199	∩	768
8			72	Η	722	136			200	∪	768
9			73	Ι	333	137			201	⊃	713
10			74	ϑ	631	138			202	⊇	713
11			75	Κ	722	139			203	⊄	713
12			76	Λ	686	140			204	⊂	713
13			77	Μ	889	141			205	⊆	713
14			78	Ν	722	142			206	∈	713
15			79	Ο	722	143			207	∉	713
16			80	Π	768	144			208	∠	768
17			81	Θ	741	145			209	∇	713
18			82	Ρ	556	146			210	®	790
19			83	Σ	592	147			211	©	790
20			84	Τ	611	148			212	™	890
21			85	Υ	690	149			213	∏	823
22			86	ς	439	150			214	√	549
23			87	Ω	768	151			215	⋅	250
24			88	Ξ	645	152			216	¬	713
25			89	Ψ	795	153			217	∧	603
26			90	Ζ	611	154			218	∨	603
27			91	[	333	155			219	⇔	1042
28			92	∴	863	156			220	⇐	987
29			93	]	333	157			221	⇑	603
30			94	⊥	658	158			222	⇒	987
31			95	_	500	159			223	⇓	603
32		250	96		500	160			224	◊	494
33	!	333	97	α	631	161	ϒ	620	225	〈	329
34	∀	713	98	β	549	162	′	247	226	®	790
35	#	500	99	χ	549	163	≤	549	227	©	790
36	∃	549	100	δ	494	164	⁄	167	228	™	786
37	%	833	101	ε	439	165	∞	713	229	∑	713
38	&	778	102	φ	521	166	ƒ	500	230	⎛	384
39	∍	439	103	γ	411	167	♣	753	231	⎜	384
40	(	333	104	η	603	168	♦	753	232	⎝	384
41	)	333	105	ι	329	169	♥	753	233	⎡	384
42	∗	500	106	ϕ	603	170	♠	753	234	⎢	384
43	+	549	107	κ	549	171	↔	1042	235	⎣	384
44	,	250	108	λ	549	172	←	987	236	⎧	494
45	−	549	109	μ	576	173	↑	603	237	⎨	494
46	.	250	110	ν	521	174	→	987	238	⎩	494
47	/	278	111	ο	549	175	↓	603	239	⎪	494
48	0	500	112	π	549	176	°	400	240		
49	1	500	113	θ	521	177	±	549	241	〉	329
50	2	500	114	ρ	549	178	″	411	242	∫	274
51	3	500	115	σ	603	179	≥	549	243	⌠	686
52	4	500	116	τ	439	180	×	549	244	⎮	686
53	5	500	117	υ	576	181	∝	713	245	⌡	686
54	6	500	118	ϖ	713	182	∂	494	246	⎞	384
55	7	500	119	ω	686	183	•	460	247	⎟	384
56	8	500	120	ξ	493	184	÷	549	248	⎠	384
57	9	500	121	ψ	686	185	≠	549	249	⎤	384
58	:	278	122	ζ	494	186	≡	549	250	⎥	384
59	;	278	123	{	480	187	≈	549	251	⎦	384
60	<	549	124	|	200	188	…	1000	252	⎫	494
61	=	549	125	}	480	189	⏐	603	253	⎬	494
62	>	549	126	~	549	190	⎯	1000	254	⎭	494
63		444	127			191	↵	658	255		

ICS 35.140

Price based on 448 pages

INTERNATIONAL STANDARD ISO/IEC 8632-1:1999
TECHNICAL CORRIGENDUM 1

Published 2006-04-01

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 1: **Functional specification**

TECHNICAL CORRIGENDUM 1

Technologies de l'information — Infographie — Métafichier de stockage et de transfert des informations de description d'images —

Partie 1: Description fonctionnelle

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO/IEC 8632-1:1999 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics, image processing and environmental data representation.*

Page 177:

In the last sentence of the second paragraph of the "Description:" section of 7.6.24, change:

"The end value shall be less than the *n-th* knot value (where *n* is the number of control points)."

to:

"The end value shall be less than or equal to the $(n+1)^{st}$ knot value (where *n* is the number of control points)."

ICS 35.140 **Ref. No. ISO/IEC 8632-1:1999/Cor.1:2006(E)**

Published in Switzerland

Page 178:

In the last sentence of the third paragraph of the "Description:" section of 7.6.25, change:

"The end value shall be less than or equal to the *n-th* knot value (where *n* is the number of control points)."

to:

"The end value shall be less than or equal to the $(n+1)^{st}$ knot value (where *n* is the number of control points)."

INTERNATIONAL STANDARD ISO/IEC 8632-1:1999
TECHNICAL CORRIGENDUM 2

Published 2007-11-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 1: Functional specification

TECHNICAL CORRIGENDUM 2

Technologies de l'information — Infographie — Métafichier de stockage et de transfert des informations de description d'images —

Partie 1: Description fonctionnelle

RECTIFICATIF TECHNIQUE 2

Technical Corrigendum 2 to ISO/IEC 8632-1:1999 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics, image processing and environmental data representation*.

Page 105, Table 8a

In the APPEND TEXT line, change the SOS column from blank to “X”.

Page 108, Table 8b

Add a BEGIN APPLICATION STRUCTURE line before the BEGIN APPLICATION STRUCTURE BODY line, with all columns blank except for “X” in the TOS column.

ICS 35.140 **Ref. No. ISO/IEC 8632-1:1999/Cor.2:2007(E)**

Page 161, 7.6.4

Before the paragraph beginning "When the value of the *flag* parameter…", insert the following new paragraph:

"The *flag* parameter also allows, in Version 4 metafiles, the definition within a text string of an Application Structure that consists of a sub-string of the text string."

Page 162, 7.6.5

Before the paragraph beginning "When the value of the *flag* parameter…", insert the following new paragraph:

"The *flag* parameter also allows, in Version 4 metafiles, the definition within a text string of an Application Structure that consists of a sub-string of the text string."

Page 163, 7.6.6

Before the paragraph beginning "When the value of the *flag* parameter…", insert the following new paragraph:

"The *flag* parameter also allows, in Version 4 metafiles, the definition within a text string of an Application Structure that consists of a sub-string of the text string."

Page 229, 9.5.4.6

Replace:

> "one for the group of elements with data type D parameters which have type SF data within the D parameters, in the case that the D parameter uses SDR formatting and contain type SF data."

with:

> "one for the group of elements with either data type SDR parameters which contain type SF data within the SDR parameter, or data type D parameters which contain SF data in the D parameter, in the case that the D parameter uses SDR formatting."

Page 230, 9.5.4.6

Delete FONT PROPERTIES from the list titled "Type SF Parameters" and insert it into the list titled "SF data within SDR parameters". Insert APPLICATION STRUCTURE ATTRIBUTE and GLYPH MAPPING in the list titled "SF data within SDR parameters".

Page 326, H.3.1

Replace the 8-line `<application structure>` production with the following equivalent formulation:

```
<application structure> ::=
        <BEGIN APPLICATION STRUCTURE>
        <application structure descriptor>
        <BEGIN APPLICATION STRUCTURE BODY>
        <picture element>*
        <END APPLICATION STRUCTURE>

<application structure descriptor> ::=
        <application structure identifier>
        <application structure type>
        <inheritance flag enumerated>
        <application structure descriptor element>
```

Page 337, H.3.5

Replace the four productions, `<text tail>` through `<spanned text>` (ten lines), with the following:

```
<text tail> ::= <simple tail> | <continuation tail>

<simple tail> ::= <FINAL> <string>

<continuation tail> ::=
        <NOT FINAL>
        <string>
        <nonfinal substring>* <final substring>

<final substring> ::=
        <partial text attribute element>*
        <APPEND TEXT> <FINAL>
        <string>

<nonfinal substring> ::= <substring object> | <simple substring>

<simple substring> ::=
        <partial text attribute element>*
        <APPEND TEXT> <NOT FINAL>
        <string>
        <partial text attribute element>*

<substring object> ::=
        <BEGIN APPLICATION STRUCTURE>
        <application structure descriptor>
        <BEGIN APPLICATION STRUCTURE BODY>
        <simple substring>+
        <END APPLICATION STRUCTURE>
```

Page 359, Annex I, Table 14

In row T.14.5, change the two occurrences of “for type SF within type D” to “for type SF within type D and within type SDR”. (One occurrence is in the column “Specifications – PPF” and the other is in the column “Specifications – Model Profile”.)

INTERNATIONAL STANDARD

ISO/IEC 8632-3

Second edition
1999-12-15

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 3: **Binary encoding**

Technologies de l'information — Infographie — Métafichier de stockage et de transfert des informations de description d'images —

Partie 3: Codage binaire

Reference number
ISO/IEC 8632-3:1999(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 734 10 79
E-mail copyright@iso.ch
Web www.iso.ch

Printed in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 8632 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 8632-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics and image processing*.

This second edition cancels and replaces the first edition (ISO/IEC 8632-3:1992), which has been technically revised. It also incorporates Amendment 1:1994 and Amendment 2:1995. Note that the previous edition of ISO/IEC 8632-3, published in 1992, was a first edition but second edition was indicated by error on its cover page and in the foreword.

ISO/IEC 8632 consists of the following parts, under the general title *Information technology — Computer graphics — Metafile for the storage and transfer of picture description information*:

— *Part 1: Functional specification*

— *Part 3: Binary encoding*

— *Part 4: Clear text encoding*

Annex A forms a normative part of this part of ISO/IEC 8632. Annexes B and C are for information only.

NOTE In previous editions of ISO/IEC 8632, Part 2 defined a Character Encoding. Part 2 was withdrawn in 1998, due to its lack of implementation and use.

Introduction

0.1 Purpose of the Binary Encoding

The Binary Encoding of the Computer Graphics Metafile (CGM) provides a representation of the Metafile syntax that can be optimized for speed of generation and interpretation, while still providing a standard means of interchange among computer systems. The encoding uses binary data formats that are much more similar to the data representations used within computer systems than the data formats of the other encodings.

Some of the data formats may exactly match those of some computer systems. In such cases processing is reduced very much relative to the other standardized encodings.

On most computer systems processing requirements for the Binary Encoding will be substantially lower than another encoding.

In cases where a computer system's architecture does not match the standard formats used in the Binary Encoding, and where absolute minimization of processing requirements is critical, and where interchange among dissimilar systems does not matter, it may be more appropriate to use a private encoding, conforming to the rules specified in clause 7 of ISO/IEC 8632-1.

0.2 Objectives

This encoding has the following features.

a) Partitioning of parameter lists: metafile elements are coded in the Binary Encoding by one or more partitions (see clause 5); the first (or only) partition of an element contains the opcode (Element Class plus Element Id).

b) Alignment of elements: every element begins on a word boundary. When the data of an element (whether partitioned or not) does not terminate on an even-octet boundary, then the following element is aligned by padding after the data of the preceding element with zero bits to the next even-octet boundary. A no-op element is available in this encoding. It is skipped and ignored by interpreters. It may be used to align data on machine-dependent record boundaries for speed of processing.

c) Uniformity of format: all elements have an associated parameter length value. The length is specified as an octet count. As a result, it is possible to scan the metafile, without interpreting it, at high speed.

d) Alignment of coordinate data: at default precisions and by virtue of alignment of elements, coordinate data always start on word boundaries. This minimises processing by ensuring, on a wide class of computing systems, that single coordinates do not have to be assembled from pieces of multiple computer words.

e) Efficiency of encoding integer data: other data such as indexes, colour and characters are encoded as one or more octets. The precision of every parameter is determined by the appropriate precision as given in the Metafile Descriptor.

f) Order of bit data: in each word, or unit within a word, the bit with the highest number is the most significant bit. Likewise, when data words are accessed sequentially, the least significant word follows the most significant.

g) Extensibility: the arrangement of Element Class and Element Id values has been designed to allow future growth, such as new graphical elements.

h) Format of real data: real numbers are encoded using either IEEE floating point representation or a metafile fixed-point representation.

i) Run length encoding: if many adjacent cells have the same colour (or colour index) efficient encoding is possible. For each run a cell count is specified followed by the colour (or colour index).

j) Packed list encoding: if adjacent colour cells do not have the same colour (or colour index) the metafile provides bit-stream lists in which the values are packed as closely as possible.

0.3 Relationship to other International Standards

The floating point representation of real data in this part of ISO/IEC 8632 is that in ANSI/IEEE 754-1986.

The representation of character data in this part of ISO/IEC 8632 follows the rules of ISO/IEC 646 and ISO 2022.

For certain elements, the CGM defines value ranges as being reserved for registration. The values and their meanings will be defined using the established procedures (see ISO/IEC 8632-1, 6.12.)

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 3: **Binary encoding**

1 Scope

This part of ISO/IEC 8632 specifies a binary encoding of the Computer Graphics Metafile. For each of the elements specified in ISO/IEC 8632-1, this part specifies an encoding in terms of data types.

For each of these data types, an explicit representation in terms of bits, octets and words is specified. For some data types, the exact representation is a function of the precisions being used in the metafile, as recorded in the Metafile Descriptor.

This encoding of the Computer Graphics Metafile will, in many circumstances, minimize the effort required to generate and interpret the metafile.

2 Conformance

Conformance of metafiles to ISO/IEC 8632 is defined in terms of profiles. A metafile conforms to this encoding if it conforms to a profile and meets the following criteria:

— Each metafile element described in this part shall be encoded in the manner described in this part of this International Standard and a profile.

— Metafile elements which are not defined in Part 1 or in this encoding are all encoded using the GENERALIZED DRAWING PRIMITIVE or ESCAPE metafile elements as appropriate. According to the profile rules of Part 1 (see clause 9, subclause 9.5.2.8), such elements shall either be profile defined or registered, in order that the profile be valid. Inclusion of private elements is not permissible in a valid profile of ISO/IEC 8632 and this encoding.

— Values of index parameters, which are used as enumeration selectors from lists of implicitly defined attribute values, shall either be standard, registered, or profile defined. The standard and registered values are all non-negative, and the profile-defined shall be negative. Use of private, implicitly-defined negative index values which are not profile defined is not permissible in a valid profile of ISO/IEC 8632 and this encoding.

— Values specified as being "reserved for registered values" shall not be used unless their meaning has been registered or standardized.

— Inclusion of non-graphical data in the metafile shall be accomplished with the APPLICATION DATA element or with the APPLICATION STRUCTURE ATTRIBUTE element.

See clause 10 for additional conformance information about this encoding.

3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 8632. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO/IEC 8632 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 646:1991, *Information technology — ISO 7-bit coded character set for information interchange.*

ISO 2022:1986, *Information processing — ISO 7-bit and 8-bit coded character sets — Code extension techniques.*

ANSI/IEEE 754, *Standard for Binary Floating Point Arithmetic.*

4 Notational conventions

“Command Header” is used throughout this part of ISO/IEC 8632 to refer to that portion of a Binary-Encoded element that contains the opcode (element class plus element id) and parameter length information (see clause 5).

Within this part, the terms “octet” and “word” have specific meanings. These meanings may not match those of a particular computer system on which this encoding of the metafile is used.

An octet is an 8-bit entity. All bits are significant. The bits are numbered from 7 (most significant) to 0 (least significant).

A word is a 16-bit entity. All bits are significant. The bits are numbered from 15 (most significant) to 0 (least significant).

5 Overall structure

5.1 General form of metafile

All elements in the metafile are encoded using a uniform scheme. The elements are represented as variable length data structures, each consisting of opcode information (element class plus element id) designating the particular element, the length of its parameter data and finally the parameter data (if any).

The structure of the metafile is as follows. (For the purposes of this diagram only, MF is used as an abbreviation for METAFILE.)

BEGIN MF	MD	<picture> ...	END MF

The BEGIN METAFILE element is followed by the Metafile Descriptor (MD). After this the pictures follow, each logically independent of each other. Finally the Metafile is ended with an END METAFILE element.

5.2 General form of pictures

Apart from the BEGIN METAFILE, END METAFILE and Metafile Descriptor elements, the metafile is partitioned into pictures. All pictures are mutually independent. A picture consists of a BEGIN PICTURE element, a Picture Descriptor (PD), a BEGIN PICTURE BODY element, an arbitrary number of control, graphical and attribute elements, and finally an END PICTURE element.

(For the purpose of this diagram only, PIC is used as an abbreviation for PICTURE and BEGIN BODY for BEGIN PICTURE BODY.)

BEGIN PIC	PD	BEGIN BODY	<element> ..	END PIC

5.3 General structure of the binary metafile

The binary encoding of the metafile is a logical data structure consisting of a sequential collection of bits.

For convenience in describing the length and alignment of metafile elements, fields of two different sizes are defined within the structure. These fields are used in the remainder of this part of ISO/IEC 8632 for illustrating the contents and structure of elements and parameters.

For measuring the lengths of elements the metafile is partitioned into octets, which are 8-bit fields.

The structure is also partitioned into 16-bit fields called words (these are logical metafile words). To optimize processing of the binary metafile on a wide collection of computers, metafile elements are constrained to start on word boundaries within the binary data structure (this alignment may necessitate padding an element with bits to a word boundary if the parameter data of the element does not fill to such a boundary).

The octet is the fundamental unit of organization of the binary metafile.

The bits of an octet are numbered 7 to 0, with 7 being the most significant bit. The bits of a word are numbered 15 to 0, with 15 being the most significant bit.

```
            b7          b0
            +-+-+-+-+-+-+-+-+
   octet:   |               |
            +-+-+-+-+-+-+-+-+
            msb          lsb

            b15       b8.b7          b0
            +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   word:    |               |               |
            +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
            msb                          lsb
```

In the preceding diagram, msb means most significant bit and lsb means least significant bit.

If the consecutive bits of the binary data structure are numbered 1..N, and the consecutive octets are numbered 1..N/8, and the consecutive words are numbered 1..N/16, then the logical correspondence of bits, octets, and words in the binary data structure is as illustrated in the following table:

metafile bit number	octet bit number	word bit number
1	b7/octet1	b15/word1
.	.	.
.	.	.
8	b0/octet1	b8/word1
9	b7/octet2	b7/word1
.	.	.
.	.	.
16	b0/octet2	b0/word1
17	b7/octet3	b15/word2
.	.	.
.	.	.
24	b0/octet3	b8/word2
25	b7/octet4	b7/word2
.	.	.
.	.	.

5.4 Structure of the command header

Throughout this sub-clause, the term “command” is used to denote a binary-encoded element. Metafile elements are represented in the Binary Encoding in one of two forms — short-form commands and long-form commands. There are two differences between them:

— a short-form command always contains a complete element; the long-form command can accommodate partial elements (the data lists of elements can be partitioned);

— a short-form command only accommodates parameter lists up to 30 octets in length; the long-form command accommodates lengths up to 32767 octets per data partition.

The forms differ in the format of the Command Header that precedes the parameter list. The command form for an element (short or long) is established by the first word of the element. For the short-form, the Command Header consists of a single word divided into three fields: element class, element id and parameter list length.

	15 14 13 12	11 10 9 8 7 6 5	4 3 2 1 0
Word 1	element class	element id	parameter list length

Figure 1 — Format of a short-form Command Header

The fields in the short-form Command Header are as follows:

bits 15 to 12	element class (value range 0 to 15)
bits 11 to 5	element id (value range 0 to 127)
bits 4 to 0	parameter list length: the number of octets of parameter data that follow for this command (value range 0 to 30)

This Command Header is then followed by the parameter list.

The first word of a long-form command is identical in structure to the first word of a short-form command. The presence of the value 11111 binary (decimal 31) in the parameter list length field indicates that the command is a long-form command..The Command Header for the long-form command consists of two words. The second word contains the actual parameter list length. The two header words are then followed by the parameter list.

In addition to allowing longer parameter lists, the long-form command allows the parameter list to be partitioned. Bit 15 of the second word indicates whether the given data complete the element or more data follow. For subsequent data partitions of the element, the first word of the long-form Command Header (containing element class and element id) is omitted; only the second word, containing the parameter list length, is given. The parameter list length for each partition specifies the length of that partition, not the length of the complete element. The final partition of an element is indicated by bit 15 of the parameter list length word being zero.

<table>
<tr><td></td><td>15</td><td>14 13 12</td><td>11 10 9 8 7 6 5</td><td>4 3 2 1 0</td></tr>
<tr><td>Word 1</td><td colspan="2">element class</td><td>element id</td><td>1 1 1 1 1</td></tr>
<tr><td>Word 2</td><td>P</td><td colspan="3">parameter list length</td></tr>
</table>

Figure 2 — Format of a long-form Command Header

The fields in the long-form Command Header are as follows:

Word 1

bits 15 to 12	element class (value range 0 to 15)
bits 11 to 5	element id (value range 0 to 127)
bits 4 to 0	binary value 11111 (decimal 31) indicating long-form

Word 2

bit 15 partition flag

— 0 for 'last' partition

— 1 for 'not-last' partition

bits 14 to 0 parameter list length: the number of octets of parameter data that follow for this command or partition (value range 0 to 32767).

The parameter values follow the parameter list length for either the long-form or short-form commands. The number of values is determined from the parameter list length and the type and precision of the operands. These parameter values have the format illustrated in clause 5 of this part of ISO/IEC 8632. The parameter type for coordinates is indicated in the Metafile Descriptor. For non-coordinate parameters, the parameter type is as specified in clause 5 of ISO/IEC 8632-1. If the parameter type is encoding dependent, its code is specified in the coding tables of clause 7 of this part. Unless otherwise stated, the order of parameters is as listed in clause 5 of ISO/IEC 8632-1.

Every command is constrained to begin on a word boundary. This necessitates padding the command with a single null octet at the end of the command if the command contains an odd number of octets of parameter data. In addition, in elements with parameters whose precisions are shorter than one octet (i.e., those containing a 'local colour precision' parameter) it is necessary to pad the last data-containing octet with null bits if the data do not fill the octet. In all cases, the parameter list length is the count of octets actually containing parameter data — it does not include the padding octet if one is present. It is only at the end of a command that padding is performed, with the single exception of the CELL ARRAY element.

The purpose of this command alignment constraint is to optimize processing on a wide class of computers. At the default metafile precisions, the parameters which are expected to occur in greatest numbers (coordinates, etc) will align on 16-bit boundaries, and Command Headers will align on 16-bit boundaries. Thus, at the default precisions the most frequently parsed entities will lie entirely within machine words in a large number of computer designs. The avoidance of assembling single metafile parameters from pieces of several computer words will approximately halve the amount of processing required to recover element parameters and command header fields from a binary metafile data stream.

This optimization may be compromised or destroyed altogether if the metafile precisions are changed from default. Commands are still constrained to begin on 16-bit boundaries, but the most frequently expected parameters may no longer align on such boundaries as they do at the default precisions.

The short form command header with element class 15, element id 127, and parameter list length 0 is reserved for extension of the number of available element classes in future revisions of this part of ISO/IEC 8632. It should be treated by interpreters as any other element, as far as parsing is concerned. The next “normal” element encountered will have an actual class value different from that encountered in the “element class” field of the command header — it will be adjusted by a bias as will be defined in a future revision of this part of ISO/IEC 8632.

6 Primitive data forms

The Binary Encoding of the CGM uses five primitive data forms to represent the various abstract data types used to describe parameters in ISO/IEC 8632-1.

The primitive data forms and the symbols used to represent them are as follows.

SI	Signed Integer
UI	Unsigned Integer
C	Character
FX	Fixed Point Real
FP	Floating Point Real

Each of these primitive forms (except Character) can be used in a number of precisions. The definitions of the primitive data forms in 6.1 to 6.5 show the allowed precisions for each primitive data form. The definitions are in terms of 'metafile words' which are 16-bit units.

The following terms are used in the following diagrams when displaying the form of numeric values.

msb	most significant bit
lsb	least significant bit
S	sign bit

The data types in the following data diagrams are illustrated for the case that the parameter begins on a metafile word boundary. In general, parameters may align on odd or even octet boundaries, because they may be preceded by an odd or even number of octets of other parameter data. Elements containing the local colour precision parameter may have parameters shorter than one octet. It is possible in such cases that the parameters will not align on octet boundaries.

6.1 Signed integer

Signed integers are represented in “two’s complement” format. Four precisions may be specified for signed integers: 8-bit, 16-bit, 24-bit and 32-bit. (Integer coordinate data encoded with this primitive data form do not use the 8-bit precision.) In the diagrams of the following subsections, 'value' indicates the value for positive integers and the two’s complement of the value for negative integers.

6.1.1 Signed integer at 8-bit precision

Each value occupies half a metafile word (one octet).

15	14 13 12 11 10 9 8	7	6 5 4 3 2 1 0
S	msb value i lsb	S	msb value i+1 lsb

6.1.2 Signed integer at 16-bit precision

Each value occupies one metafile word.

15	14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
S	msb value lsb

6.1.3 Signed integer at 24-bit precision

Each value straddles two successive metafile words.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 1	S	msb												value i		
Word 2		value i						lsb	S	msb					value i+1	
Word 3							value i+1									lsb

6.1.4 Signed integer at 32-bit precision

Each value fills two complete metafile words.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 1	S	msb								value i						
Word 2				value i												lsb

6.2 Unsigned integer

Four precisions may be specified for unsigned integers: 8-bit, 16-bit, 24-bit and 32-bit.

6.2.1 Unsigned integers at 8-bit precision

Each value occupies half a metafile word.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
msb			value i				lsb	msb			value i+1				lsb

6.2.2 Unsigned integers at 16-bit precision

Each value occupies one metafile word.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
msb							value								lsb

6.2.3 Unsigned integers at 24-bit precision

Each value straddles two successive metafile words.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 1	msb							value i								
Word 2				value i				lsb	msb				value i+1			
Word 3								value i+1								lsb

6.2.4 Unsigned integers at 32-bit precision

Each value fills two complete metafile words.

	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		
Word 1	msb	value i	
Word 2		value i	lsb

6.3 Character

Each character is stored in 1 or more consecutive octets, depending upon the coding of the particular character set. The following illustrates characters which are coded with 1 octet each.

15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Character i	Character i+1

6.4 Fixed point real

Fixed point real values are stored as two integers; the first represents the “whole part” and has the same form as a Signed Integer (SI; see 6.1); the second represents the “fractional part” and has the same form as an Unsigned Integer (UI; see 6.2). Two precisions may be specified for Fixed Point Reals: 32-bit or 64-bit.

6.4.1 Fixed point real at 32-bit precision

Each Fixed Point Real occupies 2 complete metafile words; the first has the form of a 16-bit Signed Integer and the second the form of a 16-bit Unsigned Integer.

	15	14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		
Word 1	S	msb	Whole part	lsb
Word 2	msb		Fraction part	lsb

6.4.2 Fixed point real at 64-bit precision

Each Fixed Point Real occupies 4 complete metafile words; the first has the form of a 32-bit Signed Integer and the second the form of a 32-bit Unsigned Integer.

	15	14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		
Word 1	S	msb	whole part	
Word 2	whole part			lsb
Word 3	msb		fraction part	
Word 4	fraction part			lsb

6.4.3 Value of fixed point reals

The values of the represented real numbers are given by:

for 32 bits: $$\text{real\_value} = SI + \left[\frac{UI}{2^{16}}\right]$$

for 64 bits: real_value = $SI + \left[\dfrac{UI}{2^{32}} \right]$

SI stands for the “whole part” and UI stands for the “fractional part” in these equations. SI, the whole part, is the largest integer less than or equal to the real number being represented.

6.5 Floating point

Floating Point Real values are represented in the floating point format of ANSI/IEEE 754. This format contains three parts:

- a sign bit ('s');
- a biased exponent part ('e');
- a fraction part ('f').

The value is a function of these three values ('s’, 'e’ and 'f’). If 's’ is '0’, the value is positive; if 's’ is '1’, the value is negative. Two precisions may be specified for Floating Point Reals: 32-bit or 64-bit. The magnitude of the value is calculated as follows for 32-bit representation.

a) If e = 255 and f $\neq$ 0, then the value is undefined.

b) If e = 255 and f = 0, then the value is as large a positive (s=0) or negative (s=1) value as possible.

c) If 0 < e < 255, then the magnitude of the value is $(1.f)(2^{e-127})$.

d) If e = 0 and f $\neq$ 0, then the magnitude of the value is $(0.f)(2^{-126})$.

e) If e = 0 and f = 0, then the value is 0.

The magnitude of the value is calculated as follows for 64-bit representation.

a) If e = 2047 and f $\neq$ 0, then the value is undefined.

b) If e = 2047 and f = 0, then the value is as large a positive (s=0) or negative (s=1) value as possible.

c) If 0 < e < 2047, then the magnitude of the value is $(1.f)(2^{e-1023})$.

d) If e = 0 and f $\neq$ 0, then the magnitude of the value is $(0.f)(2^{-1022})$.

e) If e = 0 and f = 0, then the value is 0.

6.5.1 Floating point real at 32-bit precision

Each Floating Point Real value occupies 2 metafile words. The size of each field in the value is as follows:

sign	1 bit
exponent	8 bits
fraction	23 bits

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 1	S	msb		Exponent					lsb	msb		Fraction				
Word 2							Fraction									lsb

6.5.2 Floating point real at 64-bit precision

Each Floating Point Real value occupies 4 metafile words. The size of each field in the value is as follows:

sign	1 bit
exponent	11 bits
fraction	52 bits

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 1	S	msb				Exponent						lsb	msb			
Word 2						Fraction										
Word 3						Fraction										
Word 4						Fraction										lsb

7 Representation of abstract parameter types

Table 1 shows, for each of the abstract parameter types, how it is represented in the Binary Encoding of the CGM in terms of primitive data forms. The columns of the table are as follows:

1) The symbol for the abstract parameter type, as it is specified in clause 5 of ISO/IEC 8632-1.
2) The way the parameter type is constructed in terms of the primitive data forms, at the appropriate precisions. The precisions are those defined in clause 5 of ISO/IEC 8632-1.
3) The symbol for the number of octets required to represent one instance (occurrence) of the given parameter, at the given precision, and the formula for computing the number.
4) The symbol for the range of values which the parameter can assume, followed by the numerical values which the parameter can assume, followed by the numerical values which define the range.

The symbols of columns 3 and 4 are used extensively in the code tables in clause 7. Also used in the code tables are variations on those symbols:

+IR, +RR, ..	denote the range of positive integers, range of positive reals, ..
-IR, -RR, ..	denote the range of negative integers, range of negative reals, ..
++IR, ++RR, ..	denote the range of non-negative integers, range of non-negative reals, ..
mI, mR,..	denotes 'm' integers, reals, ..
I*, R*,..	denotes an unbounded number of integers, reals, ..

Combinations are used:

2R, 2I, IX*,..	indicates a parameter that is represented by 2 reals, then a parameter that is represented by 2 integers and finally a parameter that contains an unlimited number of index values.

Table 1 — Representation of abstract data types

Abstract symbol	Parameter construction from primitive forms	Octets per parameter: symbol and value	Parameter range: symbol and value
CI	UI at colour index precision (cip)	BCI {=cip/8}	CIR {0..(2^{cip} - 1)}
CCO	UI at direct colour precision (dcp)	BCCO {=dcp/8}	CCOR {0..(2^{dcp} - 1)} {see NOTE 2}
CD	(CCO,CCO,CCO) or (CCO,CCO,CCO,CCO)	BCD ={3*BCCO} BCD ={4*BCCO}	CCOR {see NOTE 1, NOTE 16}
IX	SI at index precision (ixp)	BIX {=ixp/8}	IXR {-2^{ixp-1} to 2^{ixp-1} - 1 }
E	SI at fixed precision (16-bit) {see NOTE 3}	BE {=2}	{-2^{15} to 2^{15} - 1 } {see NOTE 18}
I	SI at integer precision (ip)	BI {=ip/8}	IR {-2^{ip-1} to 2^{ip-1} - 1 }
R	FP or FX at real precision (rp)	BR {=sum(rp)/8} {see NOTE 4}	RR {=FPR or FXR, see NOTE 5, NOTE 10}
S,SF,D	UI,nC	BS {see NOTE }	SR {see NOTE 6, NOTE 12}
VDC	SI at VDC integer precision (vip) or FP or FX at VDC real precision (vrp)	BVDC {=vip/8} or BVDC {=sum(vrp)/8} {see NOTE 4}	VDCR {-2^{vip-1} to 2^{vip-1} - 1 } or VDCR {see NOTE 1, NOTE 5, NOTE 7, NOTE 8}
P	(VDC,VDC)	BP {=2*BVDC}	VDCR {see NOTE 1, NOTE 5, NOTE 7, NOTE 8}
CO	CI or CD	BCO {=BCI} or BCO {=BCD}	COR {=CIR} {see NOTE 9, NOTE 11} or COR {=CCOR}
N	SI at name precision (np)	BN {=np/8}	NR {-2^{np-1} to 2^{np-1} - 1}
VC	I or R	BVC {=BI} or BVC {=BR}	VCR {=IR} {see NOTE 13} or VCR {=RR}
VP	(VC,VC)	BVP {=2*BVC}	VCR {see NOTE 1, NOTE 13, NOTE 14}

Table 1 — Representation of abstract data types (continued)

Abstract symbol	Parameter construction from primitive forms	Octets per parameter: symbol and value	Parameter range: symbol and value
BS	nUI at fixed precision (16-bit) {see NOTE 15}	BBS {=2n}	BSR {see NOTE 15}
UI8	UI at fixed precision (8-bit)	BUI8 {=1}	UI8R {0..255}
UI32	UI at fixed precision (32-bit)	BUI32 {=4}	UI32R {0..2^{32} - 1}
SDR	{see NOTE 17}	BSDR	n/a
SS	VDC or R	BSS {=BVDC} or BSS {=BR}	SSR {=VDCR} {see NOTE 19} or SSR {=RR}

The following 19 notes contain additional normative specifications of this encoding:

NOTE 1 For parameters that are composed of multiple identical components (e.g., DIRECT COLOUR, CD, and POINT, P) the range value represents the range of a single component.

NOTE 2 For colour models RGB and CMYK a direct colour component is abstractly a real in the range [0,1]. For colour models CIELAB, CIELUV, and RGB-related it is abstractly a real in the respective colour spaces with possibly different ranges for the direct colour components. The COLOUR VALUE EXTENT element provides for the mapping between a direct colour component represented as UI and the corresponding real value.

NOTE 3 Abstract parameter type Enumeration, E, is encoded identically to abstract type Index, IX, at 16-bit precision.

NOTE 4 The REAL PRECISION element contains an indicator (fixed or floating point) and two precision components. The symbol "sum(rp)" in the table indicates the sum of the number of bits specified in the two components. The same considerations apply to the VDC REAL PRECISION element and the symbol "sum(vrp)" in the tables. The VDC REAL PRECISION control element may cause ‘vrp’ to be updated in the body of metafile.

NOTE 5 FPR and VDCR (when VDC are floating point reals) are computed following the ANSI/IEEE 754 floating point standard (see clause 6 on the floating point data form).

NOTE 6 The range for parameter types S and SF is not applicable. The range for character data is not applicable. A string is encoded as a count (unsigned integer) followed by characters. The count is a count of octets in the string, not whole character codes (the two are equal for single byte codes, but not for multi-byte codes).

The encoding of the count is similar to the encoding of length information for metafile commands themselves. If the first octet is in the range 0..254, then it represents the character count for the complete string. If the first octet is 255, then the next 16 bits contain the character count and a continuation flag. The first bit is used as a continuation flag (allowing strings longer than 32767 characters) and the next 15 bits represent the count, 0..32767, for the partial string. If the first bit is 0, then this partial string completes the string parameter. If 1, then this partial string will be followed by another.

If the number of whole character codes in a string is n, and the number of octets per character code is constant within the string and equal to m, and if the string is not continued (as a long-form string may be), then the number of octets in the string parameter is either n·m+1 or n·m+3, depending upon whether the string is short-form or long-form, respectively. If the number of octets per character code is not constant and/or the string is a continued long-form string, then the number of octets is the string is not so easily expressed, but is the total of the octets used in the "data" part of the string and the number of octets used for length information.

Table 1 — Representation of abstract data types (continued)

NOTE 7 The abstract parameter type VDC, a single VDC value, is either a real or an integer, depending on the declaration of the Metafile Descriptor function VDC TYPE. Subsequent tables use a single set of symbols, VDC, BVDC and VDCR, recognizing that they are computed differently depending on VDC TYPE.

NOTE 8 The abstract parameter type VDC is a single value; a point, P, is an ordered pair of VDC.

NOTE 9 The parameter type symbol CO corresponds to the data type CO of ISO/IEC 8632-1. It is either direct colour (CD) or indexed colour (CI), depending on the value specified in the COLOUR SELECTION MODE element. The associated octets per parameter and range symbols, BCO and COR, are thus either BCI and CIR or BCD and CDR respectively depending upon COLOUR SELECTION MODE.

NOTE 10 To eliminate the need to support IEEE floating point in applications that do not need the dynamic range for parameters of type R and VDC, a fixed point real format is provided for scalars (such as line width, character spacing) and VDC. Fixed point reals consist of a (SI,UI) pair.

Fixed point reals (FX) apply to VDC, and to all metafile parameters of type R except for:

a) the *metric scale factor* parameter of the SCALING MODE element;

b) the *metric scale factor* parameter of the DEVICE VIEWPORT SPECIFICATION MODE element;

In this encoding, these parameters are always encoded as floating point.

NOTE 11 CELL ARRAY colour can optionally specify 1, 2, 4, 8, 16, 24 or 32 bit precisions for cell colours, as well as using the default CI or CD precision.

The way in which the colour values in CELL ARRAY is represented is an extension of the representation of single colour values. The CELL ARRAY element has a 'cell representation flag' which may take one of two values:

0	run length representation
1	packed representation

For PACKED mode, each row of the cell array is represented by an array of colour values without compression. Each row starts on a word boundary. No row length information is stored since all rows are the same length.

For all rows of the cell array, except possibly the last row, the colour data thus occupies $2n_y$ (1+ [(p $\cdot$ n_x - 1) / 16]) octets, where nx is the number of cells per row, ny is the number of rows, p is the number of bits per colour, and [..] denotes "the greatest integer in .." Because the last row does not have a subsequent row which must align on a word boundary, which alignment (for all other rows but the last) potentially requires the addition to the end of the row of a padding byte, the color data of the last row occupies n_y (1+ [(p $\cdot$ n_x - 1) / 8]) octets (however, see clause 10).

For RUN LENGTH encoding, the data for each row begins on a word boundary and consists of run-length-lists for runs of constant colour value. Each 'run-length-list' consists of a count of a number of consecutive cells and the representation of that colour. In terms of the abstract terms above, the colour list is of format <I,CO>* and its length is <BI,BCO>*. With the exception of the first run of a row, the integer count of each run immediately follows the colour specifier of the preceding run with no intervening padding.

NOTE 12 Abstract parameter type Data Record, D, is encoded in this part similarly to string data. However, the constraints on character code values and the character set switching mechanisms (both those related to CHARACTER SET INDEX, and the purely ISO 2022 switching methods) do not necessarily apply to data records, as they do to the structurally similar S and SF parameters.

How the data are encoded, the meaning of the data bytes in the record, and the effect (if any) of character set switching mechanisms are part of the definitions of the individual Escape, GDP, and External elements to which the data record belongs.

The coding technique of the SDR data type (see Table 1, NOTE 17) is one valid form for a Data Record parameter. This form is recommended for GDP, Escape, and External element proposals submitted for Graphical Registration.

The coding tables in clause 8 will use the symbol D for the parameter type, and will use the S-related symbols for other information about the parameter.

Table 1 — Representation of abstract data types (continued)

NOTE 13 The abstract parameter type VC, a single VC value, is either a real or an integer, depending on the declaration of the picture descriptor element DEVICE VIEWPORT SPECIFICATION MODE. When DEVICE VIEWPORT SPECIFICATION MODE is 'fraction of display surface', the value is real. When DEVICE VIEWPORT SPECIFICATION MODE is 'millimetres with scale factor' or 'physical device coordinates', the value is integer. Subsequent tables use a single set of values, VC, BVC and VCR, recognizing that they are computed differently depending on DEVICE VIEWPORT SPECIFICATION MODE.

NOTE 14 The abstract parameter type VC is a single value; a viewport point, VP, is an ordered pair of VC.

NOTE 15 The bitstream (BS) data type is encoded as a stream of binary digits (bits) packed in 16-bit unsigned integers. The BS data type is used in part 1 of this Standard for the compressed colour specifier lists of Tile Array elements. A bitstream type parameter shall be encoded in the Binary Encoding of this part with the smallest number of whole 16-bit words which will hold the bits of the parameter data. If the parameter data bits do not exactly fill an integral number of 16-bit words, the remaining bits in the last word shall be 0. The range for parameter type BS is not applicable.

NOTE 16 The abstract parameter type CD is a 3-tuple or 4-tuple of CCO depending on COLOUR MODEL.

NOTE 17 The structured data record (SDR) of part 1 of this International Standard is composed entirely of other standardized datatypes (including SDR itself) in a structure which is self-defining. SDR is encoded by encoding each of the component operands according to the normal encoding rules for its corresponding data type. The string of octets comprising the encoded operands is then treated as an operand of type S — it is preceded by a string count, short form or long form, and can be continued if long form (see NOTE 6, above).

NOTE 18 Ranges for enumerated and index parameters indicated by {n..m} in tables 3-10 refer to standardized values. Index ranges are subject to extension by registration.

NOTE 19 The parameter type symbol SS corresponds to parameter type SS of ISO/IEC 8632-1. It is not a basic data type. It is a shorthand for data which can be VDC or Real, depending upon an associated specification mode. The associations of these modes with the various element parameters are defined in subclause 7.1 of part 1.

8 Representation of each element

8.1 Method of presentation

The elements are grouped according to their class; there are 10 classes.

Table 2 — List of element class codes

Class	Type of Elements
0	Delimiter elements
1	Metafile Descriptor elements
2	Picture Descriptor elements
3	Control elements
4	Graphical Primitive elements
5	Attribute elements
6	Escape element
7	External elements
8	Segment Control and Segment Attribute elements
9	Application Structure Descriptor elements
10-15	Reserved for future standardization

A complete list of element id codes and element class codes is given in Annex C.

For each class this clause contains a subclause which consists of a table and a set of notes. The table specifies the metafile element, element id, parameter type, parameter list length, and parameter range. The parameter list length is given in octets, which in some cases is constant and in other cases is variable. Any element that does not consist of an even number octets is padded with zero bits to the next 16-bit boundary before the command header of the next element is written to the metafile — elements begin on 16-bit boundaries.

The defaults for the elements are as given in clause 8 of ISO/IEC 8632-1.

This clause specifies some of the constraints on parameter values. The specifications are not exhaustive, for example such constraints as the non-collinearity of text vectors are not stated. All parameter value and other element state constraints of ISO/IEC 8632-1, including those of the formal grammars, shall apply to metafiles encoded according to this part.

8.2 Delimiter elements

Table 3 — Encoding of delimiter elements

Element Class 0	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
no-op	0	see below	n	n/a
BEGIN METAFILE	1	SF	BS	SR
END METAFILE	2	n/a	0	n/a
BEGIN PICTURE	3	SF	BS	SR
BEGIN PICTURE BODY	4	n/a	0	n/a
END PICTURE	5	n/a	0	n/a
BEGIN SEGMENT	6	N	BN	NR
END SEGMENT	7	n/a	0	n/a
BEGIN FIGURE	8	n/a	0	n/a
END FIGURE	9	n/a	0	n/a
BEGIN PROTECTION REGION	13	IX	BIX	+IXR
END PROTECTION REGION	14	n/a	0	n/a
BEGIN COMPOUND LINE	15	n/a	0	n/a
END COMPOUND LINE	16	n/a	0	n/a
BEGIN COMPOUND TEXT PATH	17	n/a	0	n/a
END COMPOUND TEXT PATH	18	n/a	0	n/a
BEGIN TILE ARRAY	19	P, 2E, 4I,2R, 2I,2I	BP+ 2BE+ 4BI+2BR+ 4BI	VDCR, {0..3},{0,1} +IR,+RR ++IR,+IR
END TILE ARRAY	20	n/a	0	n/a
BEGIN APPLICATION STRUCTURE	21	SF,SF,E	2BS+BE	SR,SR,{0,1}
BEGIN APPLICATION STRUCTURE BODY	22	n/a	0	n/a
END APPLICATION STRUCTURE	23	n/a	0	n/a

Additional description of the elements in Table 3:

Code	Description
0	no-op: has 1 parameter: P1: an arbitrary sequence of n octets, n=0,1,2.. The parameter, unlike all other parameters in the binary encoding, is not constructed from the primitive data forms — it is an arbitrary sequence of zero or more octets for padding purposes.
1	BEGIN METAFILE: has 1 parameter: P1: (string fixed) metafile name
2	END METAFILE: has no parameters.
3	BEGIN PICTURE: has 1 parameter: P1: (string fixed) picture name
4	BEGIN PICTURE BODY: has no parameters.
5	END PICTURE: has no parameters.
6	BEGIN SEGMENT: has 1 parameter: P1: (name) segment identifier
7	END SEGMENT: has no parameters
8	BEGIN FIGURE: has no parameters
9	END FIGURE: has no parameters
13	BEGIN PROTECTION REGION: has 1 parameter: P1: (index) region index.

14 END PROTECTION REGION: has no parameters.

15 BEGIN COMPOUND LINE: has no parameters:

16 END COMPOUND LINE: has no parameters.

17 BEGIN COMPOUND TEXT PATH: has no parameters:

18 END COMPOUND TEXT PATH: has no parameters.

19 BEGIN TILE ARRAY: has 13 parameters:
P1: (point) position.
P2: (enumerated) cell path direction: valid values are

0	0°
1	90°
2	180°
3	270°

P3: (enumerated) line progression direction: valid values are

0	90°
1	270°

P4: (integer) number of tiles in pth direction.

P5: (integer) number of tiles in line direction.

P6: (integer) number of cells/tile in path direction.

P7: (integer) number of cells/tile in line direction.

P8: (real) cell size in path direction.

P9: (real) cell size in line direction.

P10: (integer) image offset in path direction.

P11: (integer) image offset in line direction.

P12: (integer) image number of cells in path direction.

P13: (integer) image number of cells in line direction.

20 END TILE ARRAY: has no parameters.

21 BEGIN APPLICATION STRUCTURE:has three parameters

P1: (string fixed)application structure identifier

P2: (string fixed)application structure type

P3: (enumerated)inheritance flag:valid values are

0	STATE LIST
1	APPLICATION STRUCTURE

22 BEGIN APPLICATION STRUCTURE BODY:has no parameters

23 END APPLICATION STRUCTURE:has no parameters

8.3 Metafile descriptor elements

Table 4 — Encoding of metafile descriptor elements

Element Class 1	Element Id	Parameter Type	Parameter List Length	Parameter Range
METAFILE VERSION	1	I	BI	+IR(1..n)
METAFILE DESCRIPTION	2	SF	BS	SR
VDC TYPE	3	E	BE	0,1
INTEGER PRECISION	4	I	BI	8,16,24,32
REAL PRECISION	5	E,2I	BE+2BI	{0,1}, {9,12,16,32}, {23,52,16,32}
INDEX PRECISION	6	I	BI	8,16,24,32
COLOUR PRECISION	7	I	BI	8,16,24,32
COLOUR INDEX PRECISION	8	I	BI	8,16,24,32
MAXIMUM COLOUR INDEX	9	CI	BCI	CIR
COLOUR VALUE EXTENT	10	2CD or 6R	2BCD or 6BR	CCOR or RR
METAFILE ELEMENT LIST	11	I,2nIX	BI,2nBIX	++IR,IXR
METAFILE DEFAULTS REPLACEMENT	12	Metafile elements	variable	Metafile elements
FONT LIST	13	nSF	nBS	SR
CHARACTER SET LIST	14	n(E,SF)	n(BE+BS)	{0..4},SR
CHARACTER CODING ANNOUNCER	15	E	BE	0,1,2,3
NAME PRECISION	16	I	BI	8,16,24,32
MAXIMUM VDC EXTENT	17	2P	2BP	VDCR
SEGMENT PRIORITY EXTENT	18	2I	2BI	++IR
COLOUR MODEL	19	IX	BIX	+IXR
COLOUR CALIBRATION	20	IX,3R, 18R,I, 6nCCO,I, mCD,3mR	BIX+3BR+ 18BR+BI+ 6nBCCO+BI+ mBCD+3mBR	+IXR,RR, RR,++IR, CCOR,++IR, CCOR,RR
FONT PROPERTIES	21	n[IX,I,SDR]	n(BIX+BI)+ (sum of)BSDR	n/a
GLYPH MAPPING	22	IX,E, SF,I, IX,SDR	BIX+BE+ BS+BI+ BIX+BSDR	+IXR,ER SR,+IR IXR,n/a
SYMBOL LIBRARY LIST	23	nSF	nBS	SR
PICTURE DIRECTORY	24	E,n(SF,2[ldt])	BE+n(BS+2B[ldt])	{0,1,2}, (SR,[ldt]R,[ldt]R)

Additional description of the elements in table 4:

Code Description

1 METAFILE VERSION: has 1 parameter:

P1: (integer) metafile version number: valid values are 1, 2, 3, 4

2 METAFILE DESCRIPTION: has 1 parameter:

P1: (string fixed) metafile description string

3 VDC TYPE: has 1 parameter:

P1: (enumerated) VDC TYPE: valid values are

0 VDC values specified in integers

1 VDC values specified in reals

4 INTEGER PRECISION: has 1 parameter:

P1: (integer) integer precision: valid values are 8, 16, 24 or 32

5 REAL PRECISION: has 3 parameters:

P1: (enumerated) form of representation for real values: valid values are

0 floating point format

1 fixed point format

P2: (integer) field width for exponent or whole part (including 1 bit for sign)

P3: (integer) field width for fraction or fractional part

Legal combinations of values are

P1	P2	P3	Result
0	9	23	32-bit floating point
0	12	52	64-bit floating point
1	16	16	32-bit fixed point
1	32	32	64-bit fixed point

6 INDEX PRECISION: has 1 parameter:

P1: (integer) Index precision: valid values are 8,16,24,32

7 COLOUR PRECISION: has 1 parameter:

P1: (integer) Colour precision: valid values are 8,16,24,32

8 COLOUR INDEX PRECISION: has 1 parameter:

P1: (integer) Colour index precision: valid values are 8,16,24,32

9 MAXIMUM COLOUR INDEX: has 1 parameter:

P1: (colour index) maximum colour index that may be encountered in the metafile.

10 COLOUR VALUE EXTENT has variable parameters depending upon the colour model:

If the model is RGB or CMYK, then 2 parameters:

P1: (direct colour value) minimum colour value

P2: (direct colour value) maximum colour value

If the model is CIELAB, CIELUV, or RGB-related then 3 parameters:

P1: (real) scale and offset pair for first component.

P2: (real) scale and offset pair for second component.

P3: (real) scale and offset pair for third component.

11 METAFILE ELEMENTS LIST: has 2 parameters:

P1: (integer) number of elements specified

P2: (index-pair array) List of metafile elements in this metafile. Each element is represented by two values: the first is its element class code (as in Table 2) and the second is its element id code (as in Table 3 to Table 10). These codes are listed in Annex C. The shorthand pseudo-elements are represented by

drawing set:	(-1,0)
drawing-plus-control set:	(-1,1)
version-2 set:	(-1,2)
extended-primitives set:	(-1,3)
version-2-gksm set:	(-1,4)
version-3 set:	(-1,5)
version-4 set	(-1,6)

12 METAFILE DEFAULTS REPLACEMENT: has 1 parameter that itself contains metafile elements. The structure and format is identical to appropriate metafile element(s).

13 FONT LIST: has a variable parameter list:

P1-Pn: (string fixed) n font names

14 CHARACTER SET LIST: has a variable number of parameter pairs; for each of these:

P1: (enumerated) CHARACTER SET TYPE: valid codes are

0 94-character G-set
1 96-character G-set
2 94-character multibyte G-set
3 96-character multibyte G-set
4 complete code

P2: (string fixed) Designation sequence tail; see Part 1, subclause 7.3.14.

15 CHARACTER CODING ANNOUNCER: has 1 parameter:

P1: (enumerated) character coding announcer: valid values are

0 basic 7-bit
1 basic 8-bit
2 extended 7-bit
3 extended 8-bit

16 NAME PRECISION: has 1 parameter:

P1: (integer) name precision: valid values are 8, 16, 24 or 32

17 MAXIMUM VDC EXTENT: has 2 parameters:

P1: (point) first corner

P2: (point) second corner

18 SEGMENT PRIORITY EXTENT: has 2 parameters:

P1: (integer) minimum segment priority value: valid values are non-negative integers

P2: (integer) maximum segment priority value: valid values are non-negative integers

19 COLOUR MODEL: has 1 parameter:

P1: (index) colour model: valid values are

1 RGB
2 CIELAB
3 CIELUV
4 CMYK
5 RGB-related
>5 reserved for registered values.

20 COLOUR CALIBRATION: has 13 parameters

P1: (index) calibration selection, valid values are

1 unspecified
2 reference white only
3 reference white, matrix1
4 reference white, matrix1, lookup tables
5 reference white, matrix1, lookup tables, matrix2
6 reference white, matrix1, matrix2
7 lookup tables, matrix2
8 matrix2
9 reference white, grid locations + grid values
>9 reserved for registered values

P2: (real) reference white value X component

P3: (real) reference white value Y component

P4: (real) reference white value Z component

P5: (real) 3x3 RGB calibration matrix: Xr, Xg, Xb, Yr, Yg, Yb, Zr, Zg, Zb.

P6: (real) 3x3 ABC transformation matrix: Ra, Rb, Rc, Ga, Gb, Gc, Ba, Bb, Bc

P7: (integer) number of lookup table entries (=n), valid values are non-negative integers.

P8: (colour component) 2n red lookup table entries: R, R'.

P9: (colour component) 2n green lookup table entries: G, G'.

P10: (colour component) 2n blue lookup table entries: B, B'.

P11:(integer) number of grid locations (=m), valid values are non-negative integers.

P12:(direct colour list) m CMYK grid locations.

P13: (m*(3 real)) m XYZ grid locations, each being: CIEXYZ-X, CIEXYZ-Y, CIEXYZ-Z

21 FONT PROPERTIES: has a variable number of parameter 3-tuples (P1,P2,P3); each parameter 3-tuple contains

P1: (index) property indicator, valid values are

1 font index
2 standard version
3 design source
4 font family
5 posture
6 weight
7 proportionate width
8 included glyph collections
9 included glyphs
10 design size
11 minimum size
12 maximum size
13 design group
14 structure
>14 reserved for registered values

P2: (integer) priority, valid values are non-negative integers.

P3: (structured data record) property value record, each record contains a single member and is comprised of [data type indicator, data element count, data element(s)]. Valid values of the records are

[(integer: i_IX) (integer: 1) (index: font-index)]
| [(integer: i_I) (integer: 1) (integer: standard-version)]
| [(integer: i_SF) (integer: 1) (string fixed: design-source)]
| [(integer: i_SF) (integer: 1) (string fixed: font-family)]
| [(integer: i_IX) (integer: 1) (index: posture)]
| [(integer: i_IX) (integer: 1) (index: weight)]
| [(integer: i_IX) (integer: 1) (index: proportionate-width)]
| [(integer: i_IX) (integer: n) (included-glyph-collections)(n)]
| [(integer: i_UI32) (integer: m) (included-glyphs)(m)]
| [(integer: i_R) (integer: 1) (real: design-size)]
| [(integer: i_R) (integer: 1) (real: minimum-size)]
| [(integer: i_R) (integer: 1) (real: maximum-size)]
| [(integer: i_UI8) (integer: 3) (design-group)]
| [(integer: i_IX) (integer: 1) (index: structure)]

NOTE 1 i_XX in the above denotes the integer value of the 'data type indicator' for data type "XX" as assigned in annex C of ISO/IEC 8632-1. For example i_IX represents the designator for data type IX, which is assigned the value 2.

NOTE 2 See NOTE 17, Table 1, for additional SDR formatting requirements.

(index) font index, valid values are positive integers.

(integer) standard version, valid values are

1 for ISO/IEC 9541:1991, first version

(string fixed) design source

(string fixed) font family

(index) posture, valid values are

0 not applicable
1 upright

2 oblique
3 back slanted oblique
4 italic
5 back slanted italic
6 other
>6 reserved for registered values

(index) weight, valid values are

0 not applcable
1 ultra light
2 extra light
3 light
4 semi light
5 medium
6 semi bold
7 bold
8 extra bold
9 ultra bold
>9 reserved for registered values

(index) proportionate-width, valid values are

0 not applicable
1 ultra condensed
2 extra condensed
3 condensed
4 semi condensed
5 medium
6 semi expanded
7 expanded
8 extra expanded
9 ultra expanded
>9 reserved for registered values

(index list) included glyph collections: one or more character set indexes.

(index list) included glyphs: 1 or more AFII 32-bit glyph identifiers of type UI32.

(real) design size: valid values are positive reals.

(real) minimum size: valid values are positive reals.

(real) maximum size: valid values are positive reals.

(3 octets) design group: a 3-tuple of parameters of type octet, which respectively define the class, subclass, and specific group components of the design group.

(index) structure: valid values are

0 undefined or not applicable
1 solid
2 outline
>2 reserved for registered values

22 GLYPH MAPPING: has 6 parameters:

P1: (index) character set index

P2: (enumerated) basis set character set type: valid values are as for CHARACTER SET LIST.

P3: (string fixed) basis set designation sequence tail: valid values are as for CHARACTER SET LIST.

P4: (integer) octets per code (=m), valid values are positive integers.

P5: (index) glyph source, valid values are

1 afii registry of 4-byte glyph identifiers
>1 reserved for registered values

P6: (structured data record) glyph-code associations. For glyph source value 1: contains two members, a code list and a glyph-name list:

[(integer: i_UI8), (integer: n(m+1)), (n(UI8,mUI8): list of (run-count,m-byte-code))]
[(integer: i_UI32), (integer: n), (n(UI32: glyph-name))]

NOTE 3 The code list is a list of run length specifiers, (UI8,mUI8), where each specifier encodes a sequence of 1 or more character codes. The first octet is the run count. If the first octet of the specifier equals 1, then only the single explicitly specified m-octet code is encoded (m is the value of P4). If the first octet is greater than 1, then the m-octet code is the base of a run sequence. Each of code in the sequence is 1 greater than the previous code. The glyph-name sequence is "parallel" to the code sequence. The glyph names are associated with the corresponding codes, and when there is a run longer than 1 in the codes, there is also a run longer than 1 in the glyph names. Each glyph name in a run is 1 greater than its predecessor.

NOTE 4 See NOTE 17, Table 1, for additional SDR formatting requirements.

23 SYMBOL LIBRARY LIST: has a variable parameter list

P1-Pn: n symbol library names (string fixed), the first name in the list is assigned to index 1, the second to index 2, etc.

24 PICTURE DIRECTORY:has 2 parameters:

P1: (enumerated) location data type selector: valid values are

0 UI8
1 UI16
2 UI32

P2: list of 3-tuples consisting of:

Picture Identifier (string fixed)
Picture Location ([ldt]) offset, in octets, from the beginning of the metafile
Application Structure Directory Location ([ldt]) offset, in octets, from the beginning of the metafile

NOTE 5 [ldt] designates UI8, UI16, UI32 as selected by location data type selector parameter. The values of picture-location are the offsets in octets from the beginning of the metafile to the start of the associated BEGIN PICTURE element. The values of Application Structure Directory Location are the offsets in octets from the start of the metafile to the start of the APPLICATION STRUCTURE DIRECTORY element of the associated picture.

8.4 Picture descriptor elements

Table 5 — Encoding of picture descriptor elements

Element Class 2	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
SCALING MODE	1	E,R (FP)	BE+BFP	{0,1},FPR
COLOUR SELECTION MODE	2	E	BE	{0,1}
LINE WIDTH SPECIFICATION MODE	3	E	BE	{0..3}
MARKER SIZE SPECIFICATION MODE	4	E	BE	{0..3}
EDGE WIDTH SPECIFICATION MODE	5	E	BE	{0..3}
VDC EXTENT	6	2P	2BP	VDCR
BACKGROUND COLOUR	7	CD	BCD	CCOR
DEVICE VIEWPORT	8	2VP	2BVP	VCR
DEVICE VIEWPORT SPECIFICATION MODE	9	E,R(FP)	BE+BFP	{0,1,2},FPR
DEVICE VIEWPORT MAPPING	10	3E	3BE	{0,1} {0,1,2} {0,1,2}
LINE REPRESENTATION	11	2IX, SS,CO	2BIX+ BSS+BCO	+IXR,IXR, +SSR,COR
MARKER REPRESENTATION	12	2IX, SS,CO	2BIX+ BSS+BCO	+IXR,IXR, ++SSR,COR
TEXT REPRESENTATION	13	2IX, E, 2R,CO	2BIX+ BE+ 2BR+BCO	+IXR, {0,1,2} RR,++RR,COR
FILL REPRESENTATION	14	IX, E,CO, 2IX	BIX+ BE+BCO+ 2BIX	+IXR, {0..6},COR, IXR,+IXR
EDGE REPRESENTATION	15	2IX SS,CO	2BIX BSS+BCO	IXR,+IXR ++SSR,COR
INTERIOR STYLE SPECIFICATION MODE	16	E	BE	{0..3}
LINE AND EDGE TYPE DEFINITION	17	IX,SS,nI	BIX+BSS+nBI	- IXR,+SSR,++IR
HATCH STYLE DEFINITION	18	IX,E 4SS,SS I,nI, nIX	BIX+BE+ 4BSS+BSS+ BI+nBI+ nBIX	-IXR,{0,1}, ++SSR,+SSR +IR,++IR IXR
GEOMETRIC PATTERN DEFINITION	19	IX,N, 2P	BIX+BN+ 2BP	+IXR,NR VDCR
APPLICATION STRUCTURE DIRECTORY	20	E,n(SF,[ldt])	BE+n(BS+B[ldt])	{0,1,2}, (SR,[ldt]R,[ldt]R)

Additional description of the elements in table 5:

Code Description

1 SCALING MODE: has 2 parameters:

P1: (enumerated) scaling mode: valid values are

0 abstract scaling
1 metric scaling

P2: (real) metric scaling factor, ignored if P1=0

This parameter is always encoded as floating point, regardless of the value of the fixed/floating flag of REAL PRECISION. If a REAL PRECISION (floating, n, m) has preceded, then the precision used is n,m. If a REAL PRECISION element for floating point has not preceded, then a default precision of 9,23 (32-bit floating point) is used.

2 COLOUR SELECTION MODE: has 1 parameter:

P1: (enumerated) colour selection mode:

0 indexed colour mode
1 direct colour mode

3 LINE WIDTH SPECIFICATION MODE: has 1 parameter:

P1: (enumerated) line width specification mode: valid values are

0 absolute
1 scaled
2 fractional
3 mm

4 MARKER SIZE SPECIFICATION MODE: has 1 parameter:

P1: (enumerated) marker size specification mode: valid values are

0 absolute
1 scaled
2 fractional
3 mm

5 EDGE WIDTH SPECIFICATION MODE: has 1 parameter:

P1: (enumerated) edge width specification mode: valid values are

0 absolute
1 scaled
2 fractional
3 mm

6 VDC EXTENT: has 2 parameters:

P1: (point) first corner

P2: (point) second corner

7 BACKGROUND COLOUR: has 1 parameter:

P1: (direct colour) background colour.

8 DEVICE VIEWPORT: has 2 parameters:

P1: (viewport point) first corner

P2: (viewport point) second corner

9 DEVICE VIEWPORT SPECIFICATION MODE: has 2 parameters:

P1: (enumerated) VC specifier: valid values are

0 fraction of drawing surface
1 millimetres with scale factor
2 physical device coordinates

P2: (real) metric scale factor, ignored if P1=0 or P1=2

This parameter is always encoded as floating point, regardless of the value of the fixed/floating flag of REAL PRECISION. If a REAL PRECISION (floating, n, m) has preceded, then the precision used is n,m. If a REAL PRECISION element for floating point has not preceded, then a default precision of 9,23 (32-bit floating point) is used.

10 DEVICE VIEWPORT MAPPING: has 3 parameters:

P1: (enumerated) isotropy flag: valid values are

0 not forced
1 forced

P2: (enumerated) horizontal alignment flag: valid values are

0 left
1 centre
2 right

P3: (enumerated) vertical alignment flag: valid values are

0 bottom
1 centre
2 top

11 LINE REPRESENTATION: has 4 parameters:

P1: (index) line bundle index

P2: (index) line type: valid values are

1 solid
2 dash
3 dot
4 dash-dot
5 dash-dot-dot
>5 reserved for registered values
negative for private use

P3: (size specification) line width: see Part 1, subclause 7.1 for its form.

P4: (colour) line colour: its form depends on COLOUR SELECTION MODE.

12 MARKER REPRESENTATION: has 4 parameters:

P1: (index) marker bundle index

P2: (index) marker type: valid values are

1 dot
2 plus
3 asterisk
4 circle
5 cross
>5 reserved for registered values
negative for private use

P3: (size specification) marker size: see Part 1, subclause 7.1 for its form.

P4: (colour) marker colour: its form depends on COLOUR SELECTION MODE.

13 TEXT REPRESENTATION: has 6 parameters:

P1: (index) text bundle index

P2: (index) text font index

P3: (enumerated) text precision: valid values are

0 string
1 character
2 stroke

P4: (real) character spacing

P5: (real) character expansion factor

P6: (colour) text colour; its form depends on COLOUR SELECTION MODE

14 FILL REPRESENTATION: has 5 parameters:

P1: (index) fill area bundle index

P2: (enumerated) interior style: valid values are

0 hollow
1 solid
2 pattern
3 hatch
4 empty
5 geometric pattern
6 interpolated

P3: (colour) fill colour: its form depends on COLOUR SELECTION MODE

P4: (index) hatch index: the following values are standardized:

1 horizontal
2 vertical
3 positive slope
4 negative slope
5 horizontal/vertical crosshatch
6 positive/negative slope crosshatch
>6 reserved for registered values
negative for private use

P5: (index) pattern index

15 EDGE REPRESENTATION: has 4 parameters:

P1: (index) edge bundle index

P2: (index) edge type: the following values are standardized:

1 solid
2 dash
3 dot
4 dash-dot
5 dash-dot-dot
>5 reserved for registered values
negative for private use

P3: (size specification) edge width: see Part 1, subclause 7.1 for its form.

P4: (colour) edge colour: its form depends on COLOUR SELECTION MODE.

16 INTERIOR STYLE SPECIFICATION MODE: has 1 parameter:

P1: (enumerated) valid values are

0 absolute
1 scaled
2 fractional
3 mm

17 LINE AND EDGE TYPE DEFINITION: has a variable parameter list:

P1: (index) line type, valid values are negative.

P2: (size specification) dash cycle repeat length: see Part 1, subclause 7.1 for its form.

P3-P(n+2): (integer) list of n dash elements

18 HATCH STYLE DEFINITION: has a variable parameter list:

P1: (index) hatch index, valid values are negative.

P2: (enumerated) style indicator: valid values are

0 parallel
1 cross hatch

P3: (4(size specification)) hatch direction vectors specifier (x,y,x,y): see Part 1, subclause 7.1 for its form.

P4: (size specification) duty cycle length: see Part 1, subclause 7.1 for its form.

P5: (integer) number of hatch lines (=n)

P6-P(5+n): (integers) list of n gap widths

P(6+n)-P(5+2n): (integers) list of n line types

19 GEOMETRIC PATTERN DEFINITION: has 4 parameters

P1: (index) geometric pattern index

P2: (name) segment identifier

P3: (point) first corner point

P4: (point) second corner point

20 APPLICATION STRUCTURE DIRECTORY:has 2 parameters

P1: (enumerated) location data type selector: valid values are

0 UI8
1 UI16
2 UI32

P2: list of pairs consisting of:

Application Structure Identifier (string fixed)
Application Structure Location ([ldt]) offsets, in octets, from the beginning of the picture containing the APS

NOTE - [ldt] designates UI8, UI16, UI32 as selected by location data type selector parameter. The values of Application Structure Location are the offsets in octets from the beginning of the BEGIN PICTURE element to the start of the associated BEGIN APPLICATION STRUCTURE element.

8.5 Control elements

Table 6 — Encoding of control elements

Element Class 3	Element Id	Parameter Type	Parameter List Length	Parameter Range
VDC INTEGER PRECISION	1	I	BI	16,24,32
VDC REAL PRECISION	2	E,2I	BE+2BI	{0,1}, {9,12,16,32}, {23,52,16,32}
AUXILIARY COLOUR	3	CO	BCO	COR
TRANSPARENCY	4	E	BE	{0,1}
CLIP RECTANGLE	5	2P	2BP	VDCR
CLIP INDICATOR	6	E	BE	{0,1}
LINE CLIPPING MODE	7	E	BE	{0,1,2}
MARKER CLIPPING MODE	8	E	BE	{0,1,2}
EDGE CLIPPING MODE	9	E	BE	{0,1,2}
NEW REGION	10	n/a	0	n/a
SAVE PRIMITIVE CONTEXT	11	N	BN	NR
RESTORE PRIMITIVE CONTEXT	12	N	BN	NR
PROTECTION REGION INDICATOR	17	2IX	2BIX	+IXR,{1,2,3}
GENERALIZED TEXT PATH MODE	18	E	BE	{0,1,2}
MITRE LIMIT	19	R	BR	++RR
TRANSPARENT CELL COLOUR	20	E,CO	BE+BCO	{0,1},COR

Additional description of the elements in table 6:

Code Description

1 VDC INTEGER PRECISION: has 1 parameter:

P1: (integer) VDC integer precision; legal values are 16, 24 or 32; the value 8 is not permitted.

2 VDC REAL PRECISION: has 3 parameters:

P1: (enumerated) form of representation for real values: valid values are

0 floating point format
1 fixed point format

P2: (integer) field width for exponent or whole part (including 1 bit for sign)

P3: (integer) field width for fraction or fractional part

Legal combinations of values are

P1	P2	P3	Result
0	9	23	32-bit floating point
0	12	52	64-bit floating point
1	16	16	32-bit fixed point
1	32	32	64-bit fixed point

3 AUXILIARY COLOUR: has 1 parameter; its form depends on COLOUR SELECTION MODE:

P1: (colour) auxiliary colour

4 TRANSPARENCY: has 1 parameter:

P1: (enumerated) on-off indicator: valid values are

0 off: auxiliary colour background is required
1 on: transparent background is required

5 CLIP RECTANGLE: has 2 parameters:

P1: (point) first corner

P2: (point) second corner

6 CLIP INDICATOR: has 1 parameter:

P1: (enumerated) clip indicator: valid values are

0 off
1 on

7 LINE CLIPPING MODE: has 1 parameter:

P1: (enumerated) clipping mode: valid values are

0 locus
1 shape
2 locus then shape

8 MARKER CLIPPING MODE: has 1 parameter:

P1: (enumerated) clipping mode: valid values are

0 locus
1 shape
2 locus then shape

9 EDGE CLIPPING MODE: has 1 parameter:

P1: (enumerated) clipping mode: valid values are

0 locus
1 shape
2 locus then shape

10 NEW REGION: has no parameters

11 SAVE PRIMITIVE CONTEXT: has 1 parameter:

P1: (name) context name

12 RESTORE PRIMITIVE CONTEXT: has 1 parameter:

P1: (name) context name

17 PROTECTION REGION INDICATOR: has 2 parameters:

P1: (index) region index

P2: (index) region indicator: valid values are

1 off
2 clip
3 shield

18 GENERALIZED TEXT PATH MODE: has 1 parameter:

P1: (enumerated) text path mode: valid values are

0 off
1 non-tangential
2 axis-tangential

19 MITRE LIMIT: has 1 parameter:

P1: (real) mitre limit

20 TRANSPARENT CELL COLOUR: has 2 parameters:

P1: (enumerated) transparency indicator, valid values are

0 off
1 on

P2: (colour) transparent cell colour specifier

8.6 Graphical primitive elements

Table 7 — Encoding of graphical primitive elements

Element Class 4	Element Id	Parameter Type	Parameter List Length	Parameter Range
POLYLINE	1	nP	nBP	VDCR
DISJOINT POLYLINE	2	nP	nBP	VDCR
POLYMARKER	3	nP	NBP	VDCR
TEXT	4	P,E,S	BP+BE+ BS	VDCR,{0,1}, SR
RESTRICTED TEXT	5	2VDC,P, E,S	2VDC+BP+ BE+BS	++VDCR,VDCR, {0,1},SR
APPEND TEXT	6	E,S	BE+BS	{0,1},SR
POLYGON	7	nP	nBP	VDCR
POLYGON SET	8	n(P,E)	n(BP+BE)	VDCR,{0..3}

Table 7 — Encoding of graphical primitive elements (continued)

Element Class 4	Element Id	Parameter Type	Parameter List Length	Parameter Range
CELL ARRAY	9	3P,3I, E,CLIST	3BP+3BI+ BE+nBCO	VDCR,+IR,++IR, {0,1},COR
GENERALIZED DRAWING PRIMITIVE	10	I,I,nP,D	2BI+nBP+ BS	IR,++IR, VDCR,SR
RECTANGLE	11	2P	2BP	VDCR
CIRCLE	12	P,VDC	BP+BVDC	VDCR, ++VDCR
CIRCULAR ARC POINT	13	3P	3BP	VDCR
CIRCULAR ARC 3 POINT CLOSE	14	3P,E	3BP+BE	VDCR,{0,1}
CIRCULAR ARC CENTRE	15	P,4VDC, VDC	BP+4BVDC+ BVDC	VDCR,VDCR, ++VDCR
CIRCULAR ARC CENTRE CLOSE	16	P,4VDC, VDC,E	BP+4BVDC+ BVDC+BE	VDCR,VDCR, ++VDCR,{0,1}
ELLIPSE	17	3P	3BP	VDCR
ELLIPTICAL ARC	18	3P,4VDC	3BP+4BVDC	VDCR,VDCR
ELLIPTICAL ARC CLOSE	19	3P,4VDC, E	3BP+4BVDC+ BE	VDCR,VDCR, {0,1}
CIRCULAR ARC CENTRE REVERSED	20	P,4VDC, VDC	BP+4BVDC+ BVDC	VDCR,VDCR, ++VDCR
CONNECTING EDGE	21	n/a	0	n/a
HYPERBOLIC ARC	22	3P,4VDC	3BP+4BVDC	VDCR
PARABOLIC ARC	23	3P	3BP	VDCR
NON-UNIFORM B-SPLINE	24	2I,nP, (n+m)R, 2R	2BI+nBP+ (n+m)BR+ 2BR	+IR,VDCR, ++RR, ++RR
NON-UNIFORM RATIONAL B-SPLINE	25	2I,nP, (n+m)R, 2R, nR	2BI+nBP+ (n+m)BR+ 2BR+ nBR	+IR,VDCR, ++RR, ++RR, ++RR
POLYBEZIER	26	IX,4nP(or) (3n+1)P	BIX+4nBP(or) BIX+(3n+1)P	{1,2}, VDCR
POLYSYMBOL	27	IX,nP	BIX+nBP	+IXR,VDCR
BITONAL TILE	28	IX,I, 2CO, SDR,BS	BIX+BIX+ 2BCO+ BSDR,BBS	++IXR,++IR, COR, n/a,BSR
TILE	29	IX,I, I,SDR,BS	BIX+BI+ BI+BSDR+BB S	++IXR,++IR, ++IR,n/a,BSR

Additional description of the elements in Table 7:

Code Description

1 POLYLINE: has a variable parameter list:

P1-Pn: (point) n (X,Y) polyline vertices

2 DISJOINT POLYLINE: has a variable parameter list:

P1-Pn: (point) n (X,Y) line segment endpoints

3 POLYMARKER: has a variable parameter list:

P1-Pn: (point) n (X,Y) marker positions

4 TEXT: has 3 parameters:

P1: (point) text position

P2: (enumerated) final/not-final flag: valid values are

0 not final
1 final

P3: (string) text string

5 RESTRICTED TEXT: has 5 parameters:

P1: (vdc) delta width

P2: (vdc) delta height

P3: (point) text position

P4: (enumerated) final/not-final flag: valid values are

0 not final
1 final

P5: (string) text string

6 APPEND TEXT: has 2 parameters:

P1: (enumerated) final/not-final flag: valid values are

0 not final
1 final

P2: (string) text string

7 POLYGON: has a variable parameter list:

P1-Pn: (point) n (X,Y) polygon vertices

8 POLYGON SET: has a variable parameter list of pairs of values, each of which has the following form:

P(i): (point) (X,Y) polygon vertex

P(i+1): (enumerated) edge out flag, indicating closures and edge visibility: valid values are

0 invisible
1 visible
2 close, invisible
3 close, visible

9 CELL ARRAY: has 8 parameters:

P1: (point) corner point P

P2: (point) corner point Q

P3: (point) corner point R

P4: (integer) nx

P5: (integer) ny

P6: (integer) local colour precision: valid values are 0, 1, 2, 4, 8, 16, 24, and 32. If the value is zero (the 'default colour precision indicator' value), the COLOUR (INDEX) PRECISION for the picture indicates the precision with which the colour list is encoded. If the value is non-zero, the precision with which the colour data is encoded is given by the value.

P7: (enumerated) cell representation mode: valid values are

0 run length list mode
1 packed list mode

P8: (colour list) array of cell colour values.

If the COLOUR SELECTION MODE is 'direct', the values will be direct colour values. If the COLOUR SELECTION MODE is \&'indexed', the values will be indexes into the COLOUR TABLE.

If the cell representation mode is 'packed list', the colour values are represented by rows of values, each row starting on a word boundary. If the cell representation mode is 'run length', the colour list values are represented by rows broken into runs of constant colour; each row starts on a word boundary. Each list item consists of a cell count (integer) followed by a colour value. With the exception of the first run of a row, the integer count of each run immediately follows the colour specifier of the preceding run with no intervening padding.

10 GENERALIZED DRAWING PRIMITIVE: has a variable parameter list:

P1: (integer) GDP identifier

P2: (integer) n, number of points in 'list of points'

P3-P(n+2): (point array) list of points

P(n+3)...: (data record) GDP data record

The parameter P2 is required to determine where the coordinate data ends and the data record begins. Data records are bound as strings in this encoding.

11 RECTANGLE: has 2 parameters:

P1: (point) first corner

P2: (point) second corner

12 CIRCLE: has 2 parameters:

P1: (point) centre of circle

P2: (vdc) radius of circle

13 CIRCULAR ARC 3 POINT: has 3 parameters:

P1: (point) starting point

P2: (point) intermediate point

P3: (point) ending point

14 CIRCULAR ARC 3 POINT CLOSE: has 4 parameters:

P1: (point) starting point

P2: (point) intermediate point

P3: (point) ending point

P4: (enumerated) type of arc closure: valid values are

0 pie closure
1 chord closure

15 CIRCULAR ARC CENTRE: has 6 parameters:

P1: (point) centre of circle

P2: (vdc) delta X for start vector

P3: (vdc) delta Y for start vector

P4: (vdc) delta X for end vector

P5: (vdc) delta Y for end vector

P6: (vdc) radius of circle

16 CIRCULAR ARC CENTRE CLOSE: has 7 parameters:

P1: (point) centre of circle

P2: (vdc) delta X for start vector

P3: (vdc) delta Y for start vector

P4: (vdc) delta X for end vector

P5: (vdc) delta Y for end vector

P6: (vdc) radius of circle

P7: (enumerated) type of arc closure: valid values are

0 pie closure
1 chord closure

17 ELLIPSE: has 3 parameters:

P1: (point) centre of ellipse

P2: (point) endpoint of first conjugate diameter

P3: (point) endpoint of second conjugate diameter

18 ELLIPTICAL ARC: has 7 parameters:

P1: (point) centre of ellipse

P2: (point) endpoint for first conjugate diameter

P3: (point) endpoint for second conjugate diameter

P4: (vdc) delta X for start vector

P5: (vdc) delta Y for start vector

P6: (vdc) delta X for end vector

P7: (vdc) delta Y for end vector

19 ELLIPTICAL ARC CLOSE: has 8 parameters:

P1: (point) centre of ellipse

P2: (point) endpoint for first conjugate diameter

P3: (point) endpoint for second conjugate diameter

P4: (vdc) delta X for start vector

P5: (vdc) delta Y for start vector

P6: (vdc) delta X for end vector

P7: (vdc) delta Y for end vector

P8: (enumerated) type of arc closure: valid values are

0 pie closure
1 chord closure

20 CIRCULAR ARC CENTRE REVERSED: has 6 parameters:

P1: (point) centre of circle

P2: (vdc) delta X for start vector

P3: (vdc) delta Y for start vector

P4: (vdc) delta X for end vector

P5: (vdc) delta Y for end vector

P6: (vdc) radius of circle

21 CONNECTING EDGE: has no parameters

22 HYPERBOLIC ARC: has 7 parameters:

P1: (point) centre point

P2: (point) transverse radius end point

P3: (point) conjugate radius end point

P4: (vdc) start vector x component

P5: (vdc) start vector y component

P6: (vdc) end vector x component

P7: (vdc) end vector y component

23 PARABOLIC ARC: has 3 parameters:

P1: (point) tangent intersection point

P2: (point) start point

P3: (point) end point

24 NON-UNIFORM B-SPLINE: has a variable parameter list:

P1: (integer) spline order (=m)

P2: (integer) number of control points (=n)

P(3)-P(2+n): (points) array of control points

P(3+n)-P(2+2n+m): (real) list of knots, of length n+m.

P(3+2n+m): (real) parameter start value

P(4+2n+m): (real) parameter end value

25 NON-UNIFORM RATIONAL B-SPLINE: has a variable parameter list:

P1: (integer) spline order (=m)

P2: (integer) number of control points (=n)

P(3)-P(2+n): (points) array of control points

P(3+n)-P(2+2n+m): (real) list of knots, of length n+m.

P(3+2n+m): (real) parameter start value

P(4+2n+m): (real) parameter end value

P(5+2n+m)-P(4+3n+m): (real) list of weights, of length n.

26 POLYBEZIER: has a variable parameter list:

P1: (index) continuity indicator: valid values are

1: discontinuous
2: continuous
>2 reserved for registered values

P2-Pn: (point) list of point sequences: each sequence defines a single bezier curve and contains 4 or 3 points according to the continuity indicator values 1 or 2, respectively (if the indicator is 2, the first curve, and only the first, is defined by 4 points).

27 POLYSYMBOL: has a variable parameter list:

P1: (index) symbol index

P2-P(n+1): (point) n symbol position points.

28 BITONAL TILE: has 6 parameters:

P1: (index) compression type: valid values are

0	null background
1	null foreground
2	T6
3	1-dimensional
4	T4 2-dimensional
5	bitmap (uncompressed)
6	run length
>6	reserved for registered values

P2: (integer) row padding indicator: valid values are non-negative integers.

P3: (colour) cell background colour

P4: (colour) cell foreground colour

P5: (structured data record) method-specific parameters, valid values are

[null_SDR], for compression types 1-5,
[(integer: i_I), (integer: 1), (integer: run-count precision)], for type=6,
as defined in the Register, for type>6.

Note 1 See NOTE 17, Table 1, for additional SDR formatting requirements.

P6 (bitstream) compressed cell colour specifiers

29 TILE: has 5 parameters:

P1: (index) compression type: valid values are

0	null background
1	null foreground
2	T6
3	T4 1-dimensional
4	T4 2-dimensional
5	bitmap (uncompressed)
6	run length
>6	reserved for registered values

P2: (integer) row padding indicator: valid values are non-negative integers.

P3: (integer) cell colour precision: valid values are as for the local colour precision of CELL ARRAY for compression types 0 - 5, or any value specified in the Register for compression type>6.

P4 (structured data record) method-specific parameters, valid values are

[null_SDR], for compression types 1-5,
[(integer: i_I), (integer: 1), (integer: run-count precision)], for type=6,
as defined in the Register, for type>6.

NOTE 2 See NOTE 17, Table 1, for additional SDR formatting requirements.

P5: (bitstream) compressed cell colour specifiers

8.7 Attribute elements

Table 8 — Encoding of attribute elements

Element Class 5	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
LINE BUNDLE INDEX	1	IX	BIX	+IXR
LINE TYPE	2	IX	BIX	IXR
LINE WIDTH	3	SS	BSS	++SSR
LINE COLOUR	4	CO	BCO	COR
MARKER BUNDLE INDEX	5	IX	BIX	+IXR
MARKER TYPE	6	IX	BIX	IXR
MARKER SIZE	7	SS	BSS	++SSR
MARKER COLOUR	8	CO	BCO	COR
TEXT BUNDLE INDEX	9	IX	BIX	+IXR
TEXT FONT INDEX	10	IX	BIX	+IXR
TEXT PRECISION	11	E	BE	{0..2}
CHARACTER EXPANSION FACTOR	12	R	BR	++RR
CHARACTER SPACING	13	R	BR	RR
TEXT COLOUR	14	CO	BCO	COR
CHARACTER HEIGHT	15	VDC	BVDC	++VDCR
CHARACTER ORIENTATION	16	4VDC	4BVDC	VDCR
TEXT PATH	17	E	BE	{0..3}
TEXT ALIGNMENT	18	2E, R,R	2BE+ 2BR	{0..4}, {0..6}, 2RR
CHARACTER SET INDEX	19	IX	BIX	+IXR
ALTERNATE CHARACTER SET INDEX	20	IX	BIX	+IXR
FILL BUNDLE INDEX	21	IX	BIX	+IXR
INTERIOR STYLE	22	E	BE	{0..6}
FILL COLOUR	23	CO	BCO	COR
HATCH INDEX	24	IX	BIX	IXR
PATTERN INDEX	25	IX	BIX	+IXR
EDGE BUNDLE INDEX	26	IX	BIX	+IXR
EDGE TYPE	27	IX	BIX	IXR
EDGE COLOUR	29	CO	BCO	COR
EDGE VISIBILITY	30	E	BE	{0,1}
FILL REFERENCE POINT	31	P	BP	VDCR

Table 8 — Encoding of attribute elements (continued)

Element Class 5	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
PATTERN TABLE	32	IX,3I, nx*nyCO	BIX+3BI+ nx*nyBCO	+IXR,+IR, ++IR,COR
PATTERN SIZE	33	4SS	4BSS	SSR
COLOUR TABLE	34	CI,nCD	BCI+nBCD	CIR,CCOR
ASPECT SOURCE FLAGS	35	n(E,E)	n(2BE)	{0..17},{0,1}
PICK IDENTIFIER	36	N	BN	NR
LINE CAP	37	IX,IX	2BIX	+IXR
LINE JOIN	38	IX	BIX	+IXR
LINE TYPE CONTINUATION	39	IX	BIX	+IXR
LINE TYPE INITIAL OFFSET	40	R	BR	++RR
TEXT SCORE TYPE	41	n(IX,E)	nBIX+nBE	IXR,{0,1}
RESTRICTED TEXT TYPE	42	IX	BIX	+IXR
INTERPOLATED INTERIOR	43	IX,2nSS, I,mR,kCO	2BIX+2nBSS+ BI+mBR+kBCO	{1..3},SSR, +IR,RR,COR
EDGE CAP	44	IX,IX	2BIX	+IXR
EDGE JOIN	45	IX	BIX	+IXR
EDGE TYPE CONTINUATION	46	IX	BIX	+IXR
EDGE TYPE INITIAL OFFSET	47	R	BR	++RR
SYMBOL LIBRARY INDEX	48	IX	BIX	+IXR
SYMBOL COLOUR	49	CO	BCO	COR
SYMBOL SIZE	50	E,2VDC	BE+2BVDC	{0..2},VDCR
SYMBOL ORIENTATION	51	4VDC	4BVDC	VDCR

Additional description of the elements in Table 8:

Code Description

1 LINE BUNDLE INDEX: has 1 parameter:

P1: (index) line bundle index

2 LINE TYPE: has 1 parameter:

P1: (index) line type: the following values are standardized:

1 solid
2 dash
3 dot
4 dash-dot
5 dash-dot-dot
>5 reserved for registered values
negative for private use

3 LINE WIDTH: has 1 parameter:

P1: (size specification) line width: see Part 1, subclause 7.1 for its form.

4 LINE COLOUR: has 1 parameter; its form depends on COLOUR SELECTION MODE:

P1: (colour) line colour

5 MARKER BUNDLE INDEX: has 1 parameter:

P1: (index) marker bundle index

6 MARKER TYPE: has 1 parameter:

P1: (index) marker type: the following values are standardized:

1 dot
2 plus
3 asterisk
4 circle
5 cross
>5 reserved for registered values
negative for private use

7 MARKER SIZE: has 1 parameter:

P1: (size specification) marker size: see Part 1, subclause 7.1 for its form.

8 MARKER COLOUR: has 1 parameter; its form depends on COLOUR SELECTION MODE:

P1: (colour) marker colour

9 TEXT BUNDLE INDEX: has 1 parameter:

P1: (index) text bundle index

10 TEXT FONT INDEX: has 1 parameter:

P1: (index) text font index

11 TEXT PRECISION: has 1 parameter:

P1: (enumerated) text precision: valid values are

0 string
1 character
2 stroke

12 CHARACTER EXPANSION FACTOR: has 1 parameter:

P1: (real) character expansion factor

13 CHARACTER SPACING: has 1 parameter:

P1: (real) additional inter-character space

14 TEXT COLOUR: has 1 parameter; its form depends on COLOUR SELECTION MODE:

P1: (colour) text colour

15 CHARACTER HEIGHT: has 1 parameter:

P1: (vdc) character height.

16 CHARACTER ORIENTATION: has 4 parameters:

P1: (vdc) X character up component

P2: (vdc) Y character up component

P3: (vdc) X character base component

P4: (vdc) Y character base component

17 TEXT PATH: has 1 parameter:

P1: (enumerated) text path: valid values are:

0 right
1 left
2 up
3 down

18 TEXT ALIGNMENT: has 4 parameters:

P1: (enumerated) horizontal alignment: valid values are:

0 normal horizontal
1 left
2 centre
3 right
4 continuous horizontal

P2: (enumerated) vertical alignment

0 normal vertical
1 top
2 cap
3 half
4 base
5 bottom
6 continuous vertical

P3: (real) continuous horizontal alignment

P4: (real) continuous vertical alignment

19 CHARACTER SET INDEX: has 1 parameter:

P1: (index) character set index

20 ALTERNATE CHARACTER SET INDEX: has 1 parameter:

P1: (index) alternate character set index

21 FILL BUNDLE INDEX: has 1 parameter:

P1: (index) fill bundle index

22 INTERIOR STYLE: has 1 parameter:

P1: (enumerated) interior style: valid values are

0 hollow
1 solid
2 pattern
3 hatch
4 empty
5 geometric pattern
6 interpolated

23 FILL COLOUR: has 1 parameter; its form depends on COLOUR SELECTION MODE:

P1: (colour) fill colour

24 HATCH INDEX: has 1 parameter

P1: (index) hatch index: the following values are standardized:

1 horizontal
2 vertical
3 positive slope
4 negative slope
5 horizontal/vertical crosshatch
6 positive/negative slope crosshatch
>6 reserved for registered values
negative for private use

25 PATTERN INDEX: has 1 parameter

P1: (index) pattern index

26 EDGE BUNDLE INDEX: has 1 parameter:

P1: (index) edge bundle index

27 EDGE TYPE: has 1 parameter:

P1: (integer) edge type: the following values are standardized:

1 solid
2 dash
3 dot
4 dash-dot
5 dash-dot-dot
>5 reserved for registered values
negative for private use

28 EDGE WIDTH: has 1 parameter:

P1: (size specification) edge width: see part 1, subclause 7.1 for its form.

29 EDGE COLOUR: has 1 parameter; its form depends on COLOUR SELECTION MODE:

P1: (colour) edge colour

30 EDGE VISIBILITY: has 1 parameter:

P1: (enumerated) edge visibility: valid values are

0 off
1 on

31 FILL REFERENCE POINT: has 1 parameter:

P1: (point) fill reference point

32 PATTERN TABLE: has 5 parameters:

P1: (index) pattern table index

P2: (integer) nx, the dimension of colour array in the direction of the PATTERN SIZE width vector

P3: (integer) ny, the dimension of colour array in the direction of the PATTERN SIZE height vector

P4: (integer) local colour precision: valid values are as for the local colour precision parameter of CELL ARRAY.

P5: (colour array) pattern definition

33 PATTERN SIZE: has 4 parameters:

P1: (size specification) pattern height vector, x component: see part 1, subclause 7.1 for its form.

P2: (size specification) pattern height vector, y component: see part 1, subclause 7.1 for its form.

P3: (size specification) pattern width vector, x component: see part 1, subclause 7.1 for its form.

P4: (size specification) pattern width vector, y component: see part 1, subclause 7.1 for its form.

NOTE Pattern size may only be 'absolute' (VDC) in Version 1 and 2 metafiles. In Version 3 and 4 metafiles it may be expressed in any of the modes which can be selected with INTERIOR STYLE SPECIFICATION MODE.

34 COLOUR TABLE: has 2 parameters:

P1: (colour index) starting colour table index

P2: (direct colour list) list of direct colour values (>3-tuples or 4-tuples of

direct colour components (CCO))

35 ASPECT SOURCE FLAGS: has up to 18 parameter-pairs, corresponding to each attribute that may be bundled; each parameter-pair contains the ASF type and the ASF value:

(enumerated) ASF type; valid values are

0 line type ASF
1 line width ASF
2 line colour ASF
3 marker type ASF
4 marker size ASF
5 marker colour ASF
6 text font index ASF
7 text precision ASF
8 character expansion factor ASF
9 character spacing ASF
10 text colour ASF
11 interior style ASF
12 fill colour ASF
13 hatch index ASF
14 pattern index ASF
15 edge type ASF

16 edge width ASF
17 edge colour ASF

(enumerated) ASF value; valid values are

0 individual
1 bundled

36 PICK IDENTIFIER: has 1 parameter:

P1: (name) pick identifier

37 LINE CAP: has 2 parameters:

P1: (index) line cap indicator: the following values are standardized:

1 unspecified
2 butt
3 round
4 projecting square
5 triangle
>5 reserved for registered values

P2: (index) dash cap indicator: valid values are

1 unspecified
2 butt
3 match
>3 reserved for registered values

38 LINE JOIN: has 1 parameter:

P1: (index) line join indicator: the following values are standardized:

1 unspecified
2 mitre
3 round
4 bevel
>4 reserved for registered values

39 LINE TYPE CONTINUATION: has 1 parameter:

P1: (index) continuation mode: the following values are standardized:

1 unspecified
2 continue
3 restart
4 adaptive continue
>4 reserved for registered values

40 LINE TYPE INITIAL OFFSET: has 1 parameter:

P1: (real) line pattern offset

41 TEXT SCORE TYPE: has 1 parameter:

P1-Pn: list of score type, score indicator pairs (index,enumerated): the following values are standardized for the score type:

1 right score
2 left score
3 through score
4 kendot
>4 reserved for registered values

valid values for the score indicators are

0 off
1 on

42 RESTRICTED TEXT TYPE: has 1 parameter:

P1: (index) restriction type: the following values are standardized:

1 basic
2 boxed-cap
3 boxed-all
4 isotropic-cap
5 isotropic-all
6 justified
>6 reserved for registered values

43 INTERPOLATED INTERIOR: has a variable parameter list:

P1: (index) style: valid values are

1 parallel
2 elliptical
3 triangular
>3 reserved for registered values

P2: (2n(size specification)) reference geometry: see part 1, subclause 7.1 for its form.

P3: (integer) number of stages (=m)

P4: (real) array of m stage designators

P5: (colour) array of k colour specifiers: k=3 for triangular, m+1 otherwise.

44 EDGE CAP: has 2 parameters:

P1: (index) edge cap indicator: the following values are standardized:

1 unspecified
2 butt
3 round
4 projected square
5 triangle
>5 reserved for registered values

P2: (index) dash cap indicator: valid values are

1 unspecified
2 butt
3 match
>3 reserved for registered values

45 EDGE JOIN: has 1 parameter:

P1: (index) edge join indicator: the following values are standardized:

1 unspecified
2 mitre
3 round
4 bevel
>4 reserved for registered values

46 EDGE TYPE CONTINUATION: has 1 parameter:

P1: (index) continuation mode: the following values are standardized:

1 unspecified
2 continue
3 restart
4 adaptive continue
>4 reserved for registered values

47 EDGE TYPE INITIAL OFFSET: has 1 parameter:

P1: (real) edge pattern offset

48 SYMBOL LIBRARY INDEX: has 1 parameter:

P1: (index) symbol library index

49 SYMBOL COLOUR: has 1 parameter:

P1: (colour) symbol colour

50 SYMBOL SIZE: has 3 parameters:

P1: (enumerated) scale indicator: valid values are

0 height
1 width
2 both

P2: (vdc) symbol height

P2: (vdc) symbol width

51 SYMBOL ORIENTATION: has 4 parameters:

P1: (vdc) up vector x component

P2: (vdc) up vector y component

P3: (vdc) base vector x component

P4: (vdc) base vector y component

8.8 Escape element

Table 9 — Encoding of escape element

Element Class 6	Element Id	Parameter Type	Parameter List Length	Parameter Range
ESCAPE	1	I,D	BI+BS	IR,SR

Additional description of the element in Table 9:

Code Description

1 ESCAPE: has 2 parameters:

P1: (integer) escape identifier

P2: (data record) escape data record; data records are bound as strings in this encoding.

8.9 External elements

Table 10 — Encoding of external elements

Element Class 7	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
MESSAGE	1	E,SF	BE+BS	{0,1},SR
APPLICATION DATA	2	I,D	BI+BS	IR,SR

Additional description of the elements in Table 10:

Code Description

1 MESSAGE: has 2 parameters:

P1: (enumerated) action-required flag: valid values are

0 no action
1 action

P2: (string fixed) message string

2 APPLICATION DATA: has 2 parameters:

P1: (integer) identifier

P2: (data record) application data record; data records are bound as strings in this encoding.

8.10 Segment control and segment attribute elements

Table 11 — Encoding of segment elements

Element Class 8	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
COPY SEGMENT	1	N,4R, 2VDC, E	BN+4BR+ 2BVDC + BE	NR,RR, VDCR, {0,1}
INHERITANCE FILTER	2	nE,E	(n+1)BE	{0..86},{0,1}
CLIP INHERITANCE	3	E	BE	{0,1}
SEGMENT TRANSFORMATION	4	N,4R, 2VDC	BN+4BR+ 2BVDC	NR,RR, VDCR
SEGMENT HIGHLIGHTING	5	N,E	BN+BE	NR,{0,1}
SEGMENT DISPLAY PRIORITY	6	N,I	BN+BI	NR,++IR
SEGMENT PICK PRIORITY	7	N,I	BN+BI	NR,++IR

Additional description of the elements in Table 11:

Code Description

1 COPY SEGMENT: has 3 parameters:

P1: (name) segment identifier

P2: The next 6 values are components of a transformation matrix consisting of a scaling and rotation portion (2 x 2 R) and a translation portion (2 x 1 VDC). In the binary encoding this is expressed as a 2 x 3 matrix of the form:

a11: (real) x scale component
a12: (real) x rotation component
a21: (real) y rotation component
a22: (real) y scale component
a13: (vdc) x translation component
a23: (vdc) y translation component

P3: (enumerated) segment transformation application: valid values are

0: no
1: yes

2 INHERITANCE FILTER: has two parameters. The first is a list of attribute or group designators. The second is a single setting value.

P1: (enumerated list) list of one or more of:

0 line bundle index
1 line type
2 line width
3 line colour
4 line clipping mode
5 marker bundle index
6 marker type
7 marker size

8 marker colour
9 marker clipping mode
10 text bundle index
11 text font index
12 text precision
13 character expansion factor
14 character spacing
15 text colour
16 character height
17 character orientation
18 text path
19 text alignment
20 fill bundle index
21 interior style
22 fill colour
23 hatch index
24 pattern index
25 edge bundle index
26 edge type
27 edge width
28 edge colour
29 edge visibility
30 edge clipping mode
31 fill reference point
32 pattern size
33 auxiliary colour
34 transparency
35 line attributes
36 marker attributes
37 text presentation and placement attributes
38 text placement and orientation attributes
39 fill attributes
40 edge attributes
41 pattern attributes
42 output control
43 pick identifier
44 all attributes and control
45 all
46 line type asf
47 line width asf
48 line colour asf
49 marker type asf
50 marker size asf
51 marker colour asf
52 text font index asf
53 text precision asf
54 character expansion factor asf
55 character spacing asf
56 text colour asf
57 interior style asf
58 fill colour asf
59 hatch index asf
60 pattern index asf
61 edge type asf
62 edge width asf
63 edge colour asf
64 line asfs
65 marker asfs
66 text asfs
67 fill asfs

68 edge asfs
69 all asfs
70 mitre limit
71 line cap
72 line join
73 line type continuation
74 line type initial offset
75 text score type
76 restricted text type
77 interpolated interior
78 edge cap
79 edge join
80 edge type continuation
81 edge type initial offset
82 symbol library index
83 symbol colour
84 symbol size
85 symbol orientation
86 symbol attributes

P2: (enumerated) setting: valid values are

0 state list
1 segment

3 CLIP INHERITANCE: has 1 parameter

P1: (enumerated) clip inheritance: valid values are

0 state list
1 intersection

4 SEGMENT TRANSFORMATION: has 2 parameters:

P1: (name) segment identifier

P2: The next 6 values are components of a transformation matrix consisting of a scaling and rotation portion (2 x 2 R) and a translation portion (2 x 1 VDC). In the binary encoding this is expressed as a 2 x 3 matrix of the form:

a11: (real) x scale component
a12: (real) x rotation component
a21: (real) y rotation component
a22: (real) y scale component
a13: (vdc) x translation component
a23: (vdc) y translation component

5 SEGMENT HIGHLIGHTING: has 2 parameters:

P1: (name) segment identifier

P2: (enumerated) highlighting: valid values are

0 normal
1 highlighted

6 SEGMENT DISPLAY PRIORITY: has 2 parameters:

P1: (name) segment identifier

P2: (integer) segment display priority: valid values are non-negative integers

7 SEGMENT PICK PRIORITY: has 2 parameters:

P1: (name) segment identifier

P2: (integer) segment pick priority: valid values are non-negative integers

8.11 Application structure descriptor elements

Table 12 — Encoding of application structure descriptor elements

Element Class 9	**Element Id**	**Parameter Type**	**Parameter List Length**	**Parameter Range**
APPLICATION STRUCTURE ATTRIBUTE	1	SF, SDR	BS+BS	SR, SR

Additional description of the elements in Table 12:

Code Description

1 APPLICATION STRUCTURE ATTRIBUTE:has 2 parameters

P1: (string fixed) application structure attribute type

P2: (structured data record) data record

9 Defaults

The following are the defaults for the binary encoding.

REAL PRECISION:	Fixed point; whole part 16 bits; fractional part 16 bits.
INTEGER PRECISION:	16 bits
COLOUR PRECISION:	1 octet (per colour component)
COLOUR INDEX PRECISION:	1 octet
INDEX PRECISION:	16 bits
VDC REAL PRECISION:	Fixed point; whole part 16 bits; fractional part 16 bits.
VDC INTEGER PRECISION:	16 bits
COLOUR VALUE EXTENT:	if COLOUR MODEL is RGB, minimum-colour-value 0,0,0 maximum-colour-value 255,255,255. if COLOUR MODEL is CMYK, minimum-colour-value 0,0,0,0 maximum-colour-value 255,255,255,255. if COLOUR MODEL is CIELUV, CIELAB, or RGB-related colour-scale-first-component 0.0 colour-offset-first-component 0.0 colour-scale-second-component 0.0 colour-offset-second-component 0.0 colour-scale-third-component 0.0 colour-offset-third-component 0.0
NAME PRECISION:	16 bits

10 Profile encoding rules, proforma, and Model Profile

10.1 Encodings

Precisions are defined consistently with the principles of the encodings, not necessarily for inter-encoding translation. Where both considerations might apply, compatibility with the principles of the encoding are considered first and inter-encoding translation second.

10.2 Metafile defaults

Clause 9 addresses all elements which have encoding-dependent default values. While no profile can change these values, an equivalent effect may be achieved by use of the METAFILE DEFAULTS REPLACEMENT element. Profiles may require that a metafile contain a METAFILE DEFAULTS REPLACEMENT element with well-defined content.

10.3 Floating point values

Profiles shall prohibit the values NaN, positive infinity, and negative infinity for ANSI/IEEE 754 floating point numbers (see 6.5). For 32-bit floating point, NaN corresponds to e=255 and f≠0. For 64-bit floating point, NaN corresponds to e=2047 and f≠0. For 32-bit floating point, positive and negative infinity are defined respectively by s=0 and s=1, with e=255 and f=0. For 64-bit floating point, positive and negative infinity are defined respectively by s=0 and s=1, with e=2047 and f=0.

10.4 Profile proforma tables (PPF)

All elements are included in the Profile Proforma (PPF) in Part 1. These include the following elements with specific requirements for binary encoding:

Delimiter elements: no-op

Metafile descriptor elements: INTEGER PRECISION, REAL PRECISION, INDEX PRECISION, COLOUR PRECISION, COLOUR INDEX PRECISION, NAME PRECISION.

Control elements:VDC INTEGER PRECISION, and VDC REAL PRECISION.

The Profile Proforma fragments specifically directed to binary encoding are contained in table 15, table 16 and table 18 of part 1, annex I. These tables, when completed by the author of the profile, contain the normative specifications of the profile.

10.5 Permissible alternative coding representation

The following alternative coding representation shall be permissible in metafiles which conform to this encoding. For the CELL ARRAY element, in the case that a row-alignment octet is required for all rows but the last, in conformance to the first formula of the fourth paragraph of note 11 following Table 1, the last row may be coded with the same data length as the other rows, rather than applying the second formula of note 11 to the last row.

NOTE Using this alternative coding representation would cause, for the case described above, the parameter list length of the element or partition to be greater by one octet, than if the specifications of note 11 were followed precisely. However, this will not affect the starting location of the next element in the metafile.

Annex A
(normative)

Formal grammar

This annex provides explanation of the terminal symbols specified in Annex A of ISO/IEC 8632-1.

Opcodes are encoded as two integers specifying the element class and element identifier. The element classes are listed in Table 2 and the element identifiers in Table 3 to Table 11. The full list of class and element codes is given in Annex C. For example:

<METAFILE VERSION> ::= 1 1 <parameter list length>

<METAFILE DESCRIPTION> ::= 1 2 <parameter list length>

<parameter list length> ::= <integer> {encoded as described in clause 5}

The enumerated types are 16-bit signed integers. The other terminal symbols are described in detail in clause 6. A reference to the relevant tables is given here.

<integer> ::= two's complement integer {See clause 6}

<real> ::= <floating point real> | <fixed point real> {See clause 6}

<vdc value> ::= <integer> | <real>

<string> ::= <length> <character>* {See Table 1}

<string fixed> ::= <length> <character>* {See Table 1}

<character> ::= 8-bit characters, or multiples of 8 bits, depending on the character set {See 6.3}.

<cco value> ::= <unsigned integer> {See Table 1

<colour index> ::= <unsigned integer> {See Table 1}

<colour direct> ::= <color direct v1-v2> {Version 1 and 2 metafiles}
| <color direct v3> {Version 3 and 4 metafiles}

<color direct v1-v2> ::= <red green blue>

<color direct v3> ::= <red green blue>
| <LAB>
| <LUV>
| <CMYK>
| <ABC>

<red green blue> ::= <unsigned integer>(3) {See Table 1}

<LAB> ::= <unsigned integer>(3)

<LUV> ::= <unsigned integer>(3)

<CMYK> ::= <unsigned integer>(4)

<ABC> ::= <unsigned integer>(3)

<viewport point> ::= <integer>(2)|<real>(2)

<vc value> ::= <integer>|<real>

<name> ::= <integer>

<2x2 matrix of reals> ::= <real>(4)

<2x1 matrix of vdcs>	::=	<vdc value>(2)	
<3 x 3 matrix of reals>	::=	<real>(9)	
<data record>	::=	<length> <octet>*	{See Table 1}

Character substitution is not used in this encoding.

The following operands are described in Table 4:

<index precision value>
<integer precision value>
<real precision value>
<colour precision value>
<colour index precision value>
<name precision value>

The production <element name> is encoded in this encoding as an index pair: the first index is the element class code (as in Table 2) and the second is the element id code (as in Table 3 to Table 11).

The following operands are described in Table 6.

<integer VDC precision value>
<real VDC precision value>

The following operand is described in Table 7 (under CELL ARRAY):

<colour list>

The following operand value is described in Table 7 (under CELL ARRAY):

<default col precision indicator>

Annex B
(informative)

Examples

The following simple examples illustrate the use of the binary encoding of the CGM. All precisions used are the default values.

B.1 Example 1 : BEGIN METAFILE 'Example 1'

<table>
<tr><th></th><th colspan="4">15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th></tr>
<tr><td>Header:</td><td>0</td><td colspan="2">1</td><td>10</td></tr>
<tr><td>Length:</td><td colspan="2">9</td><td colspan="2">'E'</td></tr>
<tr><td>Name:</td><td colspan="2">'x'</td><td colspan="2">'a'</td></tr>
<tr><td></td><td colspan="2">'m'</td><td colspan="2">'p'</td></tr>
<tr><td></td><td colspan="2">'l'</td><td colspan="2">'e'</td></tr>
<tr><td></td><td colspan="2">' '</td><td colspan="2">'l'</td></tr>
</table>

B.2 Example 2 : BEGIN PICTURE 'Test'

<table>
<tr><th></th><th colspan="4">15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th></tr>
<tr><td>Header:</td><td>0</td><td colspan="2">3</td><td>5</td></tr>
<tr><td>Length:</td><td colspan="2">4</td><td colspan="2">'T'</td></tr>
<tr><td>Name:</td><td colspan="2">'e'</td><td colspan="2">'s'</td></tr>
<tr><td>(pad)</td><td colspan="2">'t'</td><td colspan="2">0</td></tr>
</table>

B.3 Example 3 : POLYLINE from 0,2 to 1,3 to 2,1 to 0,2

<table>
<tr><th></th><th colspan="3">15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th></tr>
<tr><td>Header:</td><td>4</td><td>1</td><td>16</td></tr>
<tr><td>Point 0,2:</td><td colspan="3">0</td></tr>
<tr><td></td><td colspan="3">2</td></tr>
<tr><td>Point 1,3:</td><td colspan="3">1</td></tr>
<tr><td></td><td colspan="3">3</td></tr>
</table>

Point 2,1:	2
	1
Point 0,2:	0
	2

B.4 Example 4 : TEXT 'Hydrogen' at 0,1

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Header:	4	4	15
Point 0,1:	0		
	1		
Flag:	0		
Length, String:	8	'H'	
String:	'y'	'd'	
	'r'	'o'	
	'g'	'e'	
(pad)	'n'	0	

B.5 Example 5 : Partitioned POLYLINE with 50 points

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Header:	4	1	31
Long (cont.) :	1	120	
Point (1) :	x(1)		
	y(1)		

Point(30):	x(30)	
	y(30)	
Long (cont.) :	0	80
Point (1) :	x(31)	
	y(31)	

Point(50):

x(50)
y(50)

B.6 Example 6 : METAFILE DEFAULT REPLACEMENT linewidth 0.5

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Header:	1	12	6
Line width:	5	3	4
Value:	0.5 (spans 2 words)		

B.7 Example 7 : Application Data # 655 with 10K octets (chars) of data

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Header:	7	2	31
Long (final) :	0	10244	
ID :	655		
Length:	255	0	10240 …
Data record :	length continued	Octet 1	
	Octet 2	Octet 3	

(pad) :	Octet 10240	0

Annex C
(informative)

List of binary encoding metafile element codes

The following list, arranged in the order of the elements given in ISO/IEC 8632-1, indicates the class and element codes associated with each metafile element. These are the codes used in the METAFILE ELEMENTS LIST element.

Delimiter Elements: Class 0

Class	Element Code	Element Name
0	0	no-op
0	1	BEGIN METAFILE
0	2	END METAFILE
0	3	BEGIN PICTURE
0	4	BEGIN PICTURE BODY
0	5	END PICTURE
0	6	BEGIN SEGMENT
0	7	END SEGMENT
0	8	BEGIN FIGURE
0	9	END FIGURE
0	13	BEGIN PROTECTION REGION
0	14	END PROTECTION REGION
0	15	BEGIN COMPOUND LINE
0	16	END COMPOUND LINE
0	17	BEGIN COMPOUND TEXT PATH
0	18	END COMPOUND TEXT PATH
0	19	BEGIN TILE ARRAY
0	20	END TILE ARRAY
0	21	BEGIN APPLICATION STRUCTURE
0	22	BEGIN APPLICATION STRUCTURE BODY
0	23	END APPLICATION STRUCTURE

Metafile Descriptor Elements: Class 1

Class	Element Code	Element Name
1	1	METAFILE VERSION
1	2	METAFILE DESCRIPTION
1	3	VDC TYPE
1	4	INTEGER PRECISION
1	5	REAL PRECISION
1	6	INDEX PRECISION
1	7	COLOUR PRECISION
1	8	COLOUR INDEX PRECISION

1	9	MAXIMUM COLOUR INDEX
1	10	COLOUR VALUE EXTENT
1	11	METAFILE ELEMENT LIST
1	12	METAFILE DEFAULTS REPLACEMENT
1	13	FONT LIST
1	14	CHARACTER SET LIST
1	15	CHARACTER CODING ANNOUNCER
1	16	NAME PRECISION
1	17	MAXIMUM VDC EXTENT
1	18	SEGMENT PRIORITY EXTENT
1	19	COLOUR MODEL
1	20	COLOUR CALIBRATION
1	21	FONT PROPERTIES
1	22	GLYPH MAPPING
1	23	SYMBOL LIBRARY LIST
1	24	PICTURE DIRECTORY

Picture Descriptor Elements : Class 2

Class	Element Code	Element Name
2	1	SCALING MODE
2	2	COLOUR SELECTION MODE
2	3	LINE WIDTH SPECIFICATION MODE
2	4	MARKER SIZE SPECIFICATION MODE
2	5	EDGE WIDTH SPECIFICATION MODE
2	6	VDC EXTENT
2	7	BACKGROUND COLOUR
2	8	DEVICE VIEWPORT
2	9	DEVICE VIEWPORT SPECIFICATION MODE
2	10	DEVICE VIEWPORT MAPPING
2	11	LINE REPRESENTATION
2	12	MARKER REPRESENTATION
2	13	TEXT REPRESENTATION
2	14	FILL REPRESENTATION
2	15	EDGE REPRESENTATION
2	16	INTERIOR STYLE SPECIFICATION MODE
2	17	LINE AND EDGE TYPE DEFINITION
2	18	HATCH STYLE DEFINITION
2	19	GEOMETRIC PATTERN DEFINITION
2	20	APPLICATION STRUCTURE DIRECTORY

Control Elements: Class 3

Class	Element Code	Element Name
3	1	VDC INTEGER PRECISION

3	2	VDC REAL PRECISION
3	3	AUXILIARY COLOUR
3	4	TRANSPARENCY
3	5	CLIP RECTANGLE
3	6	CLIP INDICATOR
3	7	LINE CLIPPING MODE
3	8	MARKER CLIPPING MODE
3	9	EDGE CLIPPING MODE
3	10	NEW REGION
3	11	SAVE PRIMITIVE CONTEXT
3	12	RESTORE PRIMITIVE CONTEXT
3	17	PROTECTION REGION INDICATOR
3	18	GENERALIZED TEXT PATH MODE
3	19	MITRE LIMIT
3	20	TRANSPARENT CELL COLOUR

Graphical Primitive Elements : Class 4

Class	Element Code	Element Name
4	1	POLYLINE
4	2	DISJOINT POLYLINE
4	3	POLYMARKER
4	4	TEXT
4	5	RESTRICTED TEXT
4	6	APPEND TEXT
4	7	POLYGON
4	8	POLYGON SET
4	9	CELL ARRAY
4	10	GENERALIZED DRAWING PRIMITIVE
4	11	RECTANGLE
4	12	CIRCLE
4	13	CIRCULAR ARC 3 POINT
4	14	CIRCULAR ARC 3 POINT CLOSE
4	15	CIRCULAR ARC CENTRE
4	16	CIRCULAR ARC CENTRE CLOSE
4	17	ELLIPSE
4	18	ELLIPTICAL ARC
4	19	ELLIPTICAL ARC CLOSE
4	20	CIRCULAR ARC CENTRE REVERSED
4	21	CONNECTING EDGE
4	22	HYPERBOLIC ARC
4	23	PARABOLIC ARC
4	24	NON-UNIFORM B-SPLINE
4	25	NON-UNIFORM RATIONAL B-SPLINE
4	26	POLYBEZIER
4	27	POLYSYMBOL

4	28	BITONAL TILE
4	29	TILE

Attribute Elements: Class 5

Class	Element Code	Element Name
5	1	LINE BUNDLE INDEX
5	2	LINE TYPE
5	3	LINE WIDTH
5	4	LINE COLOUR
5	5	MARKER BUNDLE INDEX
5	6	MARKER TYPE
5	7	MARKER SIZE
5	8	MARKER COLOUR
5	9	TEXT BUNDLE INDEX
5	10	TEXT FONT INDEX
5	11	TEXT PRECISION
5	12	CHARACTER EXPANSION FACTOR
5	13	CHARACTER SPACING
5	14	TEXT COLOUR
5	15	CHARACTER HEIGHT
5	16	CHARACTER ORIENTATION
5	17	TEXT PATH
5	18	TEXT ALIGNMENT
5	19	CHARACTER SET INDEX
5	20	ALTERNATE CHARACTER SET INDEX
5	21	FILL BUNDLE INDEX
5	22	INTERIOR STYLE
5	23	FILL COLOUR
5	24	HATCH INDEX
5	25	PATTERN INDEX
5	26	EDGE BUNDLE INDEX
5	27	EDGE TYPE
5	28	EDGE WIDTH
5	29	EDGE COLOUR
5	30	EDGE VISIBILITY
5	31	FILL REFERENCE POINT
5	32	PATTERN TABLE
5	33	PATTERN SIZE
5	34	COLOUR TABLE
5	35	ASPECT SOURCE FLAGS
5	36	PICK IDENTIFIER
5	37	LINE CAP
5	38	LINE JOIN
5	39	LINE TYPE CONTINUATION
5	40	LINE TYPE INITIAL OFFSET

5	41	TEXT SCORE TYPE
5	42	RESTRICTED TEXT TYPE
5	43	INTERPOLATED INTERIOR
5	44	EDGE CAP
5	45	EDGE JOIN
5	46	EDGE TYPE CONTINUATION
5	47	EDGE TYPE INITIAL OFFSET
5	48	SYMBOL LIBRARY INDEX
5	49	SYMBOL COLOUR
5	50	SYMBOL SIZE
5	51	SYMBOL ORIENTATION

Escape Element: Class 6

Class	Element Code	Element Name
6	1	ESCAPE

External Elements: Class 7

Class	Element Code	Element Name
7	1	MESSAGE
7	2	APPLICATION DATA

Segment Elements: Class 8

Class	Element Code	Element Name
8	1	COPY SEGMENT
8	2	INHERITANCE FILTER
8	3	CLIP INHERITANCE
8	4	SEGMENT TRANSFORMATION
8	5	SEGMENT HIGHLIGHTING
8	6	SEGMENT DISPLAY PRIORITY
8	7	SEGMENT PICK PRIORITY

Application Structure Descriptor Elements: Class 9

Class	Element Code	Element Name
9	1	APPLICATION STRUCTURE ATTRIBUTE

ICS 35.140

Price based on 68 pages

INTERNATIONAL STANDARD

ISO/IEC 8632-4

Second edition
1999-12-01

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 4: **Clear text encoding**

Technologies de l'information — Infographie — Métafichier de stockage et de transfert des informations de description d'images —

Partie 4: Codage en clair des textes

Reference number
ISO/IEC 8632-4:1999(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 734 10 79
E-mail copyright@iso.ch
Web www.iso.ch

Printed in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 8632 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 8632-4 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics and image processing*.

This second edition cancels and replaces the first edition (ISO/IEC 8632-4:1992), which has been technically revised. Note that the previous edition of ISO/IEC 8632-4, published in 1992, was a first edition but second edition was indicated by error on its cover page and in the foreword.

ISO/IEC 8632 consists of the following parts, under the general title *Information technology — Computer graphics — Metafile for the storage and transfer of picture description information*:

— *Part 1: Functional specification*

— *Part 3: Binary encoding*

— *Part 4: Clear text encoding*

Annex A forms a normative part of this part of ISO/IEC 8632. Annex B is for information only.

NOTE In previous editions of ISO/IEC 8632, Part 2 defined a Character Encoding. Part 2 was withdrawn in 1998, due to its lack of implementation and use.

Introduction

0.1 Purpose of the clear text encoding

The Clear Text Encoding of the Computer Graphics Metafile (CGM) provides a representation of the Metafile syntax that is easy to type, edit and read. It allows a metafile to be edited with any standard text editor, using the internal character code of the host computer system.

0.2 Primary objectives

a) Human editable: The Clear Text Encoding should be able to be hand edited or, if desired, hand constructed.

b) Human friendly: The Clear Text Encoding should be easy and natural for people to read and edit. Although what is easiest and most natural is a subjective judgment that varies among users, contributing factors such as ease of recognition, ease of remembering, avoidance of ambiguity, and prevention of mistyping have all been considered.

c) Machine readable: The Clear Text Encoding should be able to be parsed by software.

d) Suitable for use in a wide variety of editors: The Clear Text Encoding should not have any features that make it difficult to edit in normal text editors.

e) Facilitate interchange between diverse systems: The Clear Text Encoding should be encoded in such a way as to maximize the set of systems which can utilize it. No assumptions should be made as to word size or arithmetic modes used to interpret the metafile.

f) Use standardized abbreviations as much as possible: Where language encoding of other graphics standards have established standard abbreviations, or where common practice in the data processing and graphics industries has established well known abbreviations, these abbreviations are used. In accordance with the principle of “least astonishment”, this approach should minimize the time needed to learn to use this encoding.

0.3 Secondary objectives

Because the other CGM encoding (the CGM Binary Encoding) is targeted toward CPU efficiency and information density, these objectives are considered of secondary importance for the CGM Clear Text Encoding.

0.4 Relationship to other International Standards

The set of characters required to implement the Clear Text Encoding is a subset of those included in national versions of ISO/IEC 646. Any character set that can be mapped to and from that subset may be used to implement the encoding.

For certain elements, the CGM defines value ranges as being reserved for registration. The values and their meanings will be defined using the established procedures (see ISO/IEC 8632-1, 6.12.)

Information technology — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 4: **Clear text encoding**

1 Scope

This part of ISO/IEC 8632 specifies a clear text encoding of the Computer Graphics Metafile. For each of the elements specified in ISO/IEC 8632-1, a clear text encoding is specified. Allowed abbreviations are specified. The overall format of the metafile and the means by which comments may be interspersed in the metafile is specified.

This encoding of the CGM allows metafiles to be created and maintained in a form which is simple to type, easy to edit and convenient to read.

2 Conformance

Conformance of metafiles to ISO/IEC 8632 is defined in terms of profiles. A metafile conforms to this encoding if it conforms to a profile and meets the following criteria:

— Each metafile element described in this part shall be encoded in the manner described in this part of this International Standard and a profile.

— Metafile elements which are not defined in Part 1 or in this encoding are all encoded using the GENERALIZED DRAWING PRIMITIVE or ESCAPE metafile elements as appropriate. According to the profile rules of Part 1 (see clause 9, subclause 9.5.2.8), such elements shall either be profile defined or registered, in order that the profile be valid. Inclusion of private elements is not permissible in a valid profile of ISO/IEC 8632 and this encoding.

— Values of index parameters, which are used as enumeration selectors from lists of implicitly defined attribute values, shall either be standard, registered, or profile defined. The standard and registered values are all non-negative, and the profile-defined shall be negative. Use of private, implicitly-defined negative index values which are not profile defined is not permissible in a valid profile of ISO/IEC 8632 and this encoding.

— Values specified as being "reserved for registered values" shall not be used unless their meaning has been registered or standardized.

— All characters in the metafile shall be from the enumerated character repertoire (see 6.1), except for those within a parameters of type String and String Fixed, eligible parameters within specific data records, and format effectors as described in 6.1.

— Numbers shall be formatted as defined in 6.3.1 and 6.3.2.

— Inclusion of non-graphical data in the metafile shall be accomplished with the APPLICATION DATA element or with the APPLICATION STRUCTURE ATTRIBUTE element.

See clause 9 for additional conformance information about this encoding.

3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 8632. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO/IEC 8632 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 646:1991, *Information technology — ISO 7-bit coded character set for information interchange.*

ISO 2022:1986, *Information processing — ISO 7-bit and 8-bit coded character sets — Code extension techniques.*

4 Notational conventions

Unbracketed strings are terminals of this grammar. They appear in valid Clear Text data streams exactly as indicated in the specifications of this part, except for allowable variations on case and null characters described below.

Bracketed strings are either non-terminals (with further productions given), character symbol names (such as COMMA), or parameters of the CGM element in the form <x:y> (see ISO/IEC 8632-1 for further explanation of these items).

"::=" is read as "becomes" or "is realized as".

<...>* = star closure (0 or more occurrences).
<...>+ = plus closure (1 or more occurrences).
<...>o = optional (exactly 0 or 1 occurrences).
<x:y> = parameter type x with meaning y
<x|y> = exactly one of x or y
{...} = a comment (not part of the production)
<...>(n) = exactly n occurrences, n=0,1,2,...

SPACES are used for readability in the grammar description; SPACES in the actual metafile are indicated through the separator productions given below.

The metasymbols used in describing the grammar do not appear in the actual metafile.

5 Entering and leaving the metafile environment

5.1 Generic clear text and instantiations

The Clear Text Encoding is described in a generic fashion that permits it to be used with any character set capable of representing those characters enumerated in the Character Repertoire (see part 1, 6.7.3.2). An instantiation of the Clear Text Encoding is specified by defining the character set and coding technique to be used (for example, standard national character sets based on ISO/IEC 646, non-standard character sets such as EBCDIC, etc).

It is recommended that an instantiation of the Clear Text Encoding bound to the standard national character set based on ISO/IEC 646 be used in order to maximize portability of Clear Text metafiles between diverse systems. This also provides an encoding which can be incorporated into an ISO 2022 text environment as a complete code, to permit intermixing of text and graphics for applications which place a high priority on human readability.

5.2 Implicitly entering the metafile environment

The Clear Text coding environment may be entered implicitly by agreement between the interchanging parties. This is suitable only if there is not to be any interchange with services using other coding techniques, and if it is known by prior agreement which instantiation of the syntax is being used.

5.3 Designating and invoking the CGM coding environment from ISO 2022

For interchange with services using the code extension techniques of ISO 2022, the (standard national version) ISO/IEC 646 instantiation of the CGM Clear Text Encoding may be designated and invoked from the ISO 2022 environment by the following escape sequence:

ESC 2/5 F

where ESC is the bit combination 1/11, and F refers to a bit combination that will be assigned by the ISO Registration Authority for ISO 2375.

The first bit combination occurring after this escape sequence will then represent the beginning of a CGM metafile element or one of the "soft separators" or "null characters" defined below.

The following escape sequence may be used to return to the ISO 2022 coding environment:

ESC 2/5 4/0

This not only returns to the ISO 2022 coding environment, but also restores the designation and invocation of coded character sets to the state that existed prior to entering the ISO/IEC 646 CGM coding environment with the ESC 2/5 F sequence. (The terms "designation" and "invocation" are defined in ISO 2022.)

It is permissable to make transitions between ISO 2022 and the metafile environment between pictures in the metafile as well as between metafiles.

The state of the metafile interpreter and the state of the ISO 2022 environment are maintained separately and not stacked.

The state of the metafile interpreter before BEGIN METAFILE or after END METAFILE is undefined, and sending a picture without a preceding BEGIN METAFILE and metafile descriptor is nonconforming interchange.

6 Metafile format

A metafile in the Clear Text Encoding consists of a stream of characters forming a series of elements, each of which starts with an element name and ends with one of the element delimiters, either the SLASH character (also known as SLANT or SOLIDUS) or the SEMICOLON character. These characters do not act as element delimiters when occurring within the bounds of a string parameter, as defined below.

6.1 Character repertoire

In order to achieve objective (e) of sub-clause 0.2, the character repertoire of the Clear Text Encoding will be limited to those characters enumerated below, except for string parameters, which may contain any characters from the repertoire described in 4.7.3.2 ISO/IEC 8632-1.

- Upper-case characters:
  "A", "B", "C", "D", "E", "F", "G", "H", "I",
  "J", "K", "L", "M", "N", "O", "P", "Q", "R",
  "S", "T", "U", "V", "W", "X", "Y", "Z"
- Lower-case characters:
  "a", "b", "c", "d", "e", "f", "g", "h", "i",
  "j", "k", "l", "m", "n", "o", "p", "q", "r",
  "s", "t", "u", "v", "w", "x", "y", "z"
- Digits:
  "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"
- " " (SPACE character)
- "+" (PLUS character)
- "-" (MINUS character)
- "#" (NUMBER SIGN)
- ";" (SEMICOLON character)

- "/" (SLASH, SLANT, or SOLIDUS character)
- "(" (LEFT or OPEN PARENTHESIS character)
- ")" (RIGHT or CLOSE PARENTHESIS character)
- "," (COMMA character)
- "." (DECIMAL POINT or PERIOD character)
- "'" (APOSTROPHE or SINGLE QUOTE character)
- """ (DOUBLE QUOTE character)
- "_" (UNDERSCORE character)
- "$" (DOLLAR SIGN or CURRENCY symbol)
- "%" (PERCENT SIGN character)

Lower-case characters are considered to be the same as upper-case characters, when occurring outside of string parameters. Any combination of lower-case and upper-case characters may be used within an element or enumerated parameter name.

The UNDERSCORE and DOLLAR SIGN symbols are defined as “null characters” within this encoding. They may appear anywhere within the metafile, and are mandated to have no effect on parsing (outside of string parameters). They are available for the generator or editor of the metafile to use in enhancing readability of tokens.

EXAMPLE:

The following are all equivalent:

linetype, LINETYPE, LineType, line_type, $LINETYPE, L_I_N_E$T_Y_P_E;

similarly, the following are all equivalent:

123456, $123456, 123_456, $123_456, $12$34$56.

Those control characters that are format effectors (BACKSPACE, CARRIAGE RETURN, LINEFEED, NEWLINE, HORIZONTAL TAB, VERTICAL TAB, and FORMFEED) are permitted in the metafile, but are treated as SPACE characters (that is, as soft delimiters) by the metafile interpreter whenever they occur outside of string parameters. They may be used to assist in formatting the metafile to improve its readability. The effect of such format effectors within string parameters is as defined in ISO/IEC 8632-1.

A metafile written in the Clear Text Encoding is considered to be non-conforming if it includes characters other than those listed in the repertoire and the format effectors (outside of string parameters). Implementation-dependent extensions which require use of characters other than the above should be embedded in the string parameters of the ESCAPE, MESSAGE, or APPLICATION DATA elements, or in comments.

The code set of the characters is not fixed by this part of ISO/IEC 8632. In order to accomplish the objective of editability, it is permitted to encode the Clear Text Encoding using the character set codes native to the system. It is presumed that standard conversion facilities can be used in translating Clear Text CGM files from one system’s character set codes to another, consistent with the treatment of other text files being transferred between systems. It is recommended that the ISO/IEC 646 codes be used to encode Clear Text metafiles for transport between diverse systems.

Null characters or format effectors outside of text strings which do not exist in the target system’s encoding may be dropped in such translation, and lower-case letters translated to upper case as necessary, without altering the information content of the metafile. Likewise, the two statement delimiter characters are interchangeable and may be changed in such a translation without affecting the information content of the metafile. The two string delimiter characters are interchangeable, but any translation shall correctly handle the possible occurrence of either string delimiter character within the string parameter.

6.2 Separators

6.2.1 Element separators

<TERM> ::= <OPTSEP> <SLASH **|** SEMICOLON> <OPTSEP>

The SEMICOLON and SLASH characters may be used interchangeably to delimit elements in a Clear Text metafile. These elements do not, however, terminate an element when they occur within a string parameter, as described below.

The elements of the metafile are not terminated by the ends of records, as indicated by control characters such as CR (carriage return) or LF (linefeed). Multiple elements may exist on one line, and any element may extend over multiple lines.

6.2.2 Parameter separators

The following productions are used in the Clear Text Encoding for parameter separators:

<SEPCHAR>	::=	<SPACE | CARRIAGE RETURN | LINEFEED | HORIZONTAL TAB | VERTICAL TAB | FORMFEED>
<SOFTSEP>	::=	<SEPCHAR>+
<OPTSEP>	::=	<SEPCHAR>*
<HARDSEP>	::=	<OPTSEP> <COMMA> <OPTSEP>
<SEP>	::=	<SOFTSEP> | <HARDSEP>

Most commands require a SOFTSEP after the element name (e.g., at least one space). This permits element names to be formed from a mixture of alpha and numeric characters.

The separator between parameters is usually a SEP. This format permits omission of parameters. (Two consecutive COMMAs indicate an omitted parameter.)

Since the enclosing APOSTROPHE or DOUBLE QUOTE character sufficiently delineates string parameters, and the statement delimiter SLASH also sets off the data on either side of it, the separators between these characters and adjacent parameters or element names are optional (OPTSEP).

SEPCHAR characters are not permitted within a name (element or enumerated type), or within the representation of a numeric parameter. Any place where a SEPCHAR is permitted (other than inside a string parameter), an arbitrary number of SEPCHARs may be used.

6.2.3 Comments in the metafile

Comments may be included in a Clear Text metafile, to enhance its readability and usefulness. Some uses of comments might be to document hand-edited changes to the metafile, or as “notes to one’s self” made while reading a metafile. To include other forms of nongraphical information in the metafile, it is suggested that the APPLICATION DATA element be used. If it is desired to convert a Clear Text metafile to one of the other encodings, comments may be either dropped or converted to APPLICATION DATA elements.

Comments are encoded as a series of printing characters and <SEPCHAR>s surrounded by “%” (PERCENT SIGN) characters. The text of the comment may not include this comment delimiter character.

Comments may be included any place that a separator may be used, and are equivalent to a <SOFTSEP>; they may be replaced by a SPACE character in parsing, without affecting the meaning of the metafile.

6.3 Encoding of parameter types

6.3.1 Integer-bound types

Integers, integer coordinates, indexes, names, and the components of direct colour parameters are all bound to signed integers, indicated in the encoding as I.

The data types UI8 and UI32 of ISO/IEC 8632-1 are bound to non-negative values of signed integers, also indicated in this encoding as I.

<I> ::= <decimal integer> | <based integer>

<decimal integer> ::= <sign>o <digit>+

<sign> ::= <PLUS SIGN> | <MINUS SIGN>

<based integer> ::= <sign>o <base> <NUMBER SIGN> <extended digit>+

<base> ::= 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16

<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<extended digit> ::= <digit> | A | B | C | D | E | F | a | b | c | d | e | f

The null characters are permitted within numbers, but are not shown in the productions for simplicity.

A decimal integer has an optional sign and at least one digit. If the sign appears, it immediately precedes the number with no intervening SPACE (or other <SEPCHAR>) characters allowed.

A based integer has an optional sign, a base (an unsigned integer in the range 2..16 inclusive, represented in base 10), a “#”, and a string of one or more extended digits. If the sign appears, it immediately precedes the number with no intervening SPACE (or other <SEPCHAR>) characters allowed. The extended digits used shall be valid for the base named or the metafile is not conforming; e.g., for base 8 the digits “8”, “9”, etc. are not valid, for base 2 only the digits “0” and “1” are valid, and so forth. Case is not significant for the extended digits.

If the sign is omitted for either form, the number is considered non-negative.

Both the base and the <extended digit>+ are interpreted as unsigned numbers, and the final result negated if a MINUS SIGN preceded the number. No assumptions should be made as to the word size of the metafile interpreter, or whether the underlying arithmetic is one’s complement, two’s complement, or sign-magnitude. For example, -1 would be encoded in hexadecimal as -16#1, -16#0001, etc. rather than 16#FFFF. Of course, metafiles may be created utilizing prior knowledge of the intended target machine, but any such assumptions will limit the portability of the metafile and are discouraged.

EXAMPLE:

> 0, 007, -5, +123_456
>
> The following are equivalent: 65535, 16#FFFF, 16#ffff, 8#177777, 2#1111_1111_1111_1111
>
> The following are equivalent: -32_768, -16#8000, -8#100000, -2#10000000_00000000

Interpretation of numerically bound parameters will be “free field”, that is, there is an implied radix point to the right of the rightmost digit, and neither leading nor trailing spaces are significant. Leading zeroes are not significant.

6.3.2 Real-bound types

Reals and real coordinates are bound to real numbers, indicated in the encoding as R. These are written as either explicit-point numbers or scaled-real numbers (or decimal integers, where appropriate).

<R> ::= < explicit-point number > |
< scaled-real number> |
< decimal integer >

< explicit-point number > ::= <sign>o
<
<<digit>+ <PERIOD> <digit>*>
|
<<digit>* <PERIOD> <digit>+>
>

< scaled-real number>	::=	<body> < E | e > <exponent>
<body>	::=	<explicit-point number> | <decimal integer>
<exponent>	::=	<decimal integer>

The interpretation of the scaled-real number is the same as standard scientific notation (similar to FORTRAN “E” format), where the number represented by <body> is multiplied by 10 taken to the power <exponent>.

There shall be at least one digit in an explicit-point number and in the body of a scaled-real number, which in the case of a single-digit number may appear on either side of the radix point. It is recommended but not required that there be at least one digit before the radix point, for numbers with only a fractional part. Zero may be encoded as “0.”, “.0”, “0.0”, “0”, etc., although the second form is not recommended.

In the case of a scaled-real number (one where an “E” or “e” appears), at least one digit shall appear in the <exponent>. No SPACE or other <SEPCHAR> characters are permitted between the <body> and the “E” or “e”, or between the “E” or “e” and the <exponent>.

The interpretation of parameters bound to this data type will be “free field”, that is, if there is an explicit radix point, it sets the radix point of the internal representation, and neither leading nor trailing spaces or zeroes are significant. If the radix point is omitted, it is implied to be at the right of the rightmost digit of the explicit-point number or of the <body> of the scaled-real number. Thus, decimal I-format numbers are permitted to appear in a conforming metafile for parameters bound to real numbers when there is no fractional part.

For real numbers in all formats, the only permitted base of representation is base 10.

If the <sign> (“+” or “-“) is omitted, the number is assumed to be non-negative. If the sign is present, it immediately precedes the body of the number, with no SPACE (or other <SEPCHAR>) characters allowed between it and the leftmost digit or radix point of the body of the number.

COMMA, SPACE and other <SEPCHAR> characters are not permitted within a number, but <NULLCHAR> characters are permitted (and have no effect on parsing).

EXAMPLE:

3.14159

7.853982E-7

271828e-5

42

-.04321 (not recommended form)

-0.043_21

42E2

6.3.3 String-bound types

STRING parameters, both String (S) and String Fixed (SF) types, are represented as character strings immediately surrounded by a matched pair of either APOSTROPHE (SINGLE QUOTE) or DOUBLE QUOTE characters.

If an APOSTROPHE is required in a string delimited with APOSTROPHES, it is represented by two adjacent APOSTROPHES at that position in the string. Likewise, if a DOUBLE QUOTE character is required in a string delimited with DOUBLE QUOTE characters, it is represented by two adjacent DOUBLE QUOTE characters. For example, the following are equivalent

TEXT 0 0 FINAL "Murphy's Law: ""If it can go wrong, it will.""";

TEXT 0 0 FINAL 'Murphy''s Law: "If it can go wrong, it will."';

DATA RECORD data type is represented as a string in this encoding.

STRING and DATA RECORD parameters are indicated in the element encodings as S. STRING FIXED is indicated in the element encodings as SF.

6.3.4 Enumerated types

Enumerated types are bound to names, similarly to the element names.

6.3.5 Derived types

In addition to the I, R, and S parameter formats, the following abbreviations are used as shorthand for the productions shown.

VDC ::= <I> | <R>
{ coordinate data, depending on VDC TYPE }

<RED GREEN BLUE> ::= <I:RED> <SEP> <I:GREEN> <SEP> <I:BLUE>

<L A B> ::= <I:L><SEP><I:A><SEP><I:B>

<L U V> ::= <I:L><SEP><I:U><SEP><I:V>

<CYAN MAGENTA YELLOW BLACK>

::= <I:CYAN><SEP><I:MAGENTA>
<SEP><I:YELLOW><SEP><I:BLACK>

<A B C> ::= <I:A><SEP><I:B><SEP><I:C><

K ::= <I> | <DIRECTCOLR>
{colour specifier, depending on COLOUR SELECTION MODE}

<DIRECTCOLR> ::= <RED GREEN BLUE> {if COLOUR MODEL is RGB}
| <L A B> {if COLOUR MODEL is CIELAB}
| <L U V> {if COLOUR MODEL is CIELUV}
| <CYAN MAGENTA YELLOW BLACK>
{if COLOUR MODEL is CMYK}
| <A B C> {if COLOUR MODEL is RGB-related}<

POINTREC ::= <VDC> <SEP> <VDC>

P ::= <POINTREC>
| < <LEFT PAREN> <OPTSEP> <POINTREC> <OPTSEP>
<RIGHT PAREN> >

POINT in VDC space. Parentheses are optional. If they are used, they shall group two VDC numbers. The parenthesized form is intended to aid readability of the metafile. If there are not two numbers in each parenthesized group, the metafile is non-conforming.

V ::= <VDC> | <R>

This corresponds to the SS data type of ISO/IEC 8632-1. It is used for line width, marker size, fill interiors, and edge width, as well as some parameters of user line definition. The resolution of this parameter type to VDC or R depends on the corresponding SPECIFICATION MODE element.

See part 1, 7.1.

N ::= <I> {name}

VC ::= <R> | <I> {viewport coordinate data}

The abstract parameter type VC, a single VC value, is either a real or an integer, depending on the declaration of the picture descriptor element DEVICE VIEWPORT SPECIFICATION MODE. When DEVICE VIEWPORT SPECIFICATION MODE is 'fraction of display surface', the value is real.

When DEVICE VIEWPORT SPECIFICATION MODE is 'millimetres with scale factor' or 'physical device coordinates', the value is integer.

VPOINTREC ::= <VC><SEP><VC>

VP ::= <VPOINTREC>
| < <LEFT PAREN> <OPTSEP> <VPOINTREC> <OPTSEP> <RIGHT PAREN> >

POINT in viewport coordinate space. Parentheses are optional. If they are used, they shall group two real or integer numbers, depending on DEVICE VIEWPORT SPECIFICATION MODE. The parenthesized form is intended to aid readability of the metafile.

TM ::= <<R: a11><SEP><R: a12><SEP>
<R: a21><SEP><R: a22><SEP>
<VDC: a13><SEP><VDC: a23>>

6.3.6 Bitstream datatype

The parameters of type Bitstream, of tile array elements, shall be represented as follows. The bits taken 4 at a time are represented by a single hexidecimal digit in the Clear Text metafile. Null characters, SPACE, and format effector characters may be interspersed in the stream for readability. For example, a space character every 4 digits and a newline every 60 digits would provide well-formatted output. Bitstream datatype is indicated by “B” in the element definitions. If the cell colour precision indicated by the cell colour precision parameter is 2^N - 1, for some integer N greater than 0, then N bits are used to encode the BS parameter. Otherwise the next-largest bit width, N+1, such that 2^N - 1<cell colour precision<2^{N+1} -1, is used to encode the BS parameter.

6.3.7 Structured data record operands

The structured data record (SDR) of part 1 of this International Standard is composed entirely of other standardized datatypes (including SDR itself) in a structure which is self-defining. SDR is encoded by encoding each of the component operands according to the normal encoding rules for its corresponding data type. The string of characters comprising the encoded operands is then delimited, as is an operand of type String in this part, by matching APOSTROPHE or matching QUOTE characters. The fact that SDR can contain operands of type SDR means that the “strings” can be nested. Nested SDR operands shall be delimited by alternating QUOTE and APOSTROPHE delimiters at successive levels of nesting. Adjacent SDR operands at the same level of nesting shall have at least one SEP character separating them. Within any encoded member, there shall be a <SEP> between the 'data type index’ and the 'data element count’, and there shall be a <SEP> between the 'data element count’ and the list of data elements. If the list of data elements contains more than one item, then adjacent items shall have a <SEP> between them.

6.4 Forming names

The approach was taken of selecting abbreviations for words used to name elements and enumeration types in the CGM, and concatenating them in order.

6.4.1 Words deleted

ANNOUNCER	INDICATOR
AREA	NORMALIZED
BUNDLE	REPLACEMENT
CIRCULAR	SELECTION
CURVE	SPECIFICATION
FACTOR	TANGENTIAL

6.4.2 Words added

INCREMENTAL

6.4.3 Words used unabbreviated

ABSTRACT	FILTER	PICK
ACTION	FINAL	PIE
ALL	FONT	PLUS
ARC	FORCED	POLYBEZIER
ARRAY	FRACTION	POLYGON
AXIS	GKSM	PRIVATE
BASE	GLYPH	REAL
BASIC	HALF	REGION
BEVEL	HATCH	RIGHT
BIT	HEIGHT	ROUND
BITONAL	HOLLOW	SAVE
BODY	INDEX	SCALED
BOTH	INDEXED	SCORE
BOTTOM	INTEGER	SET
BUNDLED	INTERSECTION	SHAPE
CAP	JOIN	SHIELD
CELL	LEFT	SIZE
CHORD	LIMIT	SOLID
CIRCLE	LINE	START
CLIP	LIST	STRING
CLOSE	LOCUS	STROKE
CODING	MARKER	STYLE
COPY	MATRIX	SYMBOL
DATA	MESSAGE	TABLE
DEFAULTS	METRIC	TEXT
DIRECT	MITRE	THEN
DOWN	MODE	TILE
DRAWING	NAME	TRANSPARENCY
EDGE	NEW	TYPE
ELLIPSE	NO	UP
EMPTY	NOT	VALUE
END	OFF	VDC
ESCAPE	OFFSET	VERSION
FIGURE	ON	WIDTH
FILL	OUTPUT	3 (numeral three)
	PATH	

6.4.4 Abbreviations

ABSOLUTE	ABS
ALIGNMENT	ALIGN
ALTERNATE	ALT
APPEND	APND

APPLICATION	APPL
APPLICATION STRUCTURE	APS
ASPECT SOURCE FLAG(S)	ASF
ATTRIBUTE(S)	ATTR
AUXILIARY	AUX
BACKGROUND	BACK
BEGIN	BEG
CALIBRATION	CALIB
CENTRE	CTR
CHARACTER	CHAR
CLIPPING	CLIP
COLOUR	COLR
COMPOUND	COMPO
CONNECTING	CONN
CONTEXT	CONT
CONTINUATION	CONT
CONTINUOUS	CONT
CONTROL	CTRL
COORDINATE(S)	COORD
DEFINITION	DEF
DESCRIPTION	DESC
DEVICE	DEV
DIRECTORY	DIR
DISPLAY	DISP
DISJOINT	DISJT
EIGHT	8
ELEMENT	ELEM
ELLIPTICAL	ELLIP
EXPANSION	EXPAN
EXTENDED	EXTD
EXTENT	EXT
GENERALIZED	GEN
GENERALIZED DRAWING PRIMITIVE	GDP
GEOMETRIC	GEO
HIGHLIGHTING	HIGHL
HYPERBOLIC	HYPERB
IDENTIFIER	ID
INCREMENTAL	INCR
INDIVIDUAL	INDIV
INHERITANCE	INH
INITIAL	INIT
INTERIOR	INT
INTERPOLATED	INTERP
INVISIBLE	INVIS
LIBRARY	LIB
MAPPING	MAP
MAXIMUM	MAX

METAFILE	MF
MILLIMETRE	MM
NON-UNIFORM B-SPLINE	NUB
NON-UNIFORM RATIONAL B-SPLINE	NURB
NORMAL	NORM
ORIENTATION	ORI
PARABOLIC	PARAB
PATTERN	PAT
PHYSICAL	PHY
PICTURE	PIC
PLACEMENT	PLACEM
PRESENTATION	PRES
PRIMITIVE(S)	PRIM
PRIORITY	PRI
PROPERTIES	PROP
PROTECTION	PROT
POINT	PT
POLYLINE	LINE
POLYMARKER	MARKER
POLYSYMBOL	SYMBOL
PRECISION	PREC
RECTANGLE	RECT
REFERENCE	REF
REPRESENTATION	REP
RESTORE	RES
RESTRICTED	RESTR
REVERSED	REV
SCALING	SCALE
SEGMENT	SEG
SEVEN	7
SPACING	SPACE
STATELIST	STLIST
TRANSFORMATION	TRAN
TRANSPARENT	TRANSP
TWO	2
VIEWPORT	VP
VISIBILITY	VIS
VISIBLE	VIS

6.4.5 The derived element names

Metafile Element	Element Name	Notes
BEGIN METAFILE	BEGMF	
END METAFILE	ENDMF	
BEGIN PICTURE	BEGPIC	
BEGIN PICTURE BODY	BEGPICBODY	
END PICTURE	ENDPIC	

BEGIN SEGMENT	BEGSEG	
END SEGMENT	ENDSEG	
BEGIN FIGURE	BEGFIGURE	
END FIGURE	ENDFIGURE	
BEGIN PROTECTION REGION	BEGPROTREGION	
END PROTECTION REGION	ENDPROTREGION	
BEGIN COMPOUND LINE	BEGCOMPOLINE	
END COMPOUND LINE	ENDCOMPOLINE	
BEGIN COMPOUND TEXT PATH	BEGCOMPOTEXTPATH	
END COMPOUND TEXT PATH	ENDCOMPOTEXTPATH	
BEGIN TILE ARRAY	BEGTILEARRAY	
END TILE ARRAY	ENDTILEARRAY	
BEGIN APPLICATION STRUCTURE	BEGAPS	
BEGIN APPLICATION STRUCTURE BODY	BEGAPSBODY	
END APPLICATION STRUCTURE	ENDAPS	
METAFILE VERSION	MFVERSION	
METAFILE DESCRIPTION	MFDESC	
VDC TYPE	VDCTYPE	
INTEGER PRECISION	INTEGERPREC	
REAL PRECISION	REALPREC	
INDEX PRECISION	INDEXPREC	
COLOUR PRECISION	COLRPREC	
COLOUR INDEX PRECISION	COLRINDEXPREC	
MAXIMUM COLOUR INDEX	MAXCOLRINDEX	
COLOUR VALUE EXTENT	COLRVALUEEXT	
METAFILE ELEMENT LIST	MFELEMLIST	
METAFILE DEFAULTS REPLACEMENT	BEGMFDEFAULTS	(1)
	ENDMFDEFAULTS	
FONT LIST	FONTLIST	
CHARACTER SET LIST	CHARSETLIST	
CHARACTER CODING ANNOUNCER	CHARCODING	
NAME PRECISION	NAMEPREC	
MAXIMUM VDC EXTENT	MAXVDCEXT	
SEGMENT PRIORITY EXTENT	SEGPRIEXT	
COLOUR MODEL	COLRMODEL	
COLOUR CALIBRATION	COLRCALIB	
FONT PROPERTIES	FONTPROP	
GLYPH MAPPING	GLYPHMAP	
SYMBOL LIBRARY LIST	SYMBOLLIBLIST	
PICTURE DIRECTORY	PICDIR	
SCALING MODE	SCALEMODE	
COLOUR SELECTION MODE	COLRMODE	
LINE WIDTH SPECIFICATION MODE	LINEWIDTHMODE	
MARKER SIZE SPECIFICATION MODE	MARKERSIZEMODE	
EDGE WIDTH SPECIFICATION MODE	EDGEWIDTHMODE	

VDC EXTENT	VDCEXT	
BACKGROUND COLOUR	BACKCOLR	
DEVICE VIEWPORT	DEVVP	
DEVICE VIEWPORT SPECIFICATION MODE	DEVVPMODE	
DEVICE VIEWPORT MAPPING	DEVVPMAP	
LINE REPRESENTATION	LINEREP	
MARKER REPRESENTATION	MARKERREP	
TEXT REPRESENTATION	TEXTREP	
FILL REPRESENTATION	FILLREP	
EDGE REPRESENTATION	EDGEREP	
INTERIOR STYLE SPECIFICATION MODE	INTSTYLEMODE	
LINE AND EDGE TYPE DEFINITION	LINEEDGETYPEDEF	
HATCH STYLE DEFINITION	HATCHSTYLEDEF	
GEOMETRIC PATTERN DEFINITION	GEOPATDEF	
APPLICATION STRUCTURE DIRECTORY	APSDIR	
VDC INTEGER PRECISION	VDCINTEGERPREC	
VDC REAL PRECISION	VDCREALPREC	
AUXILIARY COLOUR	AUXCOLR	
TRANSPARENCY	TRANSPARENCY	
CLIP RECTANGLE	CLIPRECT	
CLIP INDICATOR	CLIP	
LINE CLIPPING MODE	LINECLIPMODE	
MARKER CLIPPING MODE	MARKERCLIPMODE	
EDGE CLIPPING MODE	EDGECLIPMODE	
NEW REGION	NEWREGION	
SAVE PRIMITIVE CONTEXT	SAVEPRIMCONT	
RESTORE PRIMITIVE CONTEXT	RESPRIMCONT	
PROTECTION REGION INDICATOR	PROTREGION	
GENERALIZED TEXT PATH MODE	GENTEXTPATHMODE	
MITRE LIMIT	MITRELIMIT	
TRANSPARENT CELL COLOUR	TRANSPCELLCOLR	
POLYLINE	LINE	(2)
	INCRLINE	
DISJOINT POLYLINE	DISJTLINE	(2)
	INCRDISJTLINE	
POLYMARKER	MARKER	(2)
	INCRMARKER	
TEXT	TEXT	
RESTRICTED TEXT	RESTRTEXT	
APPEND TEXT	APNDTEXT	
POLYGON	POLYGON	(2)
	INCRPOLYGON	
POLYGON SET	POLYGONSET	(2)
	INCRPOLYGONSET	
CELL ARRAY	CELLARRAY	

GENERALIZED DRAWING PRIMITIVE	GDP	
RECTANGLE	RECT	
CIRCLE	CIRCLE	
CIRCULAR ARC 3 POINT	ARC3PT	
CIRCULAR ARC 3 POINT CLOSE	ARC3PTCLOSE	
CIRCULAR ARC CENTRE	ARCCTR	
CIRCULAR ARC CENTRE CLOSE	ARCCTRCLOSE	
ELLIPSE	ELLIPSE	
ELLIPTICAL ARC	ELLIPARC	
ELLIPTICAL ARC CLOSE	ELLIPARCCLOSE	
CIRCULAR ARC CENTRE REVERSED	ARCCTRREV	
CONNECTING EDGE	CONNEDGE	
HYPERBOLIC ARC	HYPERBARC	
PARABOLIC ARC	PARABARC	
NON-UNIFORM B-SPLINE	NUB	
NON-UNIFORM RATIONAL B-SPLINE	NURB	
POLYBEZIER	POLYBEZIER	
POLYSYMBOL	SYMBOL	(2)
	INCRSYMBOL	
BITONAL TILE	BITONALTILE	
TILE	TILE	
LINE BUNDLE INDEX	LINEINDEX	
LINE TYPE	LINETYPE	
LINE WIDTH	LINEWIDTH	
LINE COLOUR	LINECOLR	
MARKER BUNDLE INDEX	MARKERINDEX	
MARKER TYPE	MARKERTYPE	
MARKER SIZE	MARKERSIZE	
MARKER COLOUR	MARKERCOLR	
TEXT BUNDLE INDEX	TEXTINDEX	
TEXT FONT INDEX	TEXTFONTINDEX	
TEXT PRECISION	TEXTPREC	
CHARACTER EXPANSION FACTOR	CHAREXPAN	
CHARACTER SPACING	CHARSPACE	
TEXT COLOUR	TEXTCOLR	
CHARACTER HEIGHT	CHARHEIGHT	
CHARACTER ORIENTATION	CHARORI	
TEXT PATH	TEXTPATH	
TEXT ALIGNMENT	TEXTALIGN	
CHARACTER SET INDEX	CHARSETINDEX	
ALTERNATE CHARACTER SET INDEX	ALTCHARSETINDEX	
FILL BUNDLE INDEX	FILLINDEX	
INTERIOR STYLE	INTSTYLE	
FILL COLOUR	FILLCOLR	
HATCH INDEX	HATCHINDEX	
PATTERN INDEX	PATINDEX	

EDGE BUNDLE INDEX	EDGEINDEX
EDGE TYPE	EDGETYPE
EDGE WIDTH	EDGEWIDTH
EDGE COLOUR	EDGECOLR
EDGE VISIBILITY	EDGEVIS
FILL REFERENCE POINT	FILLREFPT
PATTERN TABLE	PATTABLE
PATTERN SIZE	PATSIZE
COLOUR TABLE	COLRTABLE
ASPECT SOURCE FLAGS	ASF
PICK IDENTIFIER	PICKID
LINE CAP	LINECAP
LINE JOIN	LINEJOIN
LINE TYPE CONTINUATION	LINETYPECONT
LINE TYPE INITIAL OFFSET	LINETYPEINITOFFSET
TEXT SCORE TYPE	TEXTSCORETYPE
RESTRICTED TEXT TYPE	RESTRTEXTTYPE
INTERPOLATED INTERIOR	INTERPINT
EDGE CAP	EDGECAP
EDGE JOIN	EDGEJOIN
EDGE TYPE CONTINUATION	EDGETYPECONT
EDGE TYPE INITIAL OFFSET	EDGETYPEINITOFFSET
SYMBOL LIBRARY INDEX	SYMBOLLIBINDEX
SYMBOL COLOUR	SYMBOLCOLR
SYMBOL SIZE	SYMBOLSIZE
SYMBOL ORIENTATION	SYMBOLORI
ESCAPE	ESCAPE
MESSAGE	MESSAGE
APPLICATION DATA	APPLDATA
COPY SEGMENT	COPYSEG
INHERITANCE FILTER	INHFILTER
CLIP INHERITANCE	CLIPINH
SEGMENT TRANSFORMATION	SEGTRAN
SEGMENT HIGHLIGHTING	SEGHIGHL
SEGMENT DISPLAY PRIORITY	SEGDISPPRI
SEGMENT PICK PRIORITY	SEGPICKPRI
APPLICATION STRUCTURE ATTRIBUTE	APSATTR

NOTE 1 This element is implemented by a pair of Clear Text element names in the metafile, one to begin defaults replacement and the second to end defaults replacement.

NOTE 2 These elements have point list parameters, the points of which may be represented either with absolute or incremental coordinates. For each of these elements there are two possible Clear Text element names, one corresponding to absolute coordinate representation and one to incremental coordinate representation — the element name used shall correspond to the coordinate representation of point list.

7 Encoding the CGM elements

The defaults for the elements are as given in clause 8 of ISO/IEC 8632-1.

This clause specifies some of the constraints on parameter values. The specifications are not exhaustive, for example such constraints as the non-collinearity of text vectors are not stated. All parameter value and other element state constraints of ISO/IEC 8632-1, including those of the formal grammars, shall apply to metafiles encoded according to this part.

All productions given below which do not appear in the table in subclause 6.4.5 are merely "syntactic shorthand" for describing element productions, and may not appear by themselves in the metafile. For example, CELLROW is a handy way to describe a piece of the CELL ARRAY element, and is not a primitive in itself.

7.1 Encoding delimiter elements

```
BEGIN METAFILE                ::=  BEGMF
                                     <OPTSEP>
                                     <SF:NAME>
                                   <TERM>

END METAFILE                  ::=  ENDMF <TERM>

BEGIN PICTURE                 ::=  BEGPIC
                                     <OPTSEP>
                                     <SF:PICTURENAME>
                                   <TERM>

BEGIN PICTURE BODY            ::=  BEGPICBODY <TERM>

END PICTURE                   ::=  ENDPIC <TERM>

BEGIN SEGMENT                 ::=  BEGSEG
                                     <SOFTSEP>
                                     <N:SEGID>
                                   <TERM>

END SEGMENT                   ::=  ENDSEG  <TERM>

BEGIN FIGURE                  ::=  BEGFIGURE <TERM>

END FIGURE                    ::=  ENDFIGURE <TERM>

BEGIN PROTECTION REGION       ::=  BEGPROTREGION
                                     <SOFTSEP>
                                     <I:REGIONINDEX> {positive}
                                   <TERM>

END PROTECTION REGION         ::=  ENDPROTREGION<TERM>

BEGIN COMPOUND LINE           ::=  BEGCOMPOLINE<TERM>

END COMPOUND LINE             ::=  ENDCOMPOLINE<TERM>

BEGIN COMPOUND TEXT PATH      ::=  BEGCOMPOTEXTPATH<TERM>

END COMPOUND TEXT PATH        ::=  ENDCOMPOTEXTPATH<TERM>

BEGIN TILE ARRAY              ::=  <BEGTILEARRAY>
                                     <SOFTSEP>
                                     <P:POSITION>
                                     <SEP>
                                     <0 | 90 | 180 | 270>
```

```
                                   <SEP>
                                   <90 | 270>
                                   <SEP>
                                   <I:TILESPERPATH> {positive}
                                   <SEP>
                                   <I:TILESPERLINE> {positive}
                                   <SEP>
                                   <I:CELLSPERTILEPATH> {positive}
                                   <SEP>
                                   <I:CELLSPERTILELINE> {positive}
                                   <SEP>
                                   <R:CELLSPACING> {positive}
                                   <SEP>
                                   <R:LINESPACING> {positive}
                                   <SEP>
                                   <I:PATHOFFSET> {non-negative}
                                   <SEP>
                                   <I:LINEOFFSET> {non-negative}
                                   <SEP>
                                   <I:CELLSPERPATH> {positive}
                                   <SEP>
                                   <I:CELLSPERLINE> {positive}
                               <TERM>

END TILE ARRAY              ::= ENDTILEARRAY<TERM>

BEGIN APPLICATION STRUCTURE    ::= BEGAPS
                                   <SOFTSEP>
                                   <SF:APSID>
                                   <SEP>
                                   <SF:APSTYPE>
                                   <SEP>
                                   <STLIST | APS>
                               <TERM>

BEGIN APPLICATION STRUCTURE BODY
                            ::=  BEGAPSBODY<TERM>

END APPLICATION STRUCTURE
                            ::=  ENDAPS<TERM>
```

7.2 Encoding metafile descriptor elements

```
METAFILE VERSION            ::= MFVERSION
                                   <SOFTSEP>
                                   <I:VERSION>
                                      {1=Version 1, 2=Version 2, 3=Version 3, 4=Version 4}
                               <TERM>

METAFILE DESCRIPTION        ::= MFDESC
                                   <OPTSEP>
                                   <SF:DESCRIPTION>
                               <TERM>

VDC TYPE                    ::= VDCTYPE
                                   <SOFTSEP>
                                   < INTEGER | REAL >
                               <TERM>

INTEGER PRECISION           ::= INTEGERPREC
                                   <SOFTSEP>
                                   <I:MININT>
```

```
                                  <SEP>
                                  <I:MAXINT>
                              <TERM>
```

The most negative and most positive integers (base 10) are given. These parameters are interpreted independently of all precisions currently set.

```
REAL PRECISION            ::=  REALPREC
                                  <SOFTSEP>
                                  <R:MINREAL>
                                  <SEP>
                                  <R:MAXREAL>
                                  <SEP>
                                  <I:DIGITS>
                               <TERM>
```

The parameters of this element are interpreted independently of all precisions currently set. The MINREAL and MAXREAL are signed real numbers giving the representable range of numbers. The DIGITS parameter gives the minimum number of DECIMAL DIGITS of accuracy assumed, and is of key importance in preventing roundoff error when the incremental forms of output primitives are used.

NOTE 1 This choice of format was patterned after the floating_point_constraint of the Ada programming language.

```
INDEX PRECISION           ::=  INDEXPREC
                                  <SOFTSEP>
                                  <I:MININT>
                                  <SEP>
                                  <I:MAXINT>
                               <TERM>
```

The most negative and most positive integers (base 10) are given. These parameters are interpreted independently of all precisions currently set.

```
COLOUR PRECISION          ::=  COLRPREC
                                  <SOFTSEP>
                                  <I:MAXCOMPONENT>
                               <TERM>
```

NOTE 2 Colour direct values are 3*I or 4*I, depending on COLOUR MODEL. COLOUR PRECISION gives a single integer range, 0..MAXCOMPONENT, within which each of the three or four components is contained. The parameters are interpreted independently of any currently set precisions.

```
COLOUR INDEX PRECISION    ::=  COLRINDEXPREC
                                  <SOFTSEP>
                                  <I:MAXINT>
                               <TERM>
```

The smallest colour index is 0. The most positive integer that may occur in a colour index parameter is given. This parameter is interpreted independently of all precisions currently set. See MAXCOLRINDEX.

```
MAXIMUM COLOUR INDEX      ::=  MAXCOLRINDEX
                                  <SOFTSEP>
                                  <I:MAXINDEXVALUE>
                               <TERM>
```

The MAXINDEXVALUE shall be less than or equal to the <MAXINT> parameter of COLRINDEXPREC.

```
COLOUR VALUE EXTENT       ::=  COLRVALUEEXT
                                  <SOFTSEP>
                                  <RGBCOLRMAP>
                                  |
                                  <LABCOLRMAP>
                                  |
```

```
                                            <LUVCOLRMAP>
                                            |
                                            <CMYKCOLRMAP>
                                            |
                                            <ABCCOLRMAP>
                                        <TERM>

<RGBCOLRMAP>                  ::=       <RED GREEN BLUE:BLACK>
                                            <SEP>
                                            <RED GREEN BLUE:WHITE>

<LABCOLRMAP>                  ::=       <R:COLRSCALE1> <SEP> <R:COLROFFSET1>
                                            <SEP>
                                            <R:COLRSCALE2> <SEP> <R:COLROFFSET2>
                                            <SEP>
                                            <R:COLRSCALE3> <SEP> <R:COLROFFSET3>

<LUVCOLRMAP>                  ::=       <LABCOLRMAP>

<CMYKCOLRMAP>                 ::=       <CYAN MAGENTA YELLOW BLACK:WHITE>
                                            <SEP>
                                            <CYAN MAGENTA YELLOW BLACK:BLACK>

<ABCCOLRMAP>                  ::=       <LABCOLRMAP>

METAFILE ELEMENT LIST         ::=       MFELEMLIST
                                            <OPTSEP>
                                            <S:ELEMENTNAMES>
                                        <TERM>
```

The string parameter consists of a list of element names separated by <SEP>. In addition, the words DRAWINGPLUS, DRAWINGSET, VERSION2, EXTDPRIM, VERSION2GKSM, VERSION3, and VERSION4 may be used in this string. See Part 1, 7.3.11 for the rules on which of these codes may be used in which metafile versions, and Part 1, 6.3.2 for the meanings of these pseudo-codes.

```
METAFILE DEFAULTS REPLACEMENT
                              ::=         BEGMFDEFAULTS <TERM>
                                            <ELEMENT>+
                                            ENDMFDEFAULTS <TERM>
```

Between the two bracketing elements, applicable CGM elements shall use the same format as described elsewhere in this section.

```
FONT LIST                     ::=       FONTLIST
                                            <OPTSEP>
                                            <SF:FONTNAME>
                                            < <SEP> <SF:FONTNAME> >*
                                        <TERM>

CHARACTER SET LIST            ::=       CHARSETLIST
                                            <SOFTSEP>
                                            <CHARSETDESIGNATOR>
                                            < <SEP> <CHARSETDESIGNATOR> >*
                                        <TERM>

CHARSETDESIGNATOR             ::=       < STD94 |
                                            STD96 |
                                            STD94MULTIBYTE |
                                            STD96MULTIBYTE |
                                            COMPLETECODE >
                                            < <OPTSEP> | <HARDSEP> >
                                        <SF:TAIL>
```

```
CHARACTER CODING ANNOUNCER
                          ::=  CHARCODING
                                 <SOFTSEP>
                                 < BASIC7BIT |
                                   BASIC8BIT |
                                   EXTD7BIT |
                                   EXTD8BIT >
                               <TERM>

NAME PRECISION            ::=  NAMEPREC
                                 <SOFTSEP>
                                 <I:MININT>
                                 <SEP>
                                 <I:MAXINT>
                               <TERM>

MAXIMUM VDC EXTENT        ::=  MAXVDCEXT
                                 <SOFTSEP>
                                 <P:FIRSTCORNER>
                                 <SEP>
                                 <P:SECONDCORNER>
                               <TERM>

SEGMENT PRIORITY EXTENT   ::=  SEGPRIEXT
                                 <SOFTSEP>
                                 <I:MINSEGPRI> {non-negative}
                                 <SEP>
                                 <I:MAXSEGPRI> {non-negative}
                               <TERM>

COLOUR MODEL              ::=  COLRMODEL
                                 <SOFTSEP>
                                 <I:MODELINDEX>
                                 {1=RGB, 2=CIELAB,
                                 3=CIELUV, 4=CMYK, 5=RGB-related,
                                 >5, reserved for registered values}
                               <TERM>

COLOUR CALIBRATION        ::=  COLRCALIB
                                 <SOFTSEP>
                                 <I:CALIBSELECTION>
                                   {1=unspecified, 2=reference white only,
                                   3=reference white, matrix1,
                                   4=reference white, matrix1, lookup tables,
                                   5=reference white, matrix1, lookup tables, matrix2,
                                   6=reference white, matrix1, matrix2,
                                   7=lookup tables, matrix2,
                                   8=matrix2,
                                   9=reference white, grid locations + grid values,
                                   >9, reserved for registered values}
                                 <SEP>
                                 <R:XN> <SEP> <R:YN> <SEP> <R:ZN>
                                 <SEP>
                                 <RGBCALIBDATA>
                                 <SEP>
                                 <CMYKCALIBDATA>
                               <TERM>

<RGBCALIBDATA>            ::=  <RGBCALIBMATRIX>
                                 <SEP>
                                 <ABCTRANMATRIX>
```

```
                                          <SEP>
                                          <I:LUTLENGTH> {n>=0}
                                          <RLUTENTRY>(n)
                                          <GLUTENTRY>(n)
                                          <BLUTENTRY>(n)

<RGBCALIBMATRIX>                  ::=   <R: XR> <SEP> <R: XG> <SEP> <R: XB> <SEP>
                                          <R: YR> <SEP> <R: YG> <SEP> <R: YB> <SEP>
                                          <R: ZR> <SEP> <R: ZG> <SEP> <R: ZB>

<ABCTRANMATRIX>                   ::=   <R: RA> <SEP> <R: RB> <SEP> <R: RC> <SEP>
                                          <R: GA> <SEP> <R: GB> <SEP> <R: GC> <SEP>
                                          <R: BA> <SEP> <R: BB> <SEP> <R: BC>

<RLUTENTRY>                       ::=   <SEP> <I:R> <SEP> <I:RPRIME>

<GLUTENTRY>                       ::=   <SEP> <I:G> <SEP> <I:GPRIME>

<BLUTENTRY>                       ::=   <SEP> <I:B> <SEP> <I:BPRIME>

<CMYKCALIBDATA>                   ::=   <I:GRIDTABLELENGTH> {n>=0}
                                          <<SEP> <DIRECTCOLR:CMYKGRIDLOCATION>>(n)
                                          <XYZGRIDLOCATION>(n)

<XYZGRIDLOCATION>                 ::=   <SEP> <R:CIEXYZX> <SEP>
                                          <R:CIEXYZY> <SEP> <R:CIEXYZZ>

FONT PROPERTIES                   ::=   FONTPROP
                                          <SOFTSEP>
                                          <PROPERTY3TUPLE>
                                          <<SEP> <PROPERTY3TUPLE>>*
                                        <TERM>

<PROPERTY3TUPLE>                  ::=   <I:PROPERTYINDICATOR>
                                            {1=font index, 2=standard version,
                                            3=design source, 4=font family, 5=posture,
                                            6=weight, 7=proportionate width,
                                            8=included glyph collections,
                                            9=included glyphs, 10=design size,
                                            11=minimum size, 12=maximum size,
                                            13=design group, 14=structure,
                                            >14, reserved for registered values}
                                          <SEP> <I:PRIORITY>
                                          <SEP> <SDR:PRIORITYVALUERECORD>

<SDR:PRIORITYVALUERECORD>
                                  ::=  <<SEP><I: i_IX><SEP><I: 1><SEP><I: FONTINDEX>>
                                  | <<SEP><I: i_I><SEP><I: 1><SEP><I:STANDARDVERSION>>
                                  | <<SEP><I: i_SF><SEP><I: 1><OPTSEP><SF:DESIGNSOURCE>>|
                                  | <SEP><I: i_SF><SEP><I: 1><OPTSEP><SF:FONTFAMILY>>
                                  | <<SEP><I: i_IX><SEP><I: 1><SEP><I:POSTURE>>
                                  | <<SEP><I: i_IX><SEP><I: 1><SEP><I:WEIGHT>>
                                  | <<SEP><I: i_IX><SEP><I: 1><SEP><I:PROPORTIONATEWIDTH>>
                                  | <<SEP><I: i_IX><SEP><I: n><SEP><INCLUDGLYPHCOLLECT>(n)>
                                  | <<SEP><I: i_UI32><SEP><I: m><SEP><INCLUDGLYPHS>(m)>
                                  | <<SEP><I: i_R><SEP><I: 1><SEP><R:DESIGNSIZE>>
                                  | <<SEP><I: i_R><SEP><I: 1><SEP><R:MINIMUMSIZE>>
                                  | <<SEP><I: i_R><SEP><I: 1><SEP><R:MAXIMUMSIZE>>
                                  | <<SEP><I: i_UI8><SEP><I: 3><SEP><DESIGNGROUP>>
                                  | <<SEP><I: i_IX><SEP><I: 1><SEP><I:STRUCTURE>>
```

NOTE 3 i_XX in the above denotes the integer value of the 'data type designator' for data type “XX” as assigned in annex C of ISO/IEC 8632-1. For example i_IX represents the designator for data type IX, which is assigned the value 11.

See 6.3.7 for complete SDR delimiting and formatting requirements.

```
<I:FONTINDEX> {positive}

<I:STANDARDVERSION> {positive}

<I:POSTURE>
                                        {0=not applicable, 1=upright,
                                        2=oblique, 3=back slanted oblique,
                                        4=italic, 5=back slanted italic, 6=other,
                                        >6 reserved for registered values}
<I:WEIGHT>
                                        {0=not applicable, 1=ultra light,
                                        2=extra light, 3=light,
                                        4=semi light, 5=medium,
                                        6=semi bold, 7=bold,
                                        8=extra bold, 9= ultra bold,
                                        >9 reserved for registered values}

<I:PROPORTIONATEWIDTH>
                                        {0=not applicable, 1=ultra condensed,
                                        2=extra condensed, 3=condensed,
                                        4=semi condensed, 5=medium,
                                        6=semi expanded, 7=expanded,
                                        8=extra expanded, 9=ultra expanded,
                                        >9 reserved for registered values}

<INCLUDGLYPHCOLLECT>        ::=   <I:CHARSETINDEX> {positive}

<INCLUDGLYPH>                     ::=   <I:AFII32BITIDENTIFIER>

<R:DESIGNSIZE> {positive}

<R:MINIMUMSIZE> {positive}

<R:MAXIMUMSIZE> {positive}

<I:STRUCTURE>
                                        {0=undefined or not applicable,
                                        1=solid, 2=outline,
                                        >2 reserved for registered values}

<DESIGNGROUP>               ::=   <I:CLASS> <SEP> <I:SUBCLASS>
                                      <SEP> <I:SPECIFICGROUP>

GLYPH MAPPING               ::=   GLYPHMAP
                                      <SOFTSEP>
                                      <I:CHARSETINDEX> {positive}
                                      <SEP>
                                      <BASISSET>
                                      <SEP>
                                      <I:OCTETSPERCODE> {this defines m>0}
                                      <SEP>
                                      <I:GLYPHSOURCE>
                                        {1=afii registry of 4-byte identifiers,
                                        >1 reserved for registered values}
                                      <SEP>
                                      <SDR:CODEGLYPHASSOCIATION>
                                  <TERM>
```

```
<BASISSET>                       ::=   <CHARSETDESIGNATOR>
                                           {see CHARACTER SET LIST}

<SDR:CODEGLYPHASSOCIATION>       ::=   <
                                           <I:i_UI8><SEP><I:2n>
                                           < <SEP><I:RUNCOUNT><SEP><I:CODE> >(n)
                                       >
                                       <SEP>
                                       <
                                           <I:i_UI32><SEP><I:n>
                                           < <SEP><I:AFII32BITIDENTIFIER> >(n)
                                       >
```

NOTE 4 i_UI8 and i_UI32 in the above respectively denote the integer value of the 'data type designator' for data type UI8 and UI32 as assigned in annex C of ISO/IEC 8632-1.

NOTE 5 The code and glyphname lists are encoded in a runlength form. The runlength syntax in the code list is explicit: each entry is of the form (N, CODE), N=1..255. If N=1, then only the single specified code is defined. If N>1, then a sequence of codes is defined. The base code of the sequence is the explicitly specified one, and each successive code is 1 greater than the previous pair. N is the number of codes in the sequence, and is limited to 255 per sequence (for uniformity of results across encodings). The list of glyph names is parallel to the code list. The glyph names are associated with the corresponding codes, and when there is a run longer than 1 in the codes, there is also a run longer than 1 in the glyph names. Each glyph name in a run is 1 greater than its predecessor.

NOTE 6 See 6.3.7 for complete SDR delimiting and formatting requirements.

NOTE 7 The first (enumerated) parameter selects the location data type ([ldt]) for the subsequent PICLOCATION and APSDIRLOCATION parameters. These are integer-bound parameters, and the [ldt] selects one of three unsigned integer precisions. The values of PICLOCATION are the offsets measured in characters from the first character of the metafile to the first character of the associated BEGIN PICTURE element. The values of APSDIRLOCATION are the offsets measured in characters from the start of the metafile to the first character of the APPLICATION STRUCTURE DIRECTORY element of the associated picture. For the purpose of measuring offsets, a character is a generic unit as defined in subclause 5.1. For string parameters, a character represents a single byte, regardless of the ISO 2022 coding environment in effect for the string parameter (See 6.7.3.2 ISO/IEC 8632-1:1992).

NOTE 8 Editing of clear-text encoded Version 4 metafiles may invalidate PICLOCATION and APSDIRLOCATION values. If so, these values must be recomputed. Also, translation of clear text metafiles between computing environments may invalidate PICLOCATION and APSDIRLOCATION values as different environments represent line endings differenly."

```
SYMBOL LIBRARY LIST              ::=   SYMBOLLIBLIST
                                           <OPTSEP>
                                           <SF:SYMBOLNAME>
                                           <<SEP><SF:SYMBOLNAME>>*
                                       <TERM>

PICTURE DIRECTORY                ::=   PICDIR
                                           <SOFTSEP>
                                           <UI8 | UI16 | UI32>
                                           <<SEP><PICDIRTRIPLE>>+
                                       <TERM>

PICDIRTRIPLE                     ::=   <SF:PICID>
                                           <SEP>
                                           <I:PICLOCATION>
                                           <SEP>
                                           <I:APSDIRLOCATION>
```

NOTE 9 The first (enumerated) parameter selects the location data type ([ldt]) for the subsequent PICLOCATION and APSDIRLOCATION parameters. These are integer-bound parameters, and the [ldt] selects one of three unsigned integer precisions. The values of PICLOCATION are the offsets measured in characters from the first character of the metafile to the first character of the associated BEGIN PICTURE element. The values of APSDIRLOCATION are the offsets measured in characters from the start of the metafile to the first character of the APPLICATION STRUCTURE DIRECTORY element of the associated picture. For the purpose of measuring offsets, a character is a generic unit as defined in Subclause 5.1. For string parameters, a character represents a single byte, regardless of the ISO 2022 coding environment in effect for the string parameter (See 6.7.3.2 ISO/IEC 8632-1:1992).

NOTE 10 Editing of clear-text encoded Version 4 metafiles may invalidate PICLOCATION and APSDIRLOCATION values. If so, these values must be recomputed. Also, translation of clear text metafiles between computing environments may invalidate PICLOCATION and APSDIRLOCATION values as different environments represent line endings differenly.

7.3 Encoding picture descriptor elements

```
SCALING MODE                 ::=  SCALEMODE
                                      <SOFTSEP>
                                      < ABSTRACT | METRIC >
                                      <SEP>
                                      <R:SCALEFACTOR>
                                  <TERM>

COLOUR SELECTION MODE        ::=  COLRMODE
                                      <SOFTSEP>
                                      < INDEXED | DIRECT >
                                  <TERM>

LINE WIDTH SPECIFICATION MODE:    ::= LINEWIDTHMODE
                                      <SOFTSEP>
                                      <ABS | SCALED | FRACTIONAL | MM>
                                  <TERM>

MARKER SIZE SPECIFICATION MODE
                             ::=    MARKERSIZEMODE
                                      <SOFTSEP>
                                      <ABS | SCALED | FRACTIONAL | MM>
                                  <TERM>

EDGE WIDTH SPECIFICATION MODE
                             ::=    EDGEWIDTHMODE
                                      <SOFTSEP>
                                      <ABS | SCALED | FRACTIONAL | MM>
                                  <TERM>

VDC EXTENT                   ::=  VDCEXT
                                      <SOFTSEP>
                                      <P:FIRSTCORNER>
                                      <SEP>
                                      <P:SECONDCORNER>
                                  <TERM>

BACKGROUND COLOUR            ::=  BACKCOLR
                                      <SOFTSEP>
                                      <DIRECTCOLR>
                                  <TERM>

DEVICE VIEWPORT              ::=  DEVVP
                                      <SOFTSEP>
                                      <VP:FIRSTCORNER>
                                      <SEP>
                                      <VP:SECONDCORNER>
                                  <TERM>

DEVICE VIEWPORT SPECIFICATION MODE
                             ::=    DEVVPMODE
                                      <SOFTSEP>
                                      <FRACTION | MM | PHYDEVCOORD>
                                      <SEP>
                                      <R:SCALEFACTOR>
                                  <TERM>
```

```
DEVICE VIEWPORT MAPPING     ::=   DEVVPMAP
                                     <SOFTSEP>
                                     <NOTFORCED|FORCED>
                                     <SEP>
                                     <LEFT | CTR | RIGHT>
                                     <SEP>
                                     <BOTTOM | CTR | TOP>
                                  <TERM>

LINE REPRESENTATION         ::=   LINEREP
                                     <SOFTSEP>
                                     <I:BUNDLEINDEX> {positive}
                                     <SEP>
                                     <I:LINETYPE>
                                        {1=solid, 2=dash
                                        3=dot, 4=dash-dot
                                        5=dash-dot-dot
                                        >5=reserved for registered values
                                        <0 private line types}
                                     <SEP>
                                     <V:LINEWIDTH> {non-negative}
                                     <SEP>
                                     <K:LINECOLR>
                                  <TERM>

MARKER REPRESENTATION       ::=   MARKERREP
                                     <SOFTSEP>
                                     <I:BUNDLEINDEX> {positive}
                                     <SEP>
                                     <I:MARKERTYPE>
                                        {1=dot, 2=plus
                                        3=asterisk, 4=circle
                                        5=cross
                                        >5=reserved for registered values
                                        <0 private marker types}
                                     <SEP>
                                     <V:MARKERSIZE> {non-negative}
                                     <SEP>
                                     <K:MARKERCOLR>
                                  <TERM>

TEXT REPRESENTATION         ::=   TEXTREP
                                     <SOFTSEP>
                                     <I:BUNDLEINDEX> {positive}
                                     <SEP>
                                     <I:FONTINDEX> {positive}
                                     <SEP>
                                     <STRING|CHAR|STROKE>
                                     <SEP>
                                     <R:SPACING>
                                     <SEP>
                                     <R:FACTOR> {non-negative}
                                     <SEP>
                                     <K:TEXTCOLR>
                                  <TERM>

FILL REPRESENTATION         ::=   FILLREP
                                     <SOFTSEP>
                                     <I:BUNDLEINDEX> {positive}
                                     <SEP>
```



```
                                    <HOLLOW | SOLID | PAT | HATCH | EMPTY |
                                     GEOPAT | INTERP>
                                    <SEP>
                                    <K:FILLCOLR>
                                    <SEP>
                                    <I:HATCHINDEX>
                                      {1=horizontal,2=vertical
                                      3=positive slope
                                      4=negative slope
                                      5=horizontal/vertical cross
                                      6=positive/negative slope cross
                                      >6=reserved for registered values
                                      <0 private hatch styles}
                                    <SEP>
                                    <I:PATINDEX> {positive}
                                <TERM>

EDGE REPRESENTATION           ::=   EDGEREP
                                    <SOFTSEP>
                                    <I:BUNDLEINDEX> {positive}
                                    <SEP>
                                    <I:EDGETYPE>
                                      {1=solid, 2=dash
                                      3=dot, 4=dash-dot
                                      5=dash-dot-dot
                                      >5=reserved for registered values
                                      <0 private edge types}
                                    <SEP>
                                    <V:EDGEWIDTH>  {non-negative}
                                    <SEP>
                                    <K:EDGECOLR>
                                <TERM>

INTERIOR STYLE SPECIFICATION MODE
                              ::=     INTSTYLEMODE
                                    <SOFTSEP>
                                    <ABS | SCALED | FRACTIONAL | MM>
                                <TERM>

LINE AND EDGE TYPE DEFINITION   ::= LINEEDGETYPEDEF
                                    <SOFTSEP>
                                    <I:LINETYPE> {negative}
                                    <SEP>
                                    <V:REPEATLENGTH>
                                    <SEP> <I:DASHELEMENT> {non-negative}
                                    <<SEP> <I:DASHELEMENT>>* {non-negative}
                                <TERM>

HATCH STYLE DEFINITION        ::=   HATCHSTYLEDEF
                                    <SOFTSEP>
                                    <I:HATCHINDEX> {negative}
                                    <SEP>
                                    <PARALLEL | CROSSHATCH>
                                    <SEP>
                                    <V:FIRSTDIRX>
                                    <SEP>
                                    <V:FIRSTDIRY>
                                    <SEP>
                                    <V:SECONDDIRX>
                                    <SEP>
```

```
                                        <V:SECONDDIRY>
                                        <SEP>
                                        <V:DUTYCYCLE>
                                        <SEP>
                                        <I:NUMBEROFLINES>  {positive, this defines n}
                                        <<SEP> <I:GAPWIDTH>>(n) {positive}
                                        <<SEP> <I:LINETYPE>>(n)
                                    <TERM>

GEOMETRIC PATTERN DEFINITION::=     GEOPATDEF
                                        <SOFTSEP>
                                        <I:GEOPATINDEX> {positive}
                                        <SEP>
                                        <N:SEGID>
                                        <SEP>
                                        <P:FIRSTPOINT>
                                        <SEP>
                                        <P:SECONDPOINT>
                                    <TERM>

APPLICATION STRUCTURE DIRECTORY
                              ::=     APSDIR
                                        <SOFTSEP>
                                        <UI8 | UI16 | UI32>
                                        <<SEP><APSDIRPAIR>>+
                                    <TERM>

APSDIRPAIR                    ::=   <SF:APSID>
                                        <SEP>
                                        <I:APSLOCATION>
```

NOTE 11 The first (enumerated) parameter selects the location data type ([ldt]) for the subsequent APSLOCATION parameter. This is an integer-bound parameter, and the [ldt] selects one of three unsigned integer precisions. The values of APSLOCATION are the offsets measured in characters from the first character of the BEGIN PICTURE element containing the APS to the first character of the associated BEGIN APPLICATION STRUCTURE element.

NOTE 12 Editing of clear text encoded Version 4 metafiles may invalidate APSLOCATION values. If so, these values must be recomputed.

7.4 Encoding control elements

```
VDC INTEGER PRECISION         ::=   VDCINTEGERPREC
                                        <SOFTSEP>
                                        <I:MININT>
                                        <SEP>
                                        <I:MAXINT>
                                    <TERM>
```

See INTEGERPREC for description of the parameters.

```
VDC REAL PRECISION            ::=   VDCREALPREC
                                        <SOFTSEP>
                                        <R:MINREAL>
                                        <SEP>
                                        <R:MAXREAL>
                                        <SEP>
                                        <I:DIGITS>
                                    <TERM>
```

See REALPREC for description of the parameters.

```
AUXILIARY COLOUR              ::=   AUXCOLR
```

```
                                      <SOFTSEP>
                                      <K:AUXCOLR>
                                  <TERM>

TRANSPARENCY                 ::=  TRANSPARENCY
                                      <SOFTSEP>
                                      < OFF | ON >
                                  <TERM>

CLIP RECTANGLE               ::=  CLIPRECT
                                      <SOFTSEP>
                                      <P:FIRSTCORNER>
                                      <SEP>
                                      <P:SECONDCORNER>
                                  <TERM>

CLIP INDICATOR               ::=  CLIP
                                      <SOFTSEP>
                                      < OFF | ON >
                                  <TERM>

LINE CLIPPING MODE           ::=  LINECLIPMODE
                                      <SOFTSEP>
                                      <LOCUS|SHAPE|LOCUSTHENSHAPE>
                                  <TERM>

MARKER CLIPPING MODE         ::=  MARKERCLIPMODE
                                      <SOFTSEP>
                                      <LOCUS|SHAPE|LOCUSTHENSHAPE>
                                  <TERM>

EDGE CLIPPING MODE           ::=  EDGECLIPMODE
                                      <SOFTSEP>
                                      <LOCUS|SHAPE|LOCUSTHENSHAPE>
                                  <TERM>

NEW REGION                   ::=  NEWREGION
                                      <TERM>

SAVE PRIMITIVE CONTEXT       ::=  SAVEPRIMCONT
                                      <SOFTSEP>
                                      <I:CONTEXTNAME>
                                  <TERM>

RESTORE PRIMITIVE CONTEXT    ::=  RESPRIMCONT
                                      <SOFTSEP>
                                      <I:CONTEXTNAME>
                                  <TERM>

PROTECTION REGION INDICATOR       ::= PROTREGION
                                      <SOFTSEP>
                                      <I:REGIONINDEX> {positive}
                                      <SEP>
                                      <I:REGIONINDICATOR>
                                         {1=off, 2=clip, 3=shield}
                                  <TERM>

GENERALIZED TEXT PATH MODE::=     GENTEXTPATHMODE
                                      <SOFTSEP>
                                      <OFF | NONAXIS | AXIS>
                                  <TERM>
```

```
MITRE LIMIT                   ::=   MITRELIMIT
                                       <SOFTSEP>
                                       <R:MITRELIMIT> {non-negative}
                                    <TERM>

TRANSPARENT CELL COLOUR       ::=   TRANSPCELLCOLR
                                       <SOFTSEP>
                                       <OFF | ON>
                                       <SEP>
                                       <K:TRANSPCELLCOLOUR>
                                    <TERM>
```

7.5 Encoding graphical primitive elements

```
POLYLINE                      ::=   < LINE
                                       <POINTLIST>
                                       <TERM>
                                    >
                                    |
                                    < INCRLINE
                                       <INCRPOINTLIST>
                                       <TERM>
                                    >
```

LINE and INCRLINE shall always have at least two points.

```
POINTLIST                     ::=   <SOFTSEP>
                                    <P:POINT>
                                    < <SEP> <P:POINT> >+
INCRPOINTLIST                 ::=   <SOFTSEP>
                                    <P:FIRSTPOINT>
                                    <DELTA>+

DELTA                         ::=   < <SEP> <DELTAPAIR> > |
                                    < <SEP> <LEFT PAREN> <DELTAPAIR> <RIGHT PAREN> >

DELTAPAIR                     ::=   <OPTSEP>
                                    <VDC:DELTAX>
                                    <SEP>
                                    <VDC:DELTAY>
                                    <OPTSEP>
```

NOTE 1 Absolute coordinates correspond in a straightforward way to the functionality definition, but incremental coordinates can realize a significant savings in this encoding. Even more important, incremental coordinates are high on the priority list for user-friendly graphics, which is a high priority objective of this encoding.

```
DISJOINT POLYLINE             ::=    < DISJTLINE
                                       <SOFTSEP>
                                       <P:POINT>
                                       <SEP>
                                       <P:POINT> >
                                       < <SEP>
                                            <P:POINT>
                                            <SEP>
                                            <P:POINT> >*
                                       <TERM>
                                       >
                                       |
                                       < INCRDISJTLINE
                                       <SOFTSEP>
                                       <P:POINT>
                                       <DELTA>
```

< <DELTA> <DELTA> >*
<TERM>
>

DISJTLINE and INCRDISJTLINE shall have pairs of points, and at least one pair.

POLYMARKER ::= < MARKER
<SOFTSEP>
<P:POINT>
<<SEP><P:POINT>>*
<TERM>
>
|
< INCRMARKER
<SOFTSEP>
<P:FIRSTPOINT>
<DELTA>*
<TERM>
>

TEXT ::= TEXT
<SOFTSEP>
<P:TEXTLOCATION>
<SEP>
<TEXTPIECE>

RESTRICTED TEXT ::= RESTRTEXT
<SOFTSEP>
<VDC:MAXWIDTH> {non-negative}
<SEP>
<VDC:MAXHEIGHT> {non-negative}
<SEP>
<P:TEXTLOCATION>
<SEP>
<TEXTPIECE>

APPEND TEXT ::= APNDTEXT
<SOFTSEP>
<TEXTPIECE>

TEXTPIECE ::= < NOTFINAL | FINAL >
< <OPTSEP> | <HARDSEP> >
<S:TEXTSTRING>
<TERM>

POLYGON ::= < POLYGON
<SOFTSEP>
<P:POINT>
<SEP>
<P:POINT>
< <SEP> <P:POINT> >+
TERM>
>
|
< INCRPOLYGON
<SOFTSEP>
<P:POINT>
<DELTA>
<DELTA>+
<TERM>
>

```
POLYGON SET            ::=   < POLYGONSET
                                 <SOFTSEP>
                                 <FLAGGEDPOINT>
                                 <SEP>
                                 <FLAGGEDPOINT>
                                 < <SEP> <FLAGGEDPOINT> >+
                                 <TERM>
                             >

                             |

                             < INCRPOLYGONSET
                                 <SOFTSEP>
                                 <FLAGGEDPOINT>
                                 < <FLAGGEDDELTA> <SEP> >+
                                 <FLAGGEDDELTA>
                                 <TERM>
                             >
```

For all POLYGON-type primitives, at least three points shall be present.

```
EDGEFLAG               ::=   INVIS | VIS | CLOSEINVIS | CLOSEVIS

FLAGGEDPOINT           ::=   <P:VERTEX> <SEP> <EDGEFLAG>

FLAGGEDDELTA           ::=   <DELTA> <SEP> <EDGEFLAG>

CELL ARRAY             ::=   CELLARRAY
                                 <SOFTSEP>
                                 <P:P_POINT>
                                 <SEP>
                                 <P:Q_POINT>
                                 <SEP>
                                 <P:R_POINT>
                                 <SEP>
                                 <I:NX> {positive}
                                 <SEP>
                                 <I:NY> {positive}
                                 <LOCLCOLRPREC>
                                 <CELLROW>+
                             <TERM>

LOCLCOLRPREC           ::=   < <SEP> <I:MAXINT> >   |
                                 < <SEP> <I:MAXCOMPONENT>  >  |
                                 < <SEP> <I:0>  >
```

The LOCLCOLRPREC takes the form of COLRINDEXPREC or COLRPREC, depending on whether the COLRMODE is <INDEXED> or <DIRECT>, respectively. The value 0 is the 'default precision indicator', and denotes that the precision currently in effect shall be used.

```
CELLROW                ::=   < <SEP> <CELLLIST> > |
                             < <SEP> <LEFT PAREN>
                             <CELLLIST> <RIGHT PAREN> >

CELLLIST               ::=   <K:CELL> < <SEP> <K:CELL> >*
```

The parenthesized form of the list is optional. If parentheses are used, they delimit a row of cells. Each row is considered to start from the side defined by the points (P,Q,R) as defined in ISO/IEC 8632-1. The number of cells between parentheses shall be less than or equal to the row length. If a row is not complete, then the last defined cell in the row is replicated to fill the row.

The local colour precision parameter, LOCLCOLRPREC, takes the form of the parameter of a colour-precision-setting element, either of COLRPREC or of COLRINDEXPREC, depending on the COLRMODE. Legal values are either legal values of one of those colour precision elements, or zero. If the value is zero, then the metafile's COLRPREC or COLRINDEXPREC is to be used within the CELLARRAY element as well.

```
GENERALIZED DRAWING PRIMITIVE   ::=  GDP
                                          <SOFTSEP>
                                          <I:GDP_IDENTIFIER>
                                          <<SEP><P:POINT>>*
                                          < <OPTSEP> | <HARDSEP> >
                                          <S:DATARECORD>
                                     <TERM>

RECTANGLE                       ::=  RECT
                                          <SOFTSEP>
                                          <P:FIRSTCORNER>
                                          <SEP>
                                          <P:SECONDCORNER>
                                     <TERM>

CIRCLE                          ::=  CIRCLE
                                          <SOFTSEP>
                                          <P:CENTRE>
                                          <SEP>
                                          <VDC:RADIUS>  {non-negative}
                                     <TERM>

CIRCULAR ARC 3 POINT            ::=  ARC3PT
                                          <3PTARCSPEC>
                                     <TERM>

CIRCULAR ARC 3 POINT CLOSE      ::=  ARC3PTCLOSE
                                          <3PTARCSPEC>
                                          <CLOSESPEC>
                                     <TERM>

CLOSESPEC                       ::=  <SEP>
                                          < PIE | CHORD >
3PTARCSPEC                      ::=  <SOFTSEP>
                                          <P:STARTPOINT>
                                          <SEP>
                                          <P:INTERMEDIATEPOINT>
                                          <SEP>
                                          <P:ENDPOINT>

CIRCULAR ARC CENTRE             ::=  ARCCTR
                                          <CTRARCSPEC>
                                     <TERM>

CIRCULAR ARC CENTRE CLOSE       ::=  ARCCTRCLOSE
                                          <CTRARCSPEC>
                                          <CLOSESPEC>
                                     <TERM>

CTRARCSPEC                      ::=  <SOFTSEP>
                                     <P:CENTREPOINT>
                                     <ARCBOUNDS>
                                     <SEP>
                                     <VDC:RADIUS>  {non-negative}

ARCBOUNDS                       ::=  <SEP>
                                     <P:STARTVECTOR>
                                     <SEP>
                                     <P:ENDVECTOR>
```

NOTE 2 The start and end vectors are given as point records rather than 2*VDC to permit the parenthesized form to be used to represent the vectors.

```
ELLIPSE                      ::=  ELLIPSE
                                      <ELLIPSESPEC>
                                  <TERM>

ELLIPSESPEC                  ::=  <SOFTSEP>
                                  <P:CENTRE>
                                  <SEP>
                                  <P:ENDPOINT_FIRST_CONJUGATE_DIAMETER>
                                  <SEP>
                                  <P:ENDPOINT_SECOND_CONJUGATE_DIAMETER>

ELLIPTICAL ARC               ::=  ELLIPARC
                                      <ELLIPSESPEC>
                                      <ARCBOUNDS>
                                  <TERM>

ELLIPTICAL ARC CLOSE         ::=  ELLIPARCCLOSE
                                      <ELLIPSESPEC>
                                      <ARCBOUNDS>
                                      <CLOSESPEC>
                                  <TERM>

CIRCULAR ARC CENTRE REVERSED      ::=  ARCCTRREV
                                      <CTRARCSPEC>
                                  <TERM>

CONNECTING EDGE              ::=  CONNEDGE
                                  <TERM>

HYPERBOLIC ARC               ::=  HYPERBARC
                                      <SOFTSEP>
                                      <P:CENTREPOINT>
                                      <SEP>
                                      <P:TRANSVERSPOINT>
                                      <SEP>
                                      <P:CONJUGATEPOINT>
                                      <SEP>
                                      <VDC:STARTX>
                                      <SEP>
                                      <VDC:STARTY>
                                      <SEP>
                                      <VDC:ENDX>
                                      <SEP>
                                      <VDC:ENDY>
                                  <TERM>

PARABOLIC ARC                ::=  PARABARC
                                      <SOFTSEP>
                                      <P:TANGENTPOINT>
                                      <SEP>
                                      <P:STARTPOINT>
                                      <SEP>
                                      <P:ENDPOINT>
                                  <TERM>

NON-UNIFORM B-SPLINE         ::=  NUB
                                      <SOFTSEP>
                                      <I:SPLINEORDER> {m}
                                      <SEP>
```

```
                                         <I:NUMBERCONTROLPOINTS> {n>=m}
                                         <<SEP> <P:CONTROLPOINT>>(n)
                                         <<SEP> <R:KNOT>>(m+n)
                                         <SEP>
                                         <R:STARTVALUE>
                                         <SEP>
                                         <R:ENDVALUE>
                                     <TERM>

NON-UNIFORM RATIONAL B-SPLINE        ::= NURB
                                         <SOFTSEP>
                                         <I:SPLINEORDER> {m}
                                         <SEP>
                                         <I:NUMBERCONTROLPOINTS> {n>=m}
                                         <<SEP> <P:CONTROLPOINT>>(n)
                                         <<SEP> <R:KNOT>>(m+n)
                                         <SEP>
                                         <R:STARTVALUE>
                                         <SEP>
                                         <R:ENDVALUE>
                                         <<SEP> <R:WEIGHT>>(n)
                                     <TERM>

POLYBEZIER                       ::= POLYBEZIER
                                         <SOFTSEP>
                                         <I:CONTINUITYINDICATOR>
                                            {1=discontinuous, 2=continuous,
                                            >2, reserved for registered values}
                                         <<SEP> <P:CONTROLPOINT>>(n) {n>=4}
                                     <TERM>

POLYSYMBOL                       ::= < SYMBOL
                                         <SOFTSEP>
                                         <I:INDEX>
                                         <SEP>
                                         <P:POINT>
                                         <<SEP><P:POINT>>*
                                         <TERM>
                                     >
                                     | < INCRSYMBOL
                                         <SOFTSEP>
                                         <I:INDEX>
                                         <SEP>
                                         <P:FIRSTPOINT>
                                         <DELTA>*
                                         <TERM>
                                     >
BITONAL TILE                     ::= BITONALTILE
                                         <SOFTSEP>
                                         <I:CMPRSNTYPE>
                                            {0=null background,  1=null foreground,
                                            2=T6, 3=T4 1-dimensional, 4=T4 2-dimensional,
                                            5=bitmap, 6=run length,
                                            >6, reserved for registered values}
                                         <SEP>
                                         <I:ROWPADINDICATOR> {non-negative}
                                         <SEP>
                                         <K:BACKGROUND>
                                         <SEP>
                                         <K:FOREGROUND>
```

```
                                                <SEP>
                                                <SDR:METHODSPECIFICPARAMS>
                                                <SEP>
                                                <B:COMPRESSEDCELLS>
                                            <TERM>

<SDR:METHODSPECIFICPARAMS>              ::=   <>{for compression types 1-5}
                                            |<<I: i_I><SEP><I: 1>
                                                <SEP><I: RUNCOUNTPRECISION>>{for type 6}
                                            |     {defined in the register, for type>6}
```

NOTE 3 i_I in the above denotes the integer value of the 'data type designator' for data type I as assigned in Annex C of ISO/IEC 8632-1.

NOTE 4 See 6.3.7 for complete SDR delimiting and formatting requirements.

```
TILE                                ::=     <TILE>
                                                <SOFTSEP>
                                                <I:CMPRSNTYPE>
                                                   {0=null background,  1=null foreground,
                                                   2=T6, 3=T4 1-dimensional, 4=T4 2-dimensional,
                                                   5=bitmap, 6=run length,
                                                   >6, reserved for registered values}
                                                <SEP>
                                                <I:ROWPADINDICATOR> {non-negative}
                                                <SEP>
                                                <CELLCOLRPREC>
                                                <SEP>
                                                <SDR:METHODSPECIFICPARAMS>
                                                <SEP>
                                                <B:COMPRESSEDCELLS>
                                            <TERM>
```

See CELL ARRAY (LOCLCOLRPREC parameter) for discussion of the CELLCOLRPREC production.

7.6 Encoding attribute elements

```
LINE BUNDLE INDEX                   ::=     LINEINDEX
                                                <SOFTSEP>
                                                <I:BUNDLEINDEX>  {positive}
                                            <TERM>

LINE TYPE                           ::=     LINETYPE
                                                <SOFTSEP>
                                                <I:LINETYPE>
                                                   { 1=solid, 2=dash, 3=dot,
                                                   4=dash-dot, 5=dash-dot-dot,
                                                   >5 reserved for registered values,
                                                   <0 private line types}
                                            <TERM>

LINE WIDTH                          ::=     LINEWIDTH
                                                <SOFTSEP>
                                                <V:LINEWIDTH>  {non-negative}
                                            <TERM>

LINE COLOUR                         ::=     LINECOLR
                                                <SOFTSEP>
                                                <K:LINECOLR>
                                            <TERM>

MARKER BUNDLE INDEX                 ::=     MARKERINDEX
                                                <SOFTSEP>
```



```
                                      <I:BUNDLEINDEX>  {positive}
                                  <TERM>

MARKER TYPE                  ::=  MARKERTYPE
                                      <SOFTSEP>
                                      <I:MARKERTYPE>
                                        { 1=dot, 2=plus, 3=asterisk, 4=circle
                                        5=cross,
                                        >5 reserved for registered values,
                                        <0 private marker types }
                                  <TERM>

MARKER SIZE                  ::=  MARKERSIZE
                                      <SOFTSEP>
                                      <V:MARKERSIZE>  {non-negative}
                                  <TERM>

MARKER COLOUR                ::=  MARKERCOLR
                                      <SOFTSEP>
                                      <K:MARKERCOLR>
                                  <TERM>

TEXT BUNDLE INDEX            ::=  TEXTINDEX
                                      <SOFTSEP>
                                      <I:BUNDLEINDEX>   {positive}
                                  <TERM>

TEXT FONT INDEX              ::=  TEXTFONTINDEX
                                      <SOFTSEP>
                                      <I:FONTINDEX>     {positive}
                                  <TERM>

TEXT PRECISION               ::=  TEXTPREC
                                      <SOFTSEP>
                                      < STRING | CHAR | STROKE >
                                  <TERM>

CHARACTER EXPANSION FACTOR        ::=  CHAREXPAN
                                      <SOFTSEP>
                                      <R:FACTOR>     {non-negative}
                                  <TERM>

CHARACTER SPACING            ::=  CHARSPACE
                                      <SOFTSEP>
                                      <R:SPACING>
                                  <TERM>

TEXT COLOUR                  ::=  TEXTCOLR
                                      <SOFTSEP>
                                      <K:TEXTCOLR>
                                  <TERM>

CHARACTER HEIGHT             ::=  CHARHEIGHT
                                      <SOFTSEP>
                                      <VDC:CHARHEIGHT>  {non-negative}
                                  <TERM>

CHARACTER ORIENTATION        ::=  CHARORI
                                      <SOFTSEP>
                                      <DELTAPAIR> {up vector}
                                      <SEP>
                                      <DELTAPAIR> {base vector}
```

```
                                          <TERM>

TEXT PATH                          ::=    TEXTPATH
                                              <SOFTSEP>
                                              < RIGHT | LEFT | UP | DOWN >
                                          <TERM>

TEXT ALIGNMENT                     ::=    TEXTALIGN
                                              <SOFTSEP>
                                              < NORMHORIZ | LEFT | CTR | RIGHT | CONTHORIZ >
                                              <SEP>
                                              < NORMVERT | TOP | CAP | HALF | BASE | BOTTOM |
                                              CONTVERT >
                                              <SEP>
                                              <R:CONTINUOUS_HORIZONTAL>
                                              <SEP>
                                              <R:CONTINUOUS_VERTICAL>
                                          <TERM>

CHARACTER SET INDEX                ::=    CHARSETINDEX
                                              <SOFTSEP>
                                              <I:CHARSETINDEX>  {positive}
                                          <TERM>

ALTERNATE CHARACTER SET INDEX      ::=    ALTCHARSETINDEX
                                              <SOFTSEP>
                                              <I:ALTCHARSETINDEX>   {positive}
                                          <TERM>

FILL BUNDLE INDEX                  ::=    FILLINDEX
                                              <SOFTSEP>
                                              <I:BUNDLEINDEX>  {positive}
                                          <TERM>

INTERIOR STYLE                     ::=    INTSTYLE
                                              <SOFTSEP>
                                              <HOLLOW | SOLID | PAT  | HATCH | EMPTY |
                                               GEOPAT | INTERP>
                                          <TERM>

FILL COLOUR                        ::=    FILLCOLR
                                              <SOFTSEP>
                                              <K:FILLCOLR>
                                          <TERM>

HATCH INDEX                        ::=    HATCHINDEX
                                              <SOFTSEP>
                                              <I:HATCHINDEX>
                                                 { 1=horizontal,
                                                 2=vertical
                                                 3=positive slope
                                                 4=negative slope
                                                 5=horizontal/vertical cross
                                                 6= positive/negative slope cross
                                                 >6 reserved for registered values,
                                                 <0 private hatch styles }
                                          <TERM>

PATTERN INDEX                      ::=    PATINDEX
                                              <SOFTSEP>
                                              <I:PATINDEX>   {positive}
                                          <TERM>
```



```
EDGE BUNDLE INDEX          ::=   EDGEINDEX
                                    <SOFTSEP>
                                    <I:BUNDLEINDEX>  {positive}
                                 <TERM>

EDGE TYPE                  ::=   EDGETYPE
                                    <SOFTSEP>
                                    <I:EDGETYPE>
                                      { 1=solid, 2=dash, 3=dot,
                                      4=dash-dot, 5=dash-dot-dot,
                                      >5 reserved for registered values,
                                      <0 private edge types }
                                 <TERM>

EDGE WIDTH                 ::=   EDGEWIDTH
                                    <SOFTSEP>
                                    <V:EDGEWIDTH>  {non-negative}
                                 <TERM>

EDGE COLOUR                ::=   EDGECOLR
                                    <SOFTSEP>
                                    <K:EDGECOLR>
                                 <TERM>

EDGE VISIBILITY            ::=   EDGEVIS
                                    <SOFTSEP>
                                    < OFF | ON >
                                 <TERM>

FILL REFERENCE POINT       ::=   FILLREFPT
                                    <SOFTSEP>
                                    <P:FILLREFPT>
                                 <TERM>

PATTERN TABLE              ::=   PATTABLE
                                    <SOFTSEP>
                                    <I:PATINDEX> {positive}
                                    <SEP>
                                    <I:NX> {positive}
                                    <SEP>
                                    <I:NY> {positive}
                                    <LOCLCOLRPREC>
                                    <CELLROW>*
                                 <TERM>
```

See CELL ARRAY for discussion of the LOCLCOLRPREC and CELLROW productions.

```
PATTERN SIZE               ::=   PATSIZE
                                    <SOFTSEP>
                                    <V_DELTAPAIR> {height vector}
                                    <SEP>
                                    <V_DELTAPAIR> {width vector}
                                 <TERM>
```

NOTE 1 Pattern size may only be 'absolute' (VDC) in Version 1 and 2 metafiles. In Version 3 and Version 4 metafiles it may be expressed in any of the 4 modes which can be selected with INTERIOR STYLE SPECIFICATION MODE.

```
<V_DELTAPAIR>              ::=   <OPTSEP>
                                 <V:DELTAX>
                                 <SEP>
                                 <V:DELTAY>
```

```
COLOUR TABLE              ::=   COLRTABLE
                                    <SOFTSEP>
                                    <I:STARTINGINDEX>   {non-negative}
                                    < <SEP> <DIRECTCOLR> >+
                                <TERM>

ASPECT SOURCE FLAGS       ::=   ASF
                                    <SOFTSEP>
                                    <ASFPAIR>
                                    < <SEP> <ASFPAIR> >*
                                <TERM>

ASFPAIR                   ::=   < ASFTYPE >
                                    <SEP>
                                    < INDIV | BUNDLED >

ASFTYPE                   ::=   < ASFNAME | PSEUDOASF >

ASFNAME                   ::=   < LINETYPE | LINEWIDTH | LINECOLR |
                                    MARKERTYPE | MARKERSIZE | MARKERCOLR |
                                    TEXTFONTINDEX | TEXTPREC | CHAREXPAN |
                                    CHARSPACE | TEXTCOLR | INTSTYLE |
                                    FILLCOLR | HATCHINDEX | PATINDEX |
                                    EDGETYPE | EDGEWIDTH | EDGECOLR >

PSEUDOASF                 ::=   < ALL | ALLLINE | ALLMARKER | ALLTEXT |
                                    ALLFILL | ALLEDGE >
```

The ASF type may either be a valid ASF name, or a pseudo-ASF. If the former, then the name shall match the name of the corresponding attribute element.

NOTE 2 The pseudo-ASFs are a shorthand convenience for setting a number of ASFs at once.

The pseudo-ASFs have the meanings:

ALL set all ASFs as indicated.

ALLLINE: set LINETYPE, LINEWIDTH, and LINECOLR ASFs as indicated.

ALLMARKER: set MARKERTYPE, MARKERSIZE, and MARKERCOLR ASFs as indicated.

ALLTEXT: set TEXTFONTINDEX, TEXTPREC, CHAREXPAN, CHARSPACE, and TEXTCOLR ASFs as indicated.

ALLFILL: set INTSTYLE, FILLCOLR, HATCHINDEX, and PATINDEX ASFs as indicated.

ALLEDGE: set EDGETYPE, EDGEWIDTH, and EDGECOLR as indicated.

```
PICK IDENTIFIER           ::=   PICKID
                                    <SOFTSEP>
                                    <I:PICKID>
                                <TERM>

LINE CAP                  ::=   LINECAP
                                    <SOFTSEP>
                                    <I:LINECAP>
                                       {1=unspecified, 2=butt, 3=round,
                                       4=projecting square, 5=triangle,
                                       >5 reserved for registered values }
                                    <SEP>
                                    <I:DASHCAP>
                                       {1=unspecified, 2=butt, 3=match,
                                       >3 reserved for registered values}
                                <TERM>
```

```
LINE JOIN                     ::=   LINEJOIN
                                        <SOFTSEP>
                                        <I:JOIN>
                                          {1=unspecified, 2=mitre,
                                          3=round, 4=bevel,
                                          >4 reserved for registered values}
                                    <TERM>

LINE TYPE CONTINUATION        ::=   LINETYPECONT
                                        <SOFTSEP>
                                        <I:CONTMODE>
                                          {1=unspecified, 2=continue,
                                          3=restart, 4=adaptive continue,
                                          >4 reserved for registered values}
                                    <TERM>

LINE TYPE INITIAL OFFSET      ::=   LINETYPEINITOFFSET
                                        <SOFTSEP>
                                        <R:PATTERNOFFSET>
                                    <TERM>

TEXT SCORE TYPE               ::=   TEXTSCORETYPE
                                        <SOFTSEP>
                                        < <I:SCORETYPE>
                                          {1=right score, 2=left score,
                                          3=through score, 4=kendot,
                                          >4 reserved for registered values}
                                        <SEP>
                                        <OFF | ON> >+
                                    <TERM>

RESTRICTED TEXT TYPE          ::=   RESTRTEXTTYPE
                                        <SOFTSEP>
                                        <I:RESTRICTIONTYPE>
                                          {1=basic, 2=boxed-cap, 3=boxed-all,
                                          4=isotropic-cap, 5=isotropic-all, 6=justified,
                                          >6 reserved for registered values}
                                    <TERM>

INTERPOLATED INTERIOR         ::=   INTERPINT
                                        <SOFTSEP>
                                        <I:STYLE>
                                          {1=parallel, 2=elliptical, 3=triangular,
                                          >3 reserved for registered values}
                                        <SEP>
                                        < <<V:X><SEP><V:Y>>(1)     {parallel}
                                    |     <<V:X><SEP><V:Y>>(2) >  {elliptical, triangular}
                                        <SEP><I:NBROFSTAGES>
                                        <<SEP><R:STAGEDESGN>>(m)
                                        <<<SEP><K:REFCOLRLIST>>(m+1) {parallel,elliptical}
                                    |   <<SEP><K:REFCOLRLIST>>(3)>      {triangular}
                                          {m is the number of stages}
                                    <TERM>

EDGE CAP                      ::=   EDGECAP
                                        <SOFTSEP>
                                        <I:EDGECAP>
                                          {1=unspecified, 2=butt, 3=round,
                                          4=projecting square, 5=triangle,
                                          >5 reserved for registered values}
                                        <SEP>
```

```
                                          <I:DASHCAP>
                                            {1=unspecified, 2=butt, 3=match,
                                            >3 reserved for registered values}
                                      <TERM>

EDGE JOIN                        ::=  EDGEJOIN
                                          <SOFTSEP>
                                          <I:JOIN>
                                            {1=unspecified, 2=mitre,
                                            3=round, 4=bevel,
                                            >4 reserved for registered values}
                                      <TERM>

EDGE TYPE CONTINUATION           ::=  EDGETYPECONT
                                          <SOFTSEP>
                                          <I:CONTMODE>
                                            {1=unspecified, 2=continue,
                                            3=restart, 4=adaptive continue,
                                            >4 reserved for registered values}
                                      <TERM>

EDGE TYPE INITIAL OFFSET         ::=  EDGETYPEINITOFFSET
                                          <SOFTSEP>
                                          <R:PATTERNOFFSET>
                                      <TERM>

SYMBOL LIBRARY INDEX             ::=  SYMBOLINDEX
                                          <SOFTSEP>
                                          <I:INDEX> {positive}
                                      <TERM>

SYMBOL COLOUR                    ::=  SYMBOLCOLR
                                          <SOFTSEP>
                                          <K:SYMBOLCOLOUR>
                                      <TERM>

SYMBOL SIZE                      ::=  SYMBOLSIZE
                                          <SOFTSEP>
                                          <HEIGHT | WIDTH | BOTH>
                                          <SEP>
                                          <VDC:HEIGHT>
                                          <SEP>
                                          <VDC:WIDTH>
                                      <TERM>

SYMBOL ORIENTATION               ::=  SYMBOLORI
                                          <SOFTSEP>
                                          <DELTAPAIR> {up vector}
                                          <SEP>
                                          <DELTAPAIR> {base vector}
                                      <TERM>
```

7.7 Encoding escape elements

```
ESCAPE                           ::=  ESCAPE
                                          <SOFTSEP>
                                          <I:IDENTIFIER>
                                          < <OPTSEP> | <HARDSEP> >
                                          <S:DATARECORD>
                                      <TERM>
```

7.8 Encoding external elements

```
MESSAGE                    ::=   MESSAGE
                                       <SOFTSEP>
                                       < NOACTION | ACTION >
                                       < <OPTSEP> | <HARDSEP> >
                                       <SF:MESSAGE_TEXT>
                                 <TERM>

APPLICATION DATA           ::=   APPLDATA
                                       <SOFTSEP>
                                       <I:IDENTIFIER>
                                       < <OPTSEP> | <HARDSEP> >
                                       <S:DATARECORD>
                                 <TERM>
```

7.9 Encoding segment control and segment attribute elements

```
COPY SEGMENT               ::=   COPYSEG
                                       <SOFTSEP>
                                       <I:SEGID>
                                       <SEP>
                                       <TM:TRANMATRIX>
                                       <NO|YES>
                                 <TERM>

INHERITANCE FILTER         ::=   INHFILTER
                                        <SOFTSEP>
                                        <ELEMORGROUPNAME>
                                        <<SEP><ELEMORGROUPNAME>>*
                                        <SEP>
                                        <STLIST|SEG>
                                 <TERM>

ELEMORGROUPNAME            ::=   <LINEINDEX |
                                       LINETYPE |
                                       LINEWIDTH |
                                       LINECOLR |
                                       LINECLIPMODE |
                                       MARKERINDEX |
                                       MARKERTYPE |
                                       MARKERSIZE |
                                       MARKERCOLR |
                                       MARKERCLIPMODE |
                                       TEXTINDEX |
                                       TEXTFONTINDEX |
                                       TEXTPREC |
                                       CHAREXPAN |
                                       CHARSPACE |
                                       TEXTCOLR |
                                       CHARHEIGHT |
                                       CHARORI |
                                       TEXTPATH |
                                       TEXTALIGN |
                                       FILLINDEX |
                                       INTSTYLE |
                                       FILLCOLR |
                                       HATCHINDEX |
                                       PATINDEX |
                                       EDGEINDEX |
```

EDGETYPE |
EDGEWIDTH |
EDGECOLR |
EDGEVIS |
EDGECLIPMODE |
FILLREFPT |
PATSIZE |
AUXCOLR |
TRANSPARENCY |
LINEATTR |
MARKERATTR |
TEXPRESANDPLACEMATTR |
TEXTPLACEMANDORIATTR |
FILLATTR |
EDGEATTR |
PATATTR |
OUTPUTCTRL |
PICKID |
ALLATTRCTRL |
ALLINH |
LINETYPEASF |
LINEWIDTHASF |
LINECOLRASF |
MARKERTYPEASF |
MARKERSIZEASF |
MARKERCOLRASF |
TEXTFONTINDEXASF |
TEXTPRECASF |
CHAREXPANASF |
CHARSPACEASF |
TEXTCOLRASF |
INTSTYLEASF |
FILLCOLRASF |
HATCHINDEXASF |
PATINDEXASF |
EDGETYPEASF |
EDGEWIDTHASF |
EDGECOLRASF |
ALLLINE |
ALLMARKER |
ALLTEXT |
ALLFILL |
ALLEDGE |
ALL |
MITRELIMIT |
LINECAP |
LINEJOIN |
LINETYPECONT |
LINETYPEINITOFFSET |
TEXTSCORETYPE |
RESTRTEXTTYPE |
INTERPOLATEDINTERIOR |
EDGECAP |
EDGEJOIN |
EDGETYPECONT |
EDGETYPEINITOFFSET |
SYMBOLLIBINDEX |
SYMBOLCOLR |
SYMBOLSIZE |
SYMBOLORI |

```
                                   SYMBOLATTR>
```

NOTE 1 ALLINH means all attributes, control elements and ASFs. ALLLINE, ALLMARKER, ALLTEXT, ALLFILL, ALLEDGE and ALL have the meaning defined in 7.7.

```
CLIP INHERITANCE             ::=    CLIPINH
                                       <SOFTSEP>
                                       <STLIST|INTERSECTION>
                                    <TERM>

SEGMENT TRANSFORMATION  ::=         SEGTRAN
                                       <SOFTSEP>
                                       <I:SEGID>
                                       <SEP>
                                       <TM:TRANMATRIX>
                                    <TERM>

SEGMENT HIGHLIGHTING         ::=    SEGHIGHL
                                       <SOFTSEP>
                                       <I:SEGID>
                                       <SEP>
                                       <NORMAL|HIGHL>
                                    <TERM>

SEGMENT DISPLAY PRIORITY     ::=    SEGDISPPRI
                                       <SOFTSEP>
                                       <I:SEGID>
                                       <SEP>
                                       <I:DISPLAYPRIORITY> {non-negative}
                                    <TERM>

SEGMENT PICK PRIORITY        ::=    SEGPICKPRI
                                       <SOFTSEP>
                                       <I:SEGID>
                                       <SEP>
                                       <I:PICKPRIORITY> {non-negative}
                                    <TERM>
```

7.10 Encoding application structure descriptor elements

```
APPLICATION STRUCTURE ATTRIBUTE::=APSATTR
                                       <SOFTSEP>
                                       <SF:APSATTRTYPE>
                                       <SEP>
                                       <SDR:DATARECORD>
                                    <TERM>
```

8 Clear text encoding defaults

CGM precisions for the Clear Text Encoding:

Non-VDC Reals:

MINREAL = -32767

MAXREAL = 32767

DIGITS = 4

VDC Precision for Reals:

MINREAL = 0.0

MAXREAL = 1.0

DIGITS = 4

Non-VDC Integers, Integer VDC:

MININT = -32767

MAXINT = 32767

Index:

MININT = 0

MAXINT = 127

Colour Index:

MAXINT = 127

Colour Precision:

MAXCOMPONENT = 255

Colour Value Extent:

BLACK = 0,0,0

WHITE = 255,255,255, if COLOUR MODEL is RGB;

WHITE = 0,0,0,0

BLACK = 255,255,255,255, if COLOUR MODEL is CMYK;

All scale and offset components are identically 0.0 if the COLOUR MODEL is CIELAB, CIELUV, or RGB-related.

NAME PRECISION:

MININT = -32767

MAXINT = 32767

9 Profile encoding rules, proforma, and Model Profile

9.1 Encodings

Precisions are defined consistently with the principles of the encodings, not necessarily for inter-encoding translation. Where both considerations might apply, compatibility with the principles of the encoding are considered first and inter-encoding translation second.

9.2 Metafile defaults

Clause 8 of part 1 and clause 8 of this part address all elements which have default values for the Clear Text encoding. While no profile can change these values, an equivalent effect may be achieved by use of the METAFILE DEFAULTS REPLACEMENT element. Profiles may require that a metafile contain a METAFILE DEFAULTS REPLACEMENT element with well-defined content.

9.3 Profile Proforma tables (PPF)

All elements are included in the Profile Proforma in Part 1. These include the following elements with specific requirements for clear text encoding:

Metafile descriptor elements: INTEGER PRECISION, REAL PRECISION, INDEX PRECISION, COLOUR PRECISION, COLOUR INDEX PRECISION, NAME PRECISION,

Control elements: VDC INTEGER PRECISION, and VDC REAL PRECISION.

The Profile Proforma fragments specifically directed to clear text encoding is contained in table 16 and table 18 of part 1, annex I. These tables, when completed by the author of the profile, contain the normative specifications of the profile.

Annex A
(normative)

Clear text encoding dependent format grammar

```
ALPHA        ::=    "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" |  "I"  |
                    “J"  | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" |
                    “S"  | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" |
                    “a"  | "b"  | "c"  | "d"  | "e"  | "f"   | "g"  | "h"  | "i"   |
                    “j"   | "k"  | "l"   | "m" | "n"  | "o"  | "p"  | "q" | "r"   |
                    “s"   | "t"  | "u"  | "v"  | "w" | "x"   | "y"  | "z"

DIGIT        ::=    "0"  | "1" | "2"  | "3"  | "4"  | "5"  | "6"  | "7"  | "8"  | “9"

NULLCHAR     ::=    "_"  | "$"
```

CGM opcodes are encoded as element names, and enumerated types are bound to names, as follows:

```
NAME         ::=    <NULLCHAR>*
                          <ALPHA>
                          <ALPHA> | <DIGIT> | <NULLCHAR> >*
```

The <element name> production is encoded in this encoding as the value of the associated clear text opcode, see 6.4.5.

Annex B
(informative)

Clear text encoding example

```
BEGMF 'metafile example';
    mfversion  1; mfdesc'24 January 1984'; vdctype real;
    indexprec -127,127;
    maxcolrindex 7; mfelemlist 'drawingplus';
    font_list 'Helvetica',
        'Perpetua Bold',
        'CGM_GENERIC: light italic'
        ;
    BEGMFDEFAULTS;
        VDCEXT 0,0,1,1;
        text_font_index 2;
        int_style solid;
    ENDMFDEFAULTS;

    % simple picture %
    BEGPIC 'PN 007' ;
        marker_size_mode abs;
    BEGPICBODY;
        % frame %
        line (0,0) (1,0) (1,1) (0,1) (0,0);

        % big dot %
        asf intstyle indiv; circle .5 .5 .3125;

        asf marker_size indiv, marker_type indiv;
        marker_size .005;
```



```
        marker_type -3;  % implementation-dependent %
        marker .01 .01
            .5 .5    % note 1 element on several lines %
            .99 .99/
        char_height .04 ;
        text_align ctr, bottom, 0, 0;
        text (.5,0) notfinal "PN 007 is a";
        text_font_index 3 ; apnd_text notfinal ' "silly" ';
        text_font_index 1 ; apnd_text final 'example';
    ENDPIC;
ENDMF ;
```

ICS 35.140

Price based on 50 pages