INTERNATIONAL STANDARD

ISO/IEC 8632-1

Second edition
1992-10-01

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 8632-1:1992(E)

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.

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.

International Standard ISO/IEC 8632-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

This second edition cancels and replaces the first edition (ISO 8632-1:1987), which has been technically revised.

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 2: Character encoding

Part 3: Binary encoding

Part 4: Clear text encoding

Annexes A, B, and C form an integral part of this part of ISO/IEC 8632. Annexes D, E, F and G are for information only

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 systems;

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.

Three standard encodings are specified in parts 2, 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 character encoding specified in ISO/IEC 8632-2 is intended to provide an encoding of minimum size. It conforms to the rules for code extension specified in ISO 2022 in the category of complete code system. It is particularly suitable for transfer through networks that cannot support binary transfers.

The binary encoding specified in ISO/IEC 8632-3 provides an encoding that requires least effort to generate and interpret on many systems.

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 character encoding specified in ISO/IEC 8632-2 conforms to the code extension techniques of ISO 2022.

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 4.12) 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 4, 5, and 6. Clause 7 contains conformance criteria defined by version.

These versions are currently defined: Version 1 (one); Version 2 (two); and Version 3 (three).

NOTES

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

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 this revision of ISO 8632; Versions 1, 2, and 3 are all defined by this revision.

3 Many of the figures in this International Standard were produced using ISO/IEC 8632 standard Version 1 metafiles, as defined herein.

This page intentionally left blank

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 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 images.

The elements specified provide for the representation of a wide range of pictures on a wide range of graphical devices. The elements are split into groups that delimit major structures (metafiles and pictures), 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 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 is well-defined; whether this is available in any system that uses ISO/IEC 8632 depends on the medium, the encoding and the implementation.

In addition to a functional specification, three standard encodings of the metafile syntax are specified. These encodings address the needs of applications that require minimum metafile size, 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 three parts of ISO 8632 specify three standardized encodings that conform to this syntax: ISO/IEC 8632-2 specifies a character encoding that conforms to the rules for code extension specified in ISO 2022 in the category of complete coding system; ISO/IEC 8632-3 specifies a binary encoding; ISO/IEC 8632-4 specifies a clear text encoding.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 8632. At the time of publication, the editions indicated were valid. All standards are subject to revision, and 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 standards listed below. Members of IEC and ISO 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/IEC 8613:1989/Amd.1:–[1)], *Information processing systems – Text and office systems – Office document architecture (ODA) and interchange format – Amendment 1: Tiled raster graphics.*

ISO/IEC 8613:1989/Amd.2:–[1)], *Information processing systems – Text and office systems – Office document architecture (ODA) and interchange format – Amendment 2: Colour.*

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 – Computer graphics – Procedures for registration of graphical items.*

ISO/IEC 10036:–[1)], *Information technology – Font information interchange – Procedure for registration of glyph and glyph collection identifiers.*

1) To be published.

Normative references

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).*

3 Definitions and abbreviations

3.1 Definitions

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

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

3.1.1 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".

3.1.2 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.

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

3.1.4 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.

3.1.5 attribute elements: Metafile elements that describe the appearance of graphical elements.

3.1.6 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.

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

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

3.1.9 bundle index: An index into a bundle table for a particular output primitive.

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

3.1.11 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.

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

3.1.13 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.)

3.1.14 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.

3.1.15 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 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.

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

3.1.17 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.

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

3.1.19 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'.

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

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

3.1.22 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.

3.1.23 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).

3.1.24 colour component: One of the dimensions of a colour space.

3.1.25 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.

3.1.26 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.

3.1.27 colour space: See COLOUR MODEL.

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

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

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

3.1.31 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.

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

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

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

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

3.1.36 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).

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

3.1.38 device coordinates: The coordinates native to a device; device-dependent coordinates; physical device coordinates.

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

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

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

3.1.42 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).

3.1.43 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".

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

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

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

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

3.1.48 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.

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

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

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

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

3.1.53 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.

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

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

3.1.56 graphical primitive elements: Metafile elements that describe images in the Metafile.

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

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

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

3.1.60 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.

3.1.61 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.

3.1.62 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.

3.1.63 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".

3.1.64 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.

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

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

3.1.67 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.

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

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

3.1.70 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.

3.1.71 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.

3.1.72 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.

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

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

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

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

3.1.77 primary colour stimuli: Three selected coloured lights used to specify the colour of any light presented 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.)

3.1.78 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.

3.1.79 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.

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

3.1.81 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.

3.1.82 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.

3.1.83 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 YC_RC_B (CCITT video codecs and CCIR601 studio standard).)

3.1.84 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).

3.1.85 saturation: Chromaticness (colourfulness) of an area judged in proportion to its brightness.

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

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

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

3.1.89 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.

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

3.1.91 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.

3.1.92 view surface: See DISPLAY SURFACE.

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

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

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

3.1.96 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.

3.1.97 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.

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

3.2 Abbreviations

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

ASF	Aspect Source Flag
CDP	Conjugate Diameter Pair
CGI	Computer Graphics Interface
CGM	Computer Graphics Metafile
GDP	Generalized Drawing Primitive
GKS	Graphical Kernel System
MD	Metafile Descriptor
NDC	Normalized Device Coordinate(s)
PD	Picture Descriptor
VDC	Virtual Device Coordinate(s)

4 Concepts

4.1 Introduction

The objective of the Computer Graphics Metafile (CGM) is to provide for the description, storage, and communication of graphical 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.

— Picture Descriptor Elements, which set the interpretation modes of attribute elements for 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 and manipulation of elements.

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 attribute 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 and Version 3 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.

4.2 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 and Version 3 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 and Version 3 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 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 elements.

In Version 3 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 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 8.

4.3 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

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.

4.3.1 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.

4.3.2 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 4.1 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.

4.3.2.1 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
VDC EXTENT
BACKGROUND COLOUR
COLOUR SELECTION MODE
POLYLINE
DISJOINT POLYLINE
POLYMARKER
TEXT
RESTRICTED TEXT

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
FILL BUNDLE INDEX
INTERIOR STYLE
FILL COLOUR
HATCH INDEX

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
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

4.3.2.2 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 for reasons of upward compatibility of versions.

4.3.2.3 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
MARKER CLIPPING MODE
EDGE CLIPPING MODE
NEW REGION
SAVE PRIMITIVE CONTEXT
RESTORE PRIMITIVE CONTEXT

MAXIMUM VDC EXTENT
SEGMENT PRIORITY EXTENT
DEVICE VIEWPORT
DEVICE VIEWPORT MAPPING
DEVICE VIEWPORT SPECIFICATION MODE
LINE REPRESENTATION
MARKER REPRESENTATION
TEXT REPRESENTATION
FILL REPRESENTATION
EDGE REPRESENTATION
LINE CLIPPING MODE
CIRCULAR ARC CENTRE REVERSED
CONNECTING EDGE
PICK IDENTIFIER
COPY SEGMENT
INHERITANCE FILTER
CLIP INHERITANCE
SEGMENT TRANSFORMATION
SEGMENT HIGHLIGHTING
SEGMENT DISPLAY PRIORITY
SEGMENT PICK PRIORITY

4.3.2.4 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 3 POINT CLOSE
CIRCULAR ARC CENTRE
CIRCULAR ARC CENTRE CLOSE
CIRCULAR ARC CENTRE REVERSED
ELLIPSE
ELLIPTICAL ARC
ELLIPTICAL ARC CLOSE
CONNECTING EDGE

4.3.2.5 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
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

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

4.3.2.6 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

4.3.3 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 6. 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.

4.3.4 Fonts and character sets

4.3.4.1 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.

4.3.4.2 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.

4.3.4.3 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 a parameters of type String 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.

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 ISO 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 ISO 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 ISO 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 ISO 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 ISO 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 ISO 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.

4.3.4.4 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.

NOTE — 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.

4.3.4.5 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 5.1, 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, 7-bit coding environment shall be used. It is not possible, either using ISO 2022 controls or any other defined facilities in this International Standard, to select other than 7-bit coding for type SF parameters.

In SF parameters, therefore, the two character sets which are currently designated as G0 and G1 may be accessed by using: SI to invoke G1 into positions 2/1 to 7/14 of the 7-bit code chart; and using SO to invoke G0 into positions 2/1 to 7/14 of the 7-bit code chart.

BEGIN METAFILE causes ISO 646 International Reference Version 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. In other words, ISO 646 is the default character set of type SF parameters, an SI embedded in the string will cause subsequent 7-bit codes to access the characters of ISO 8859-1, and an SO embedded in the string will cause subsequent 7-bit codes to access ISO 646 again. These two character set may be switched between using only SI and SO — no ISO 2022 designation sequences are required.

EXAMPLE — Many of the characters of the Latin-based alphabets would be accessible with ISO 646 and the Right Hand Side of ISO 8859-1, Latin Alphabet Number 1. These are the default G0 and G1 sets of the metafile. ISO 2022 sequences could be used to designate any other character sets as G0 and/or G1, and then SI/SO could be used to switch between these sets.

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).

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.

4.4 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
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

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

4.4.1 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.

4.4.2 Colour selection mode

COLOUR SELECTION MODE selects either indexed or direct colour specification and is described further under colour attributes, 4.7.6. For Version 1 metafiles, the selection is for the whole picture.

4.4.3 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.

4.4.4 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 6. 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.

4.4.5 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.

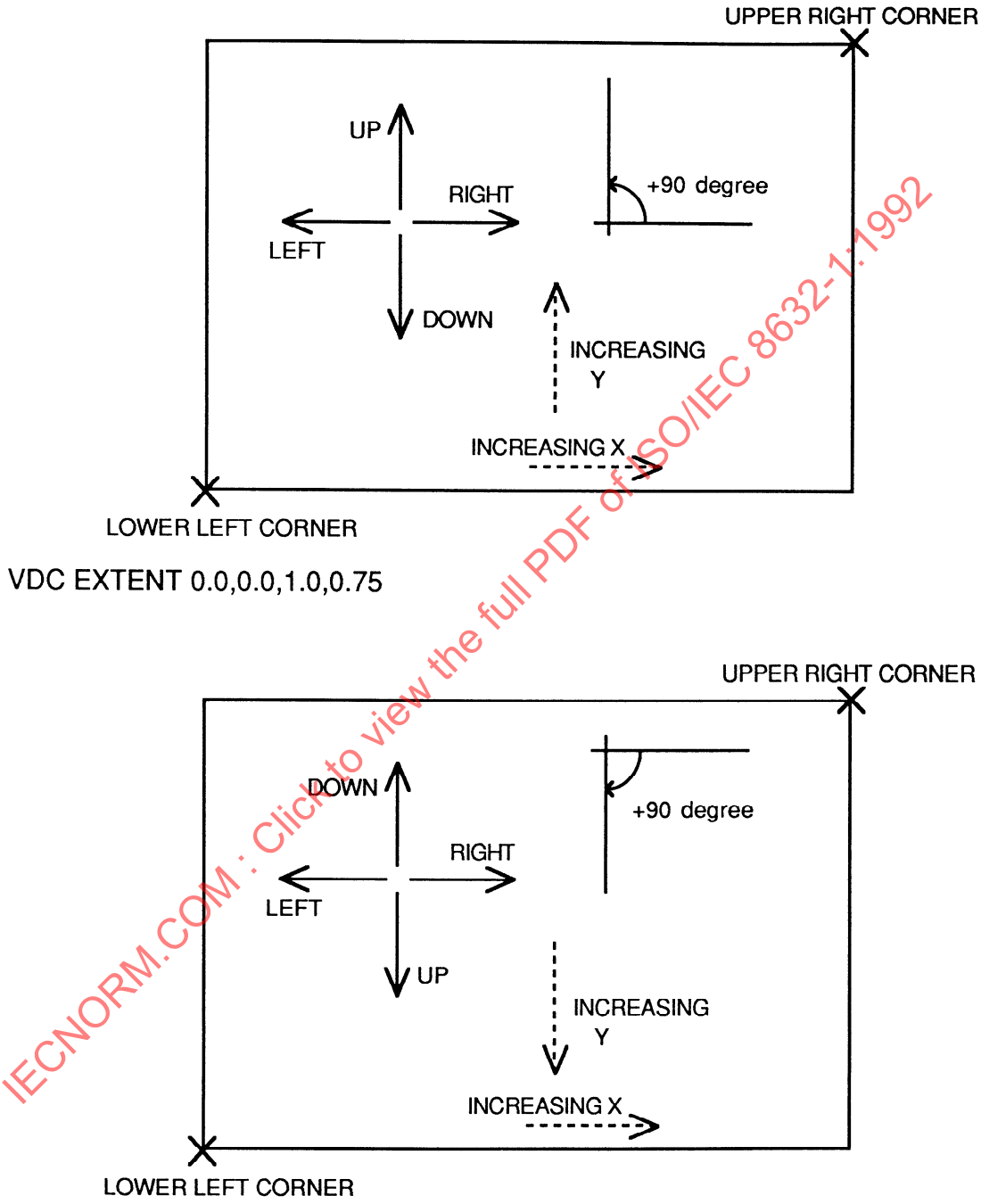


Figure 1 — VDC EXTENT establishes the direction of positive and negative angles

4.4.6 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. If that element is not contained in the Picture Descriptor, the background colour of the image is the default background colour, whether that default is as specified in clause 6. or has been specified in the METAFILE DEFAULTS REPLACEMENT element.

The single parameter of BACKGROUND COLOUR is always a direct colour, regardless of the current value of COLOUR SELECTION MODE. If the COLOUR SELECTION MODE is indexed, then the BACKGROUND COLOUR element defines the initial representation of colour index 0 for the picture.

4.4.7 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 4.6.

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.

4.4.8 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 4.7

4.4.9 Definable attributes

A precise definition of line and edge types as well as hatch styles may be accomplished as described in 4.7. 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 4.7.

4.5 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 8.

4.5.1 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.

4.5.2 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 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 4.5.4). 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 4.6.4). 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 4.10.5).

4.5.3 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 5.5.11) 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.

4.5.4 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 4.10.5).

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).

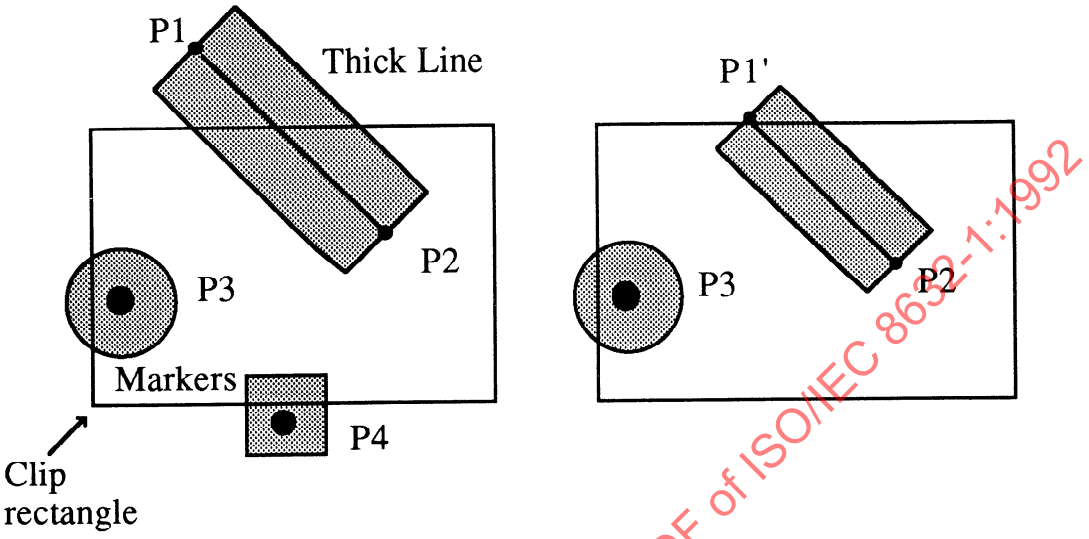
If a protection region is defined within a segment, then its definition persists after the END SEGMENT terminating that segment definition. This is exactly as for clip rectangles, and primitive attributes. Thus the definition and activation of a region within a local segment in a picture body potentially 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.

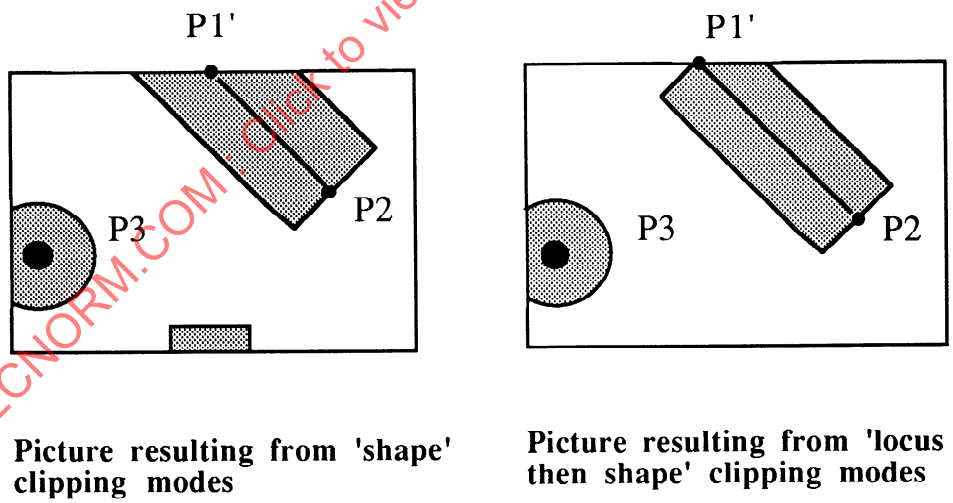
Several clip and shield regions may be in effect simultaneously. In this case, only portions of the graphical elements inside or on the boundary of the intersection of all individual clip regions and outside the union of

all individual shield regions are drawn. See figure 3.



Primitives and clip rectangle stored in the CGM

Picture resulting from 'locus' clipping modes



Picture resulting from 'shape' clipping modes

Picture resulting from 'locus then shape' clipping modes

Figure 2 — Clipping modes

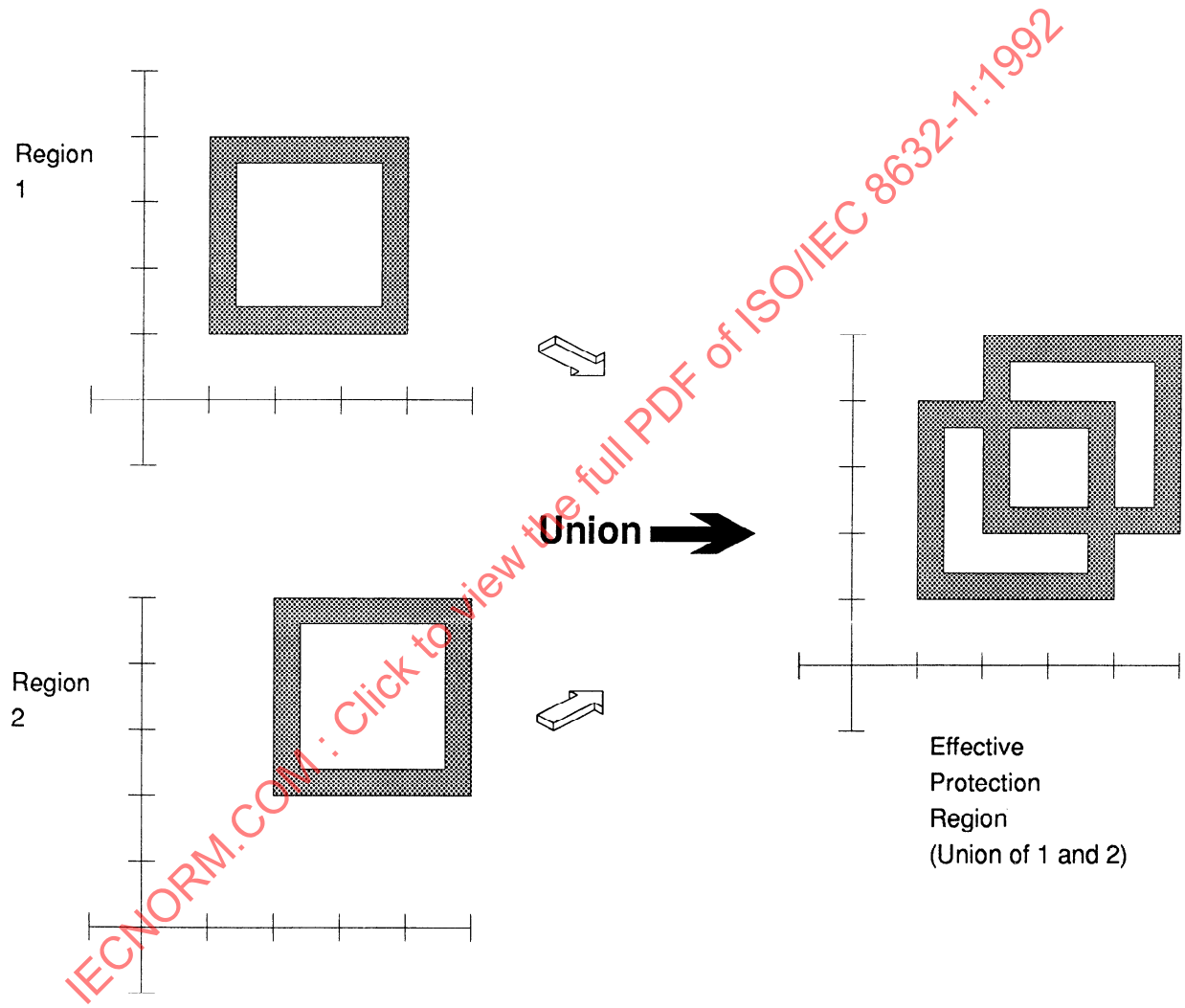


Figure 3 — Combining protection regions (hatching denotes the region)

4.5.5 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.

4.5.6 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.

4.5.7 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.

4.6 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
RESTRICTED TEXT
APPEND TEXT
POLYGON
POLYGON SET
CELL ARRAY
GENERALIZED DRAWING PRIMITIVE (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
CIRCULAR ARC CENTRE REVERSED
CONNECTING EDGE
HYPERBOLIC ARC
PARABOLIC ARC
NON-UNIFORM B-SPLINE
NON-UNIFORM RATIONAL B-SPLINE
POLYBEZIER
POLYSYMBOL
BITONAL TILE
TILE

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
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

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, RESTRICTED TEXT (Note 1)	APPEND TEXT (Note 2)		APPEND TEXT (Note 3)
Closed Figure	BEGIN FIGURE	Line Primitives, Fill Primitives (Note 4), GDP (Note 5)	NEW REGION	END FIGURE
Compound Line	BEGIN COMPOUND LINE	Line primitives, GDP (Note 5)		END COMPOUND LINE
Tile Array	BEGIN TILE ARRAY	BITONAL TILE, TILE		END TILE ARRAY

NOTES

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.

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

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

4 All primitives of the identified classes may be included.

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 4.10.4.2 and 4.10.5). 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.

4.6.1 Line elements

4.6.1.1 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 5.6.13 and 5.6.15. A reverse direction arc can also be specified; see 5.6.20.
ELLIPTICAL ARC	generates a single elliptical arc; the parameterization of the arc is described in 5.6.18.
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 5.6.22, and the principles underlying the transformable parameterization are described in 4.6.8.
PARABOLIC ARC	generates a parabolic arc; the parameterization is described in 5.6.23, and the principles underlying the transformable parameterization are described in 4.6.9.
NON-UNIFORM B-SPLINE	generates a Non-Uniform B-Spline curve; the parameterization is described in 5.6.24, and the principles underlying the definition of the element are described in 4.6.10.1.
NON-UNIFORM RATIONAL B-SPLINE	generates a Non-Uniform Rational B-Spline (NURBS) curve; the

parameterization is described in 5.6.25, and the principle underlying the definition of the element are described in 4.6.10.1.

POLYBEZIER — generates a sequence of one or more cubic Bezier curves; the parameterization is described in 5.6.26 and the principles underlying the definition of the element are described in 4.6.10.2.

4.6.1.2 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 physically 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.

4.6.1.3 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 4.7.1.

4.6.1.4 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.

4.6.1.5 Clipping of line elements

In Version 2 and Version 3 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.

4.6.1.6 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 4.4.7) as well as to both segment and copy transformation (see 4.10.4.2 and 4.10.5). 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.

4.6.2 Marker element

4.6.2.1 Description

There is a single marker element.

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

4.6.2.2 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 4.7.2.

4.6.2.3 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 and Version 3 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.

4.6.2.4 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 4.4.7) as well as to both segment and copy transformation (see 4.10.4.2 and 4.10.5). 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.

4.6.3 Text elements

4.6.3.1 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.

4.6.3.2 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 4.7.3.

Changes to certain of the text attributes, as listed in the description of APPEND TEXT (see 5.6.6), are permitted between a non-final text element and its succeeding APPEND TEXT element.

4.6.3.3 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.

4.6.3.4 Clipping of text elements

Clipping of text strings is described in 4.7.3.2.

4.6.3.5 Transformation of text elements

The vectors specified by the CHARACTER ORIENTATION element (see 4.7.3.2) are subject to the VDC-to-Device mapping (see 4.4.7) as well as to both segment and copy transformation (see 4.10.4.2 and 4.10.5).

4.6.4 Filled-area elements

4.6.4.1 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 4.6.11).

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 5.6.14 and 5.6.16; the set of styles is the same as for POLYGON.
ELLIPSE	generates an ellipse; the parameterization of the ellipse is described in 5.6.17; 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 5.6.19; the set of styles is the same as for POLYGON.

4.6.4.2 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 4.7.4.

4.6.4.3 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 4.6.11), 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.

4.6.4.4 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 4.7.4.3).

4.6.4.5 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 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.

4.6.4.6 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 4.6.1.6.

4.6.4.7 Transformation

The entire mathematical locus of rectangles, circular and elliptical filled-area elements is subject to the VDC-to-Device mapping (see 4.4.7), segment transformations (see 4.10.4.2) and copy transformations (see 4.10.5). 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.

4.6.5 Cell elements

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

4.6.5.1 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.

4.6.5.2 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.

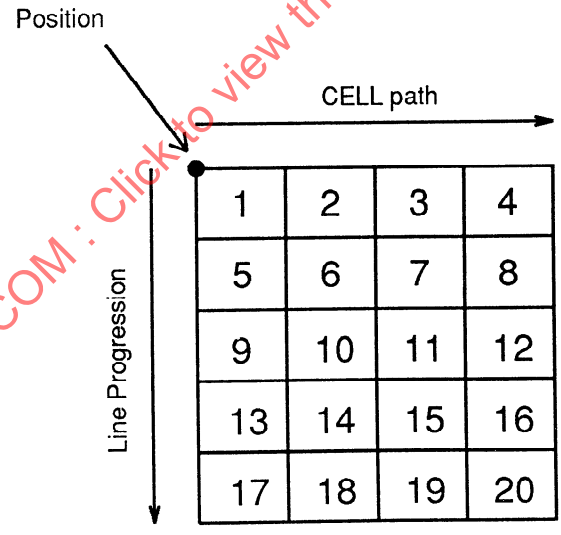


Figure 4 — Ordering and layout of tiles

4.6.5.2.1 *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).

4.6.5.2.2 *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.

4.6.5.2.3 *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.

4.6.5.2.4 *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.

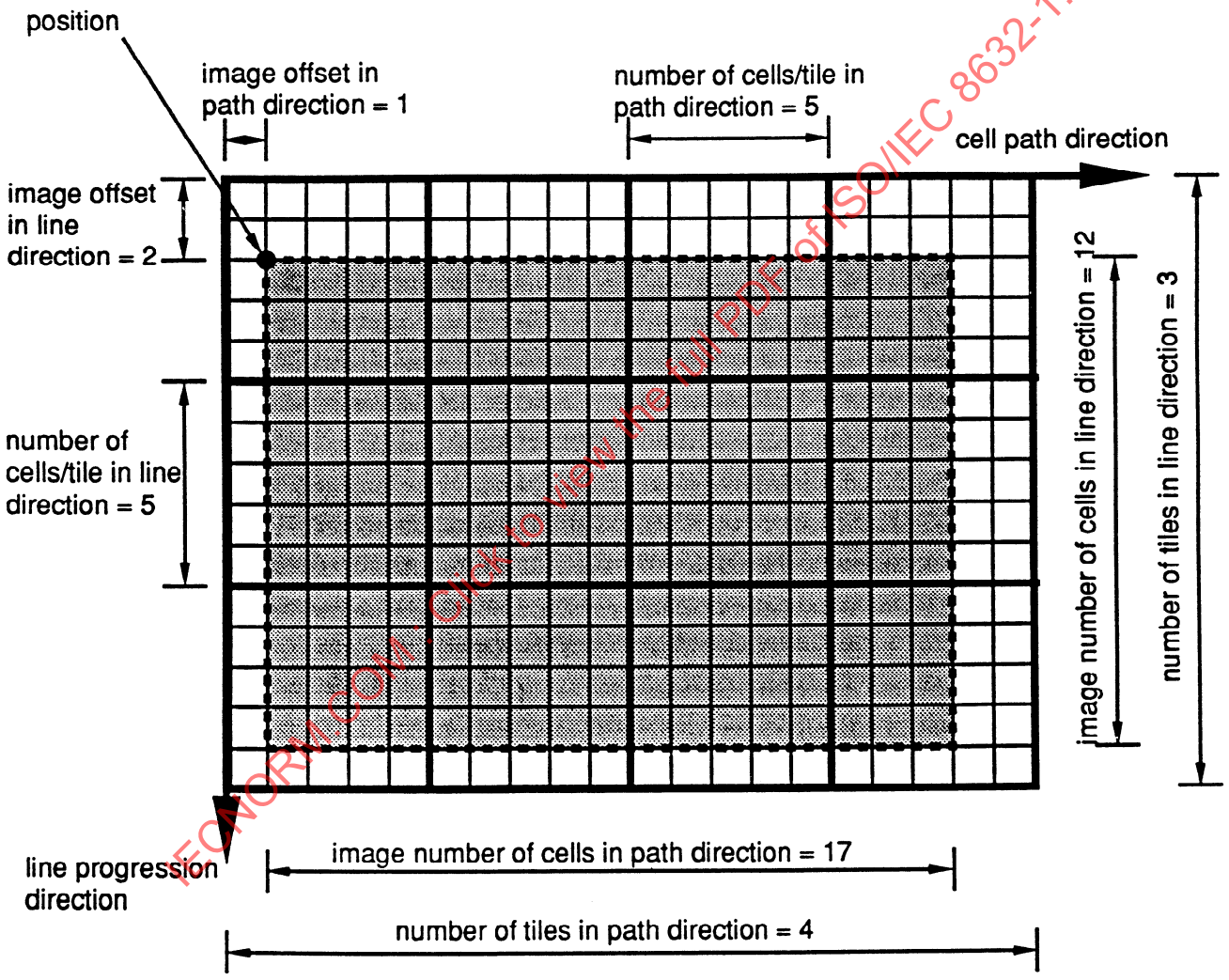


Figure 5 — Relationship of the image to the tile space

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.

4.6.6 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.

4.6.7 Elliptical elements

4.6.7.1 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 demonstrated 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).

4.6.7.2 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 $\mathbf{M}$ be the 2x2 matrix whose first column is $\mathbf{P}_1$ and whose second column is $\mathbf{P}_2$. The transformation $\mathbf{M}$ 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.

4.6.8 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.

4.6.9 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 $\mathbf{V}_1 = \mathbf{P}_1 - \mathbf{C}$ and $\mathbf{V}_2 = \mathbf{P}_2 - \mathbf{C}$, 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.

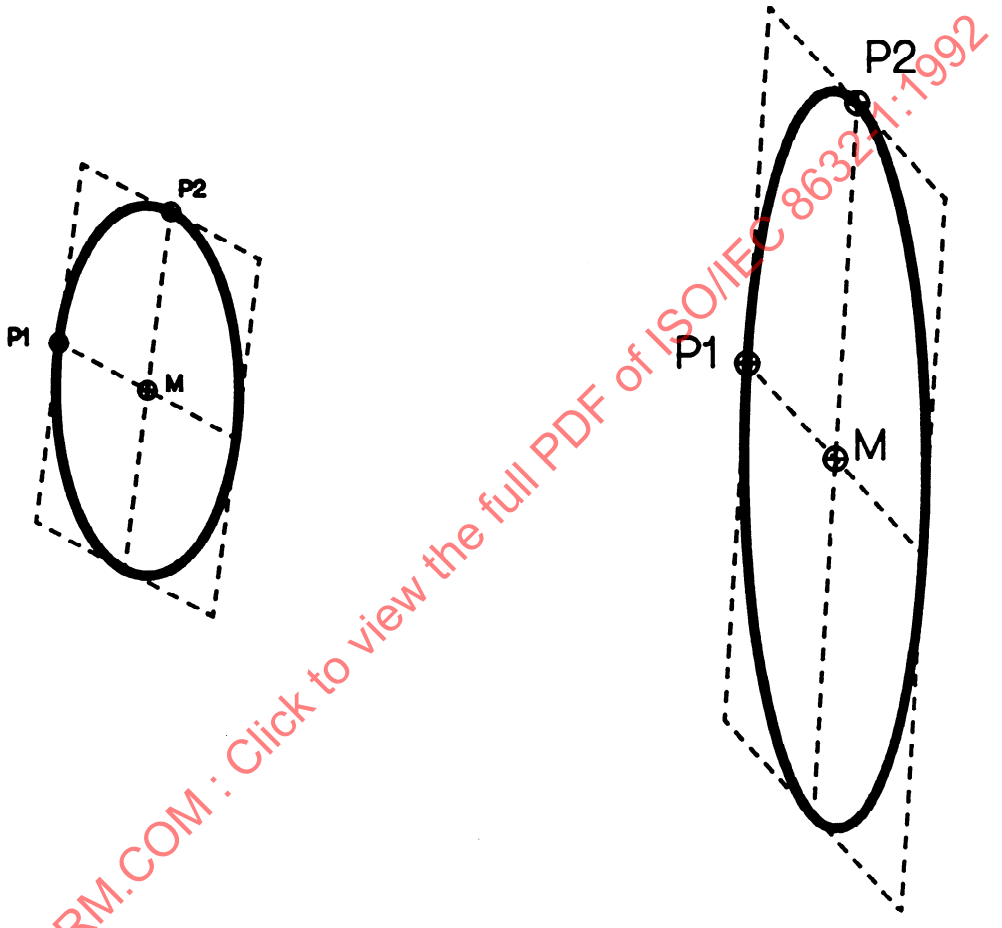


Figure 6 — Anisotropic scaling of an ellipse

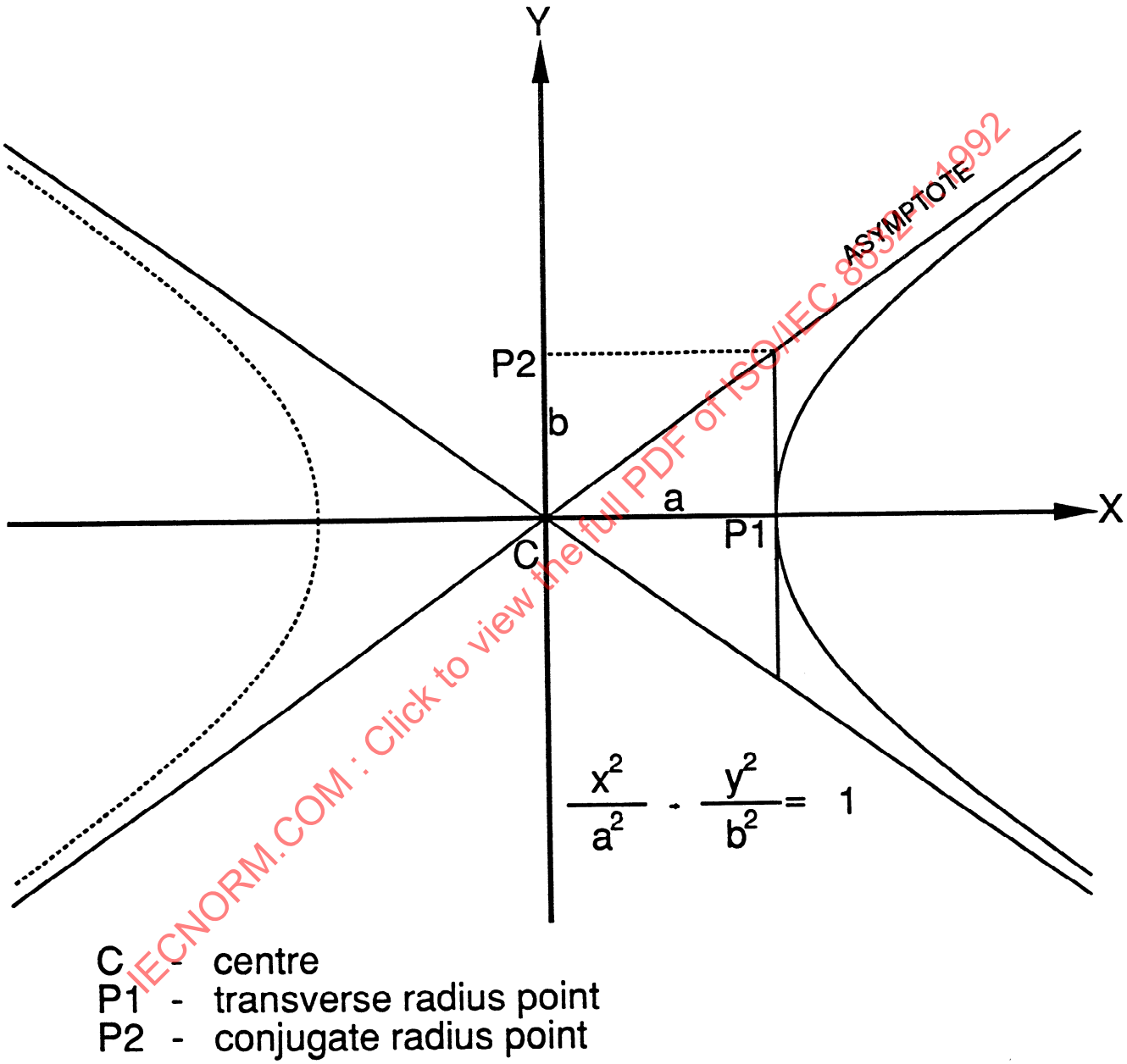


Figure 7 — Example of hyperbolic arc

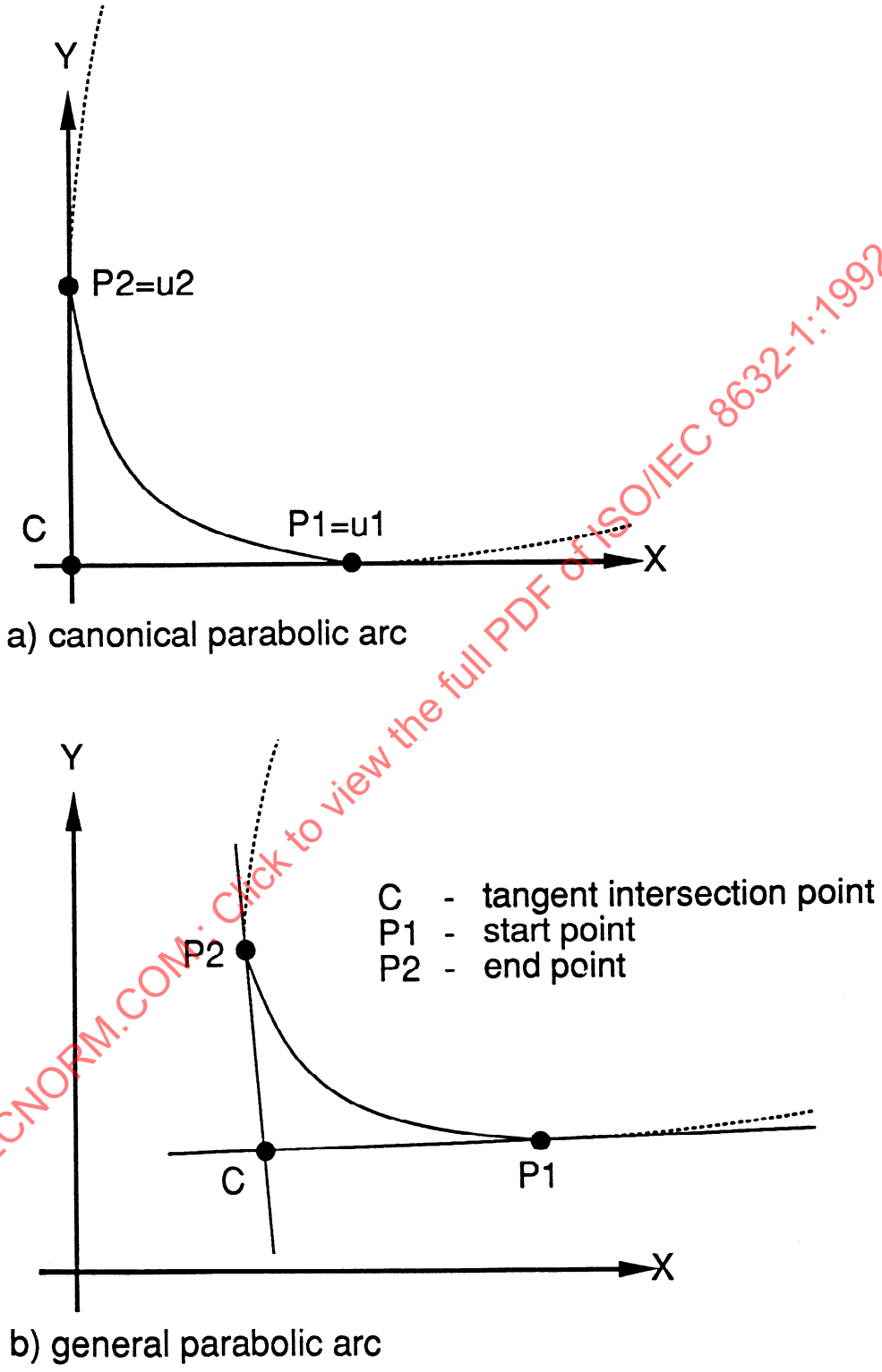


Figure 8 — Example of parabolic arc

4.6.10 Spline curve elements

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

4.6.10.1 Non-uniform B-splines

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

4.6.10.1.1 *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.

4.6.10.1.2 *Mathematical definition.* The non-uniform B-spline is expressed parametrically in the form

Equation 1 — Non-uniform B-spline

$$\mathbf{G}(t) = \sum_{i=0}^{n} \mathbf{P}_i B_i^k(t)$$

where

n — number of control points;
$\mathbf{P}_i$ — control points, (P_{x_i}, P_{y_i});
B_i^k — B-splinc basis functions defined by order k and knot vector T.

The knot vector consists of a non-decreasing sequence of real numbers $(T_1,...,T_{n+k})$ such that $T_i \le T_{i+1}$ for all $i = 1,...,n$.

The curve itself is defined for the range $T_k \le t < T_n$ and can be confined to the range $[T_{\min}, T_{\max}]$, where $T_k \le T_{\min} \le T_{\max} < T_n$. $T_{\min}$ 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:

$$B_i^1 = \begin{cases} 1 & \text{if } T_i \le t \le T_{i+1} \\ 0 & \textit{otherwise} \end{cases}$$

$$B_i^k(t) = \begin{cases} 0, & \text{if } t < t_i \textit{ or } t \ge t_{i+k} \\ \dfrac{(t - T_{i-k}) * B_{i-1}^{k-1}(t)}{T_i - T_{i-k}} + \dfrac{(T_{i+1} - t) * B_i^{k-1}(t)}{T_{i+1} - T_{i-k+1}}, & t_i \le t < t_{i+k} \end{cases}$$

For multiple identical knot values, the fractional terms above may evaluate to $\frac{0}{0}$. 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

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

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

Equation 2 — Rational non-uniform B-spline

$$\mathbf{G}(t) = \frac{\sum_{i=0}^{n} B_i^k(t)\mathbf{P}_i w_i}{\sum_{i=0}^{n} B_i^k(t) w_i}$$

Given a set of control points $\{\mathbf{P}_i\}$ and weights $\{w_i\}$, this is equivalent to creating a set of homogeneous control pointa $\{w_i\mathbf{P}_i, w_i\}$ 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.

4.6.10.2 Polybezier

This element defines one or more cubic Bezier curves.

4.6.10.2.1 *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.

4.6.10.2.2 *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

$$X(t) = A_x t^3 + B_x t^2 + C_x t + X_0$$
$$Y(t) = A_y t^3 + B_y t^2 + C_y t + Y_0$$

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

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

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

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

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

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

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

4.6.11 Closed figures

4.6.11.1 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.

4.6.11.1.1 *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.

4.6.11.1.2 *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.

4.6.11.2 Boundaries and edges

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

4.6.11.2.1 *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.

4.6.11.2.2 *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.

4.6.11.2.3 *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.

4.6.11.3 Contribution of graphical primitive elements to the closed figure

4.6.11.3.1 *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 is the first point of the point list; and the last point is the last point of the point list, as defined in the 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.

4.6.11.3.2 *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.

4.6.11.3.3 *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.

4.6.11.4 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}
```

NOTES

2 Visible edge portion P4..P5 generated.

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}
```

NOTES

5 Visible edge portion P2..P5 generated; edge portions P6..P8 and P9..P11 are drawn with the next two arcs.

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}
```

NOTES

7 Visible edge portion P5..P6 generated.

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}
```

NOTES

9 No edge portion P2..P5 generated.

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

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

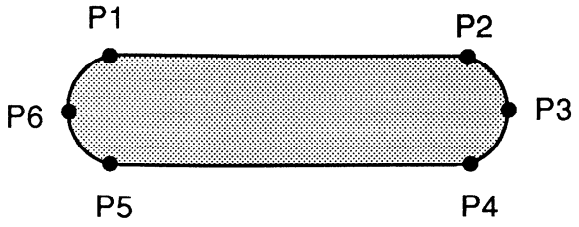


Figure 9a

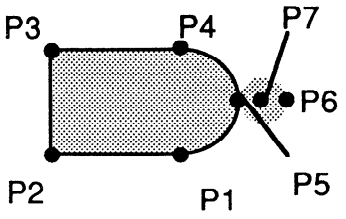


Figure 9b

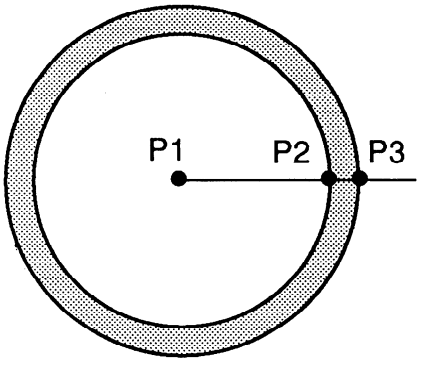


Figure 9c

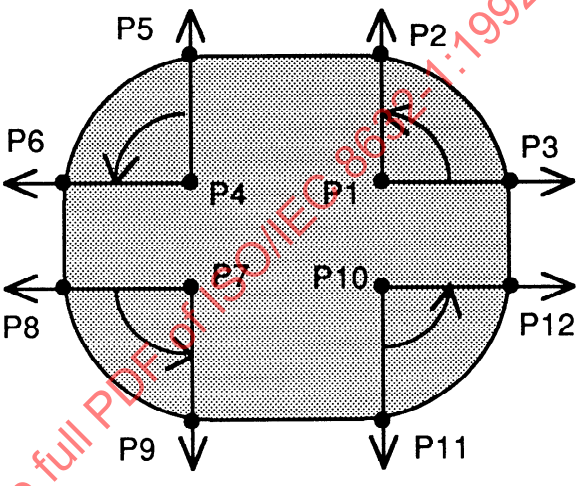


Figure 9d

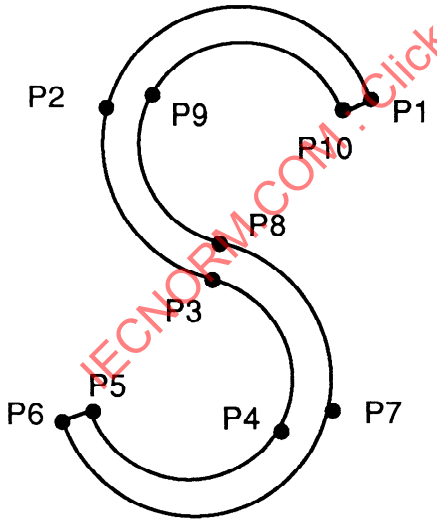


Figure 9e

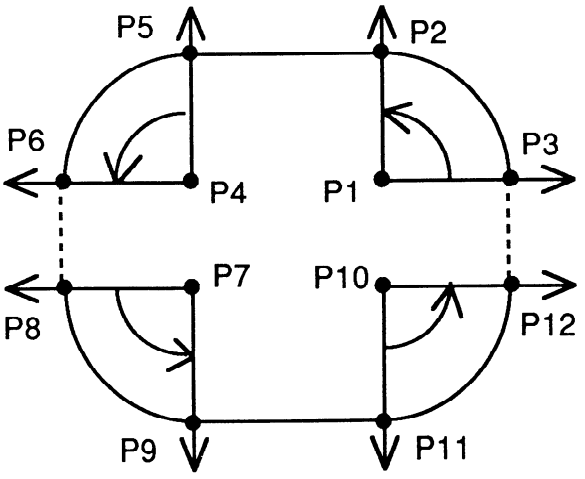


Figure 9f

Figure 9 — Examples of closed figures

4.6.12 Symbol elements

4.6.12.1 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.

4.6.12.2 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 4.7.9.

4.6.12.3 Transformation of symbol elements

The vectors specified by the SYMBOL ORIENTATION element (see 4.7.9) are subject to the VDC-to-Device mapping (see 4.4.7) as well as to both segment and copy transformation (see 4.10.4.2 and 4.10.5).

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

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
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 (note 1) GDP (note 2)	none	END COMPOUND TEXT PATH

NOTES

1 All primitives of the identified classes may be included.

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.

4.7.1 Line attributes

4.7.1.1 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.

4.7.1.2 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 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 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 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.

4.7.2 Marker attributes

4.7.2.1 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.

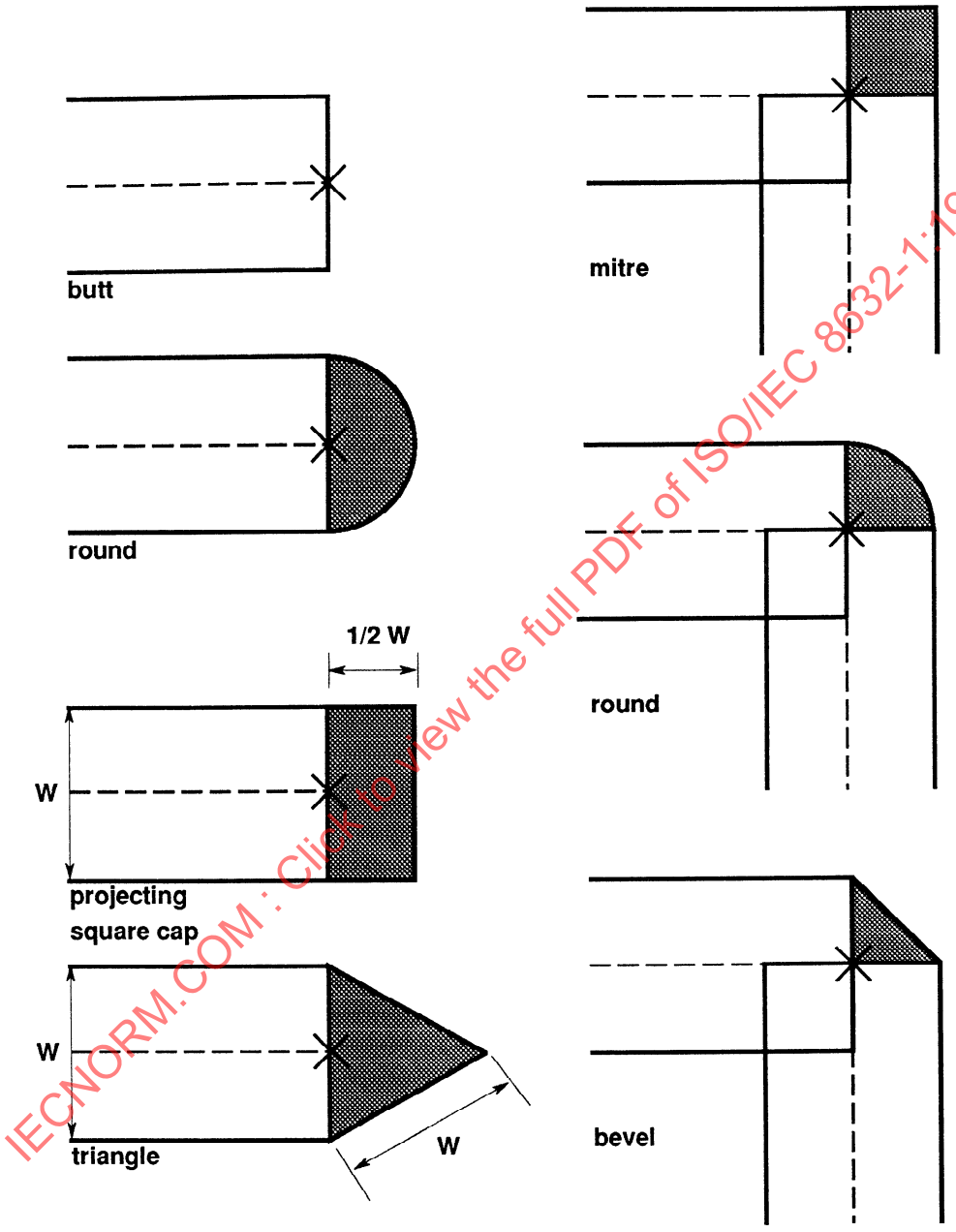


Figure 10 — Examples of LINE CAP and LINE JOIN

4.7.2.2 Individual marker attributes

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

4.7.3 Text attributes

4.7.3.1 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.

4.7.3.2 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'.

NOTE 1

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

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 PATH and 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).

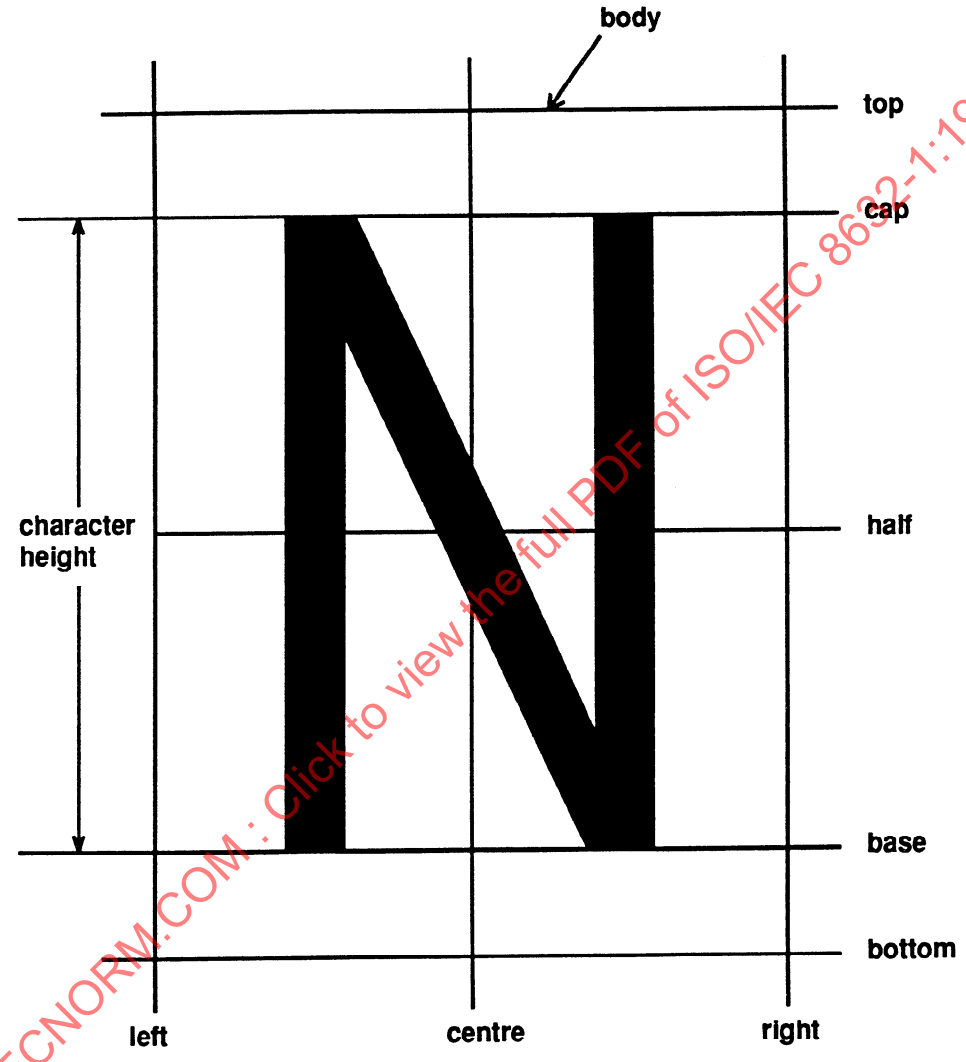


Figure 11 — Font description coordinate system

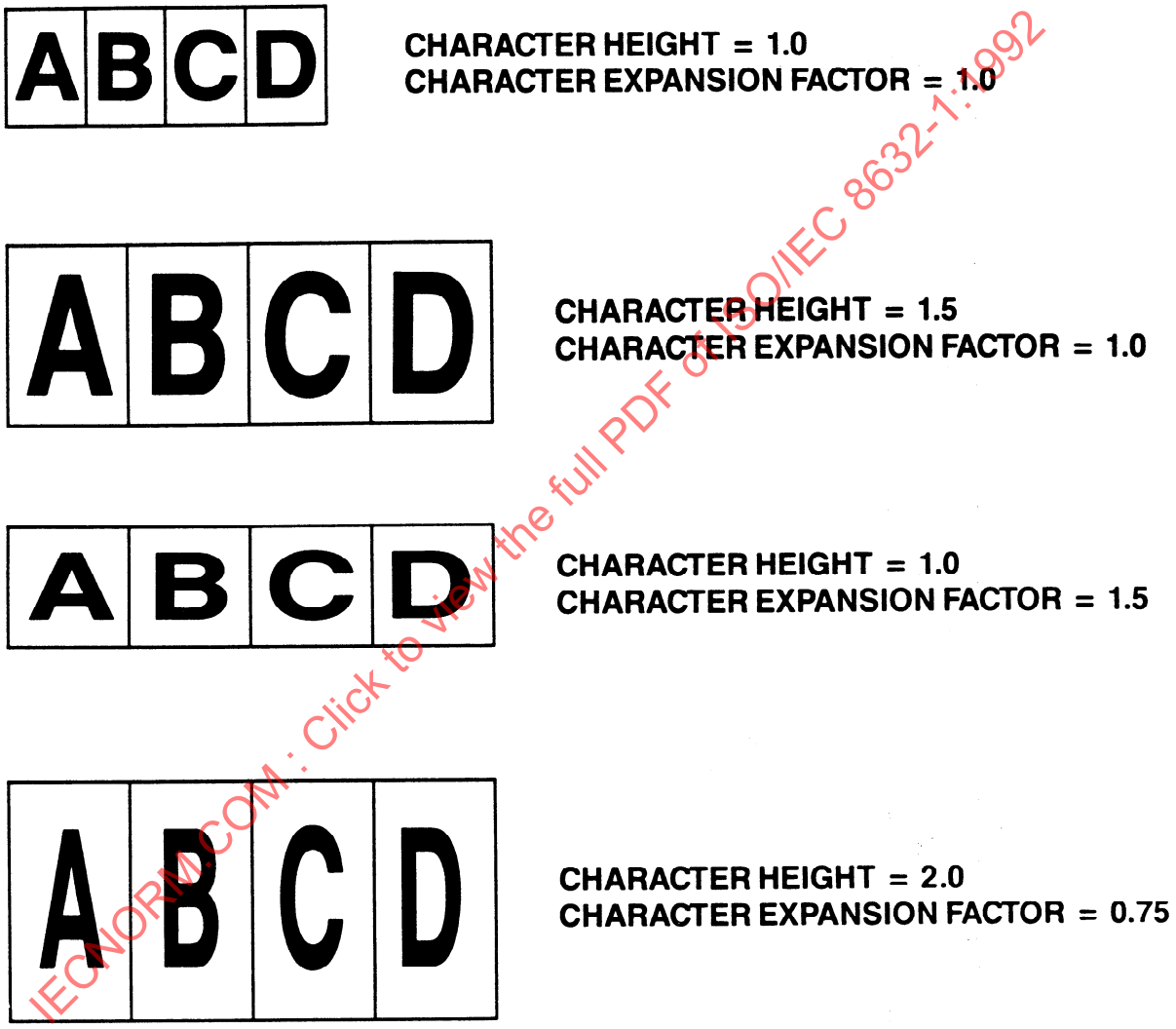


Figure 12 — CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR

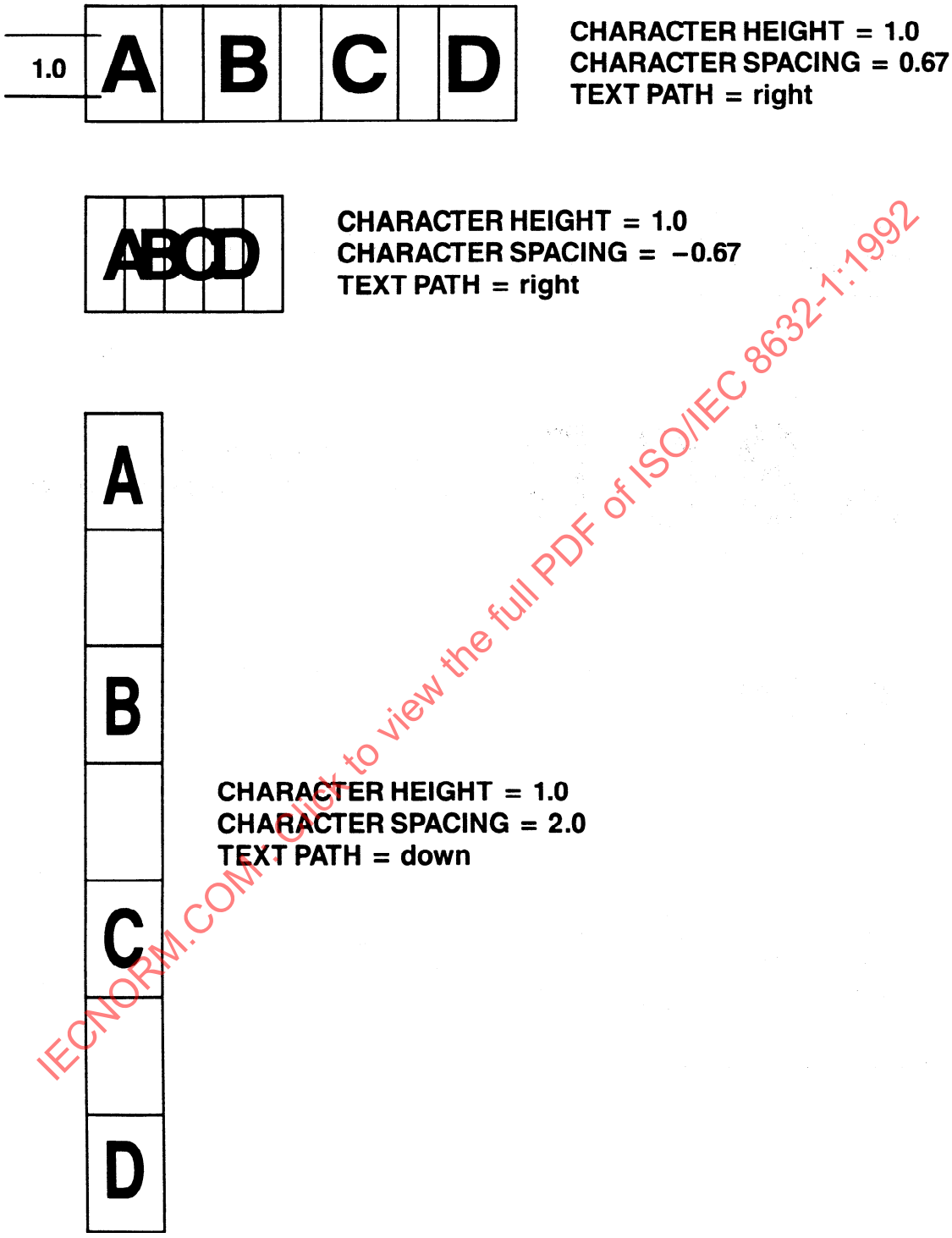


Figure 13 — CHARACTER SPACING

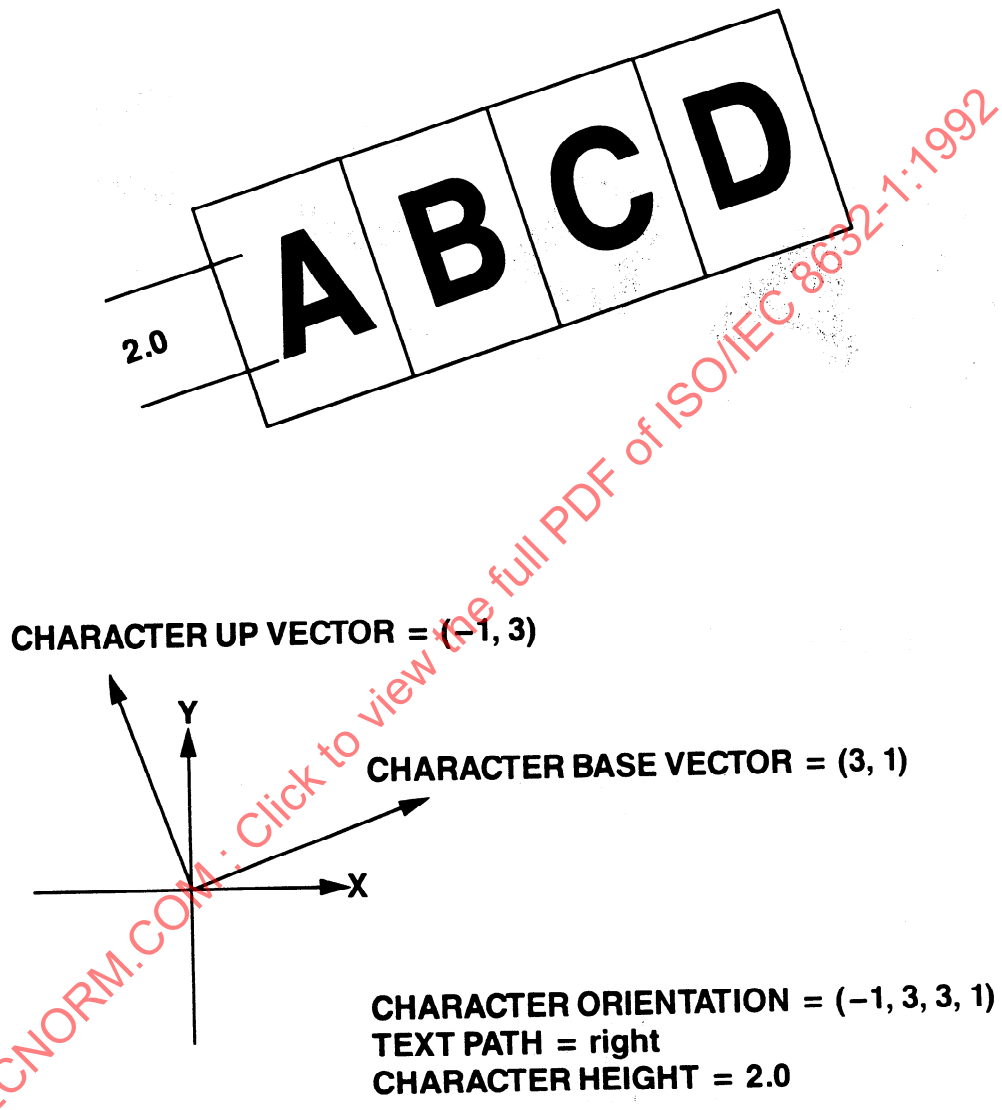


Figure 14 — CHARACTER ORIENTATION

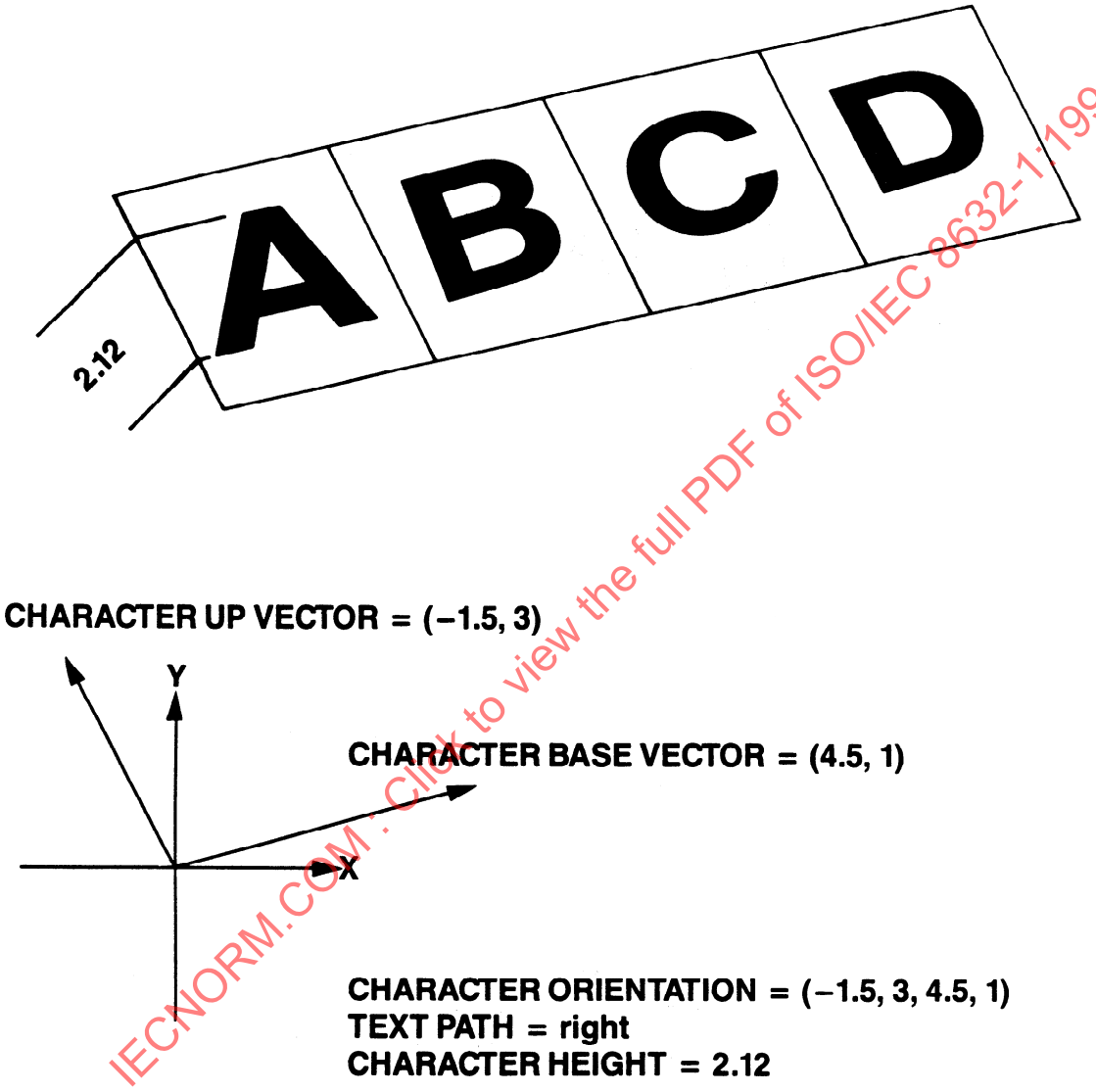


Figure 15 — CHARACTER HEIGHT and CHARACTER ORIENTATION after anisotropic transformation

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

Attribute elements **Concepts**

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.

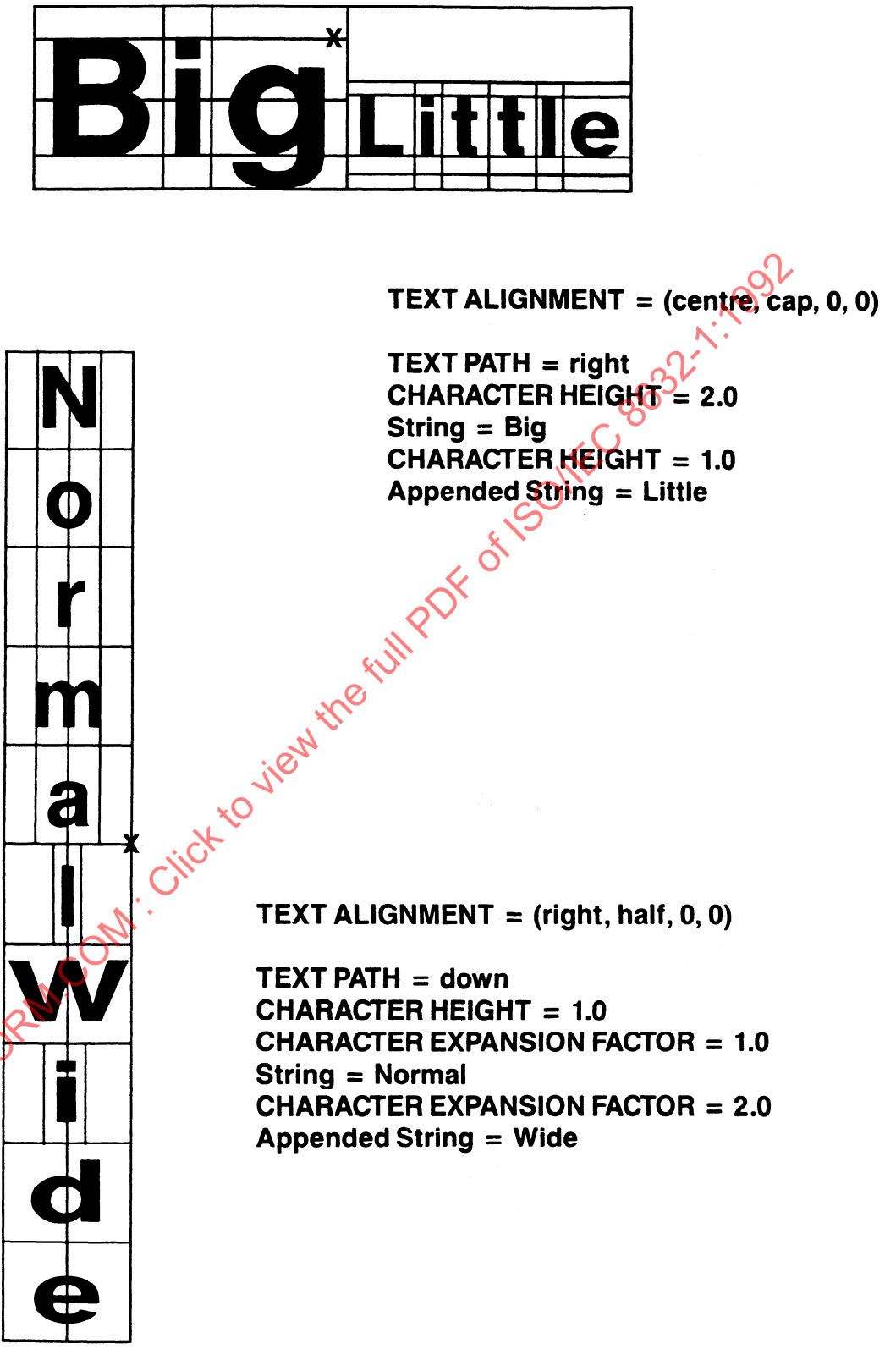


Figure 16 — Discrete text alignment with appended text and proportional spacing

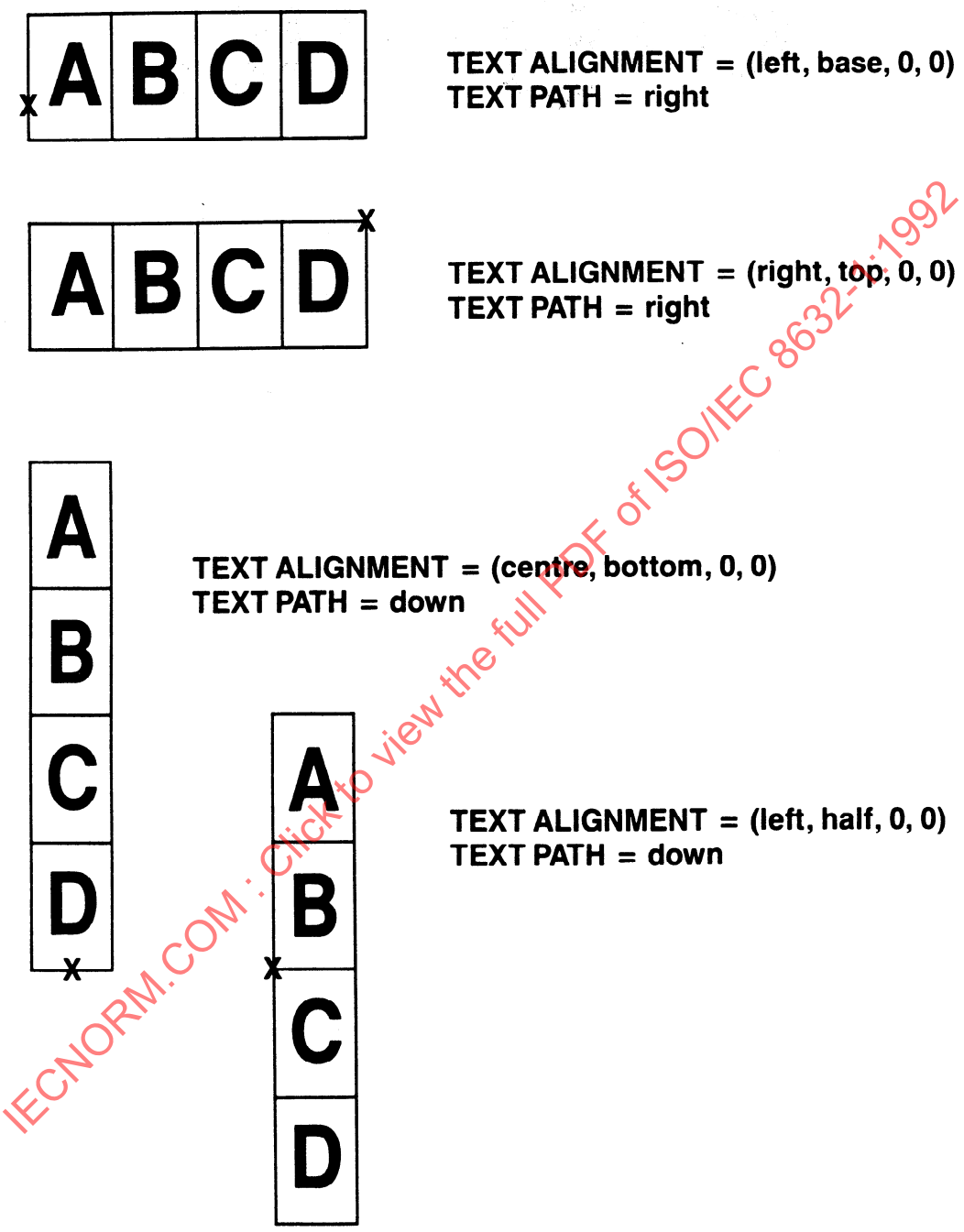
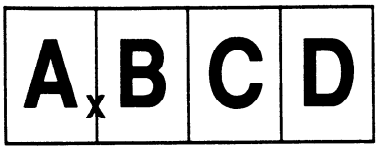


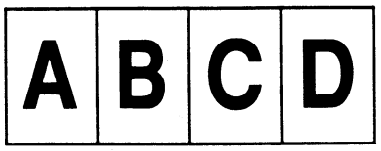
Figure 17 — Discrete text alignment

Concepts

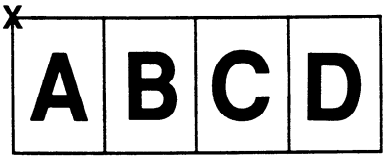
Attribute elements



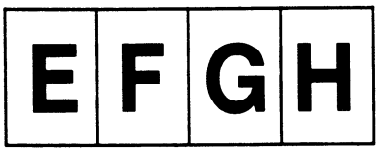
TEXT ALIGNMENT = (continous horizontal, base, 0.25, 0)
TEXT PATH = right



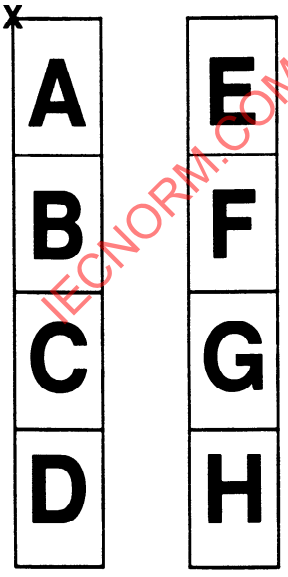
TEXT ALIGNMENT = (continuous horizontal, continuous vertical, −0.25, −0.25)
TEXT PATH = right



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

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 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.

NOTES

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 metafiles. Any of these styles is valid in Version 1 and Version 2 metafiles, however there is no element to select amongst them.

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.

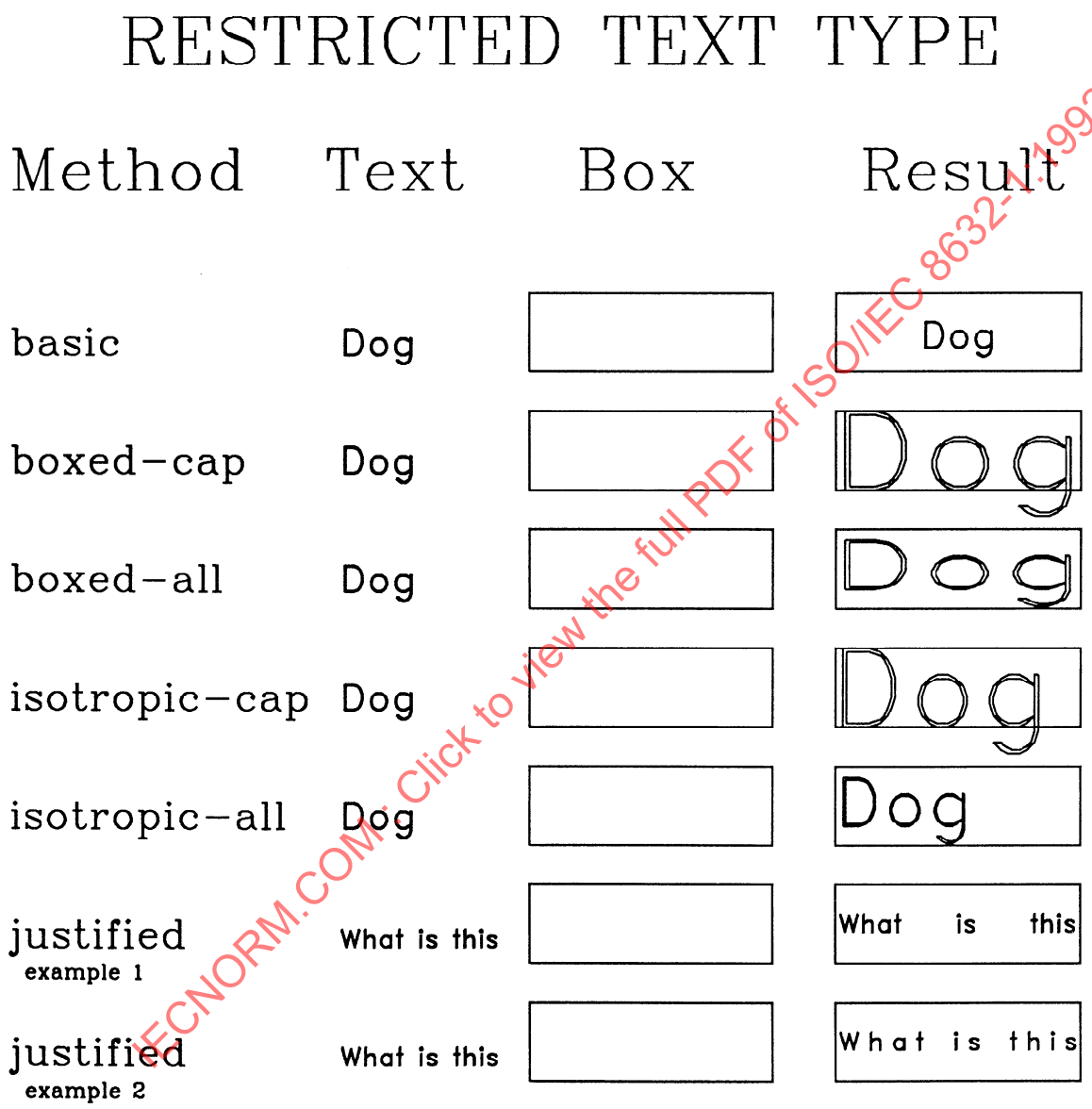


Figure 20 — Examples of RESTRICTED TEXT TYPE

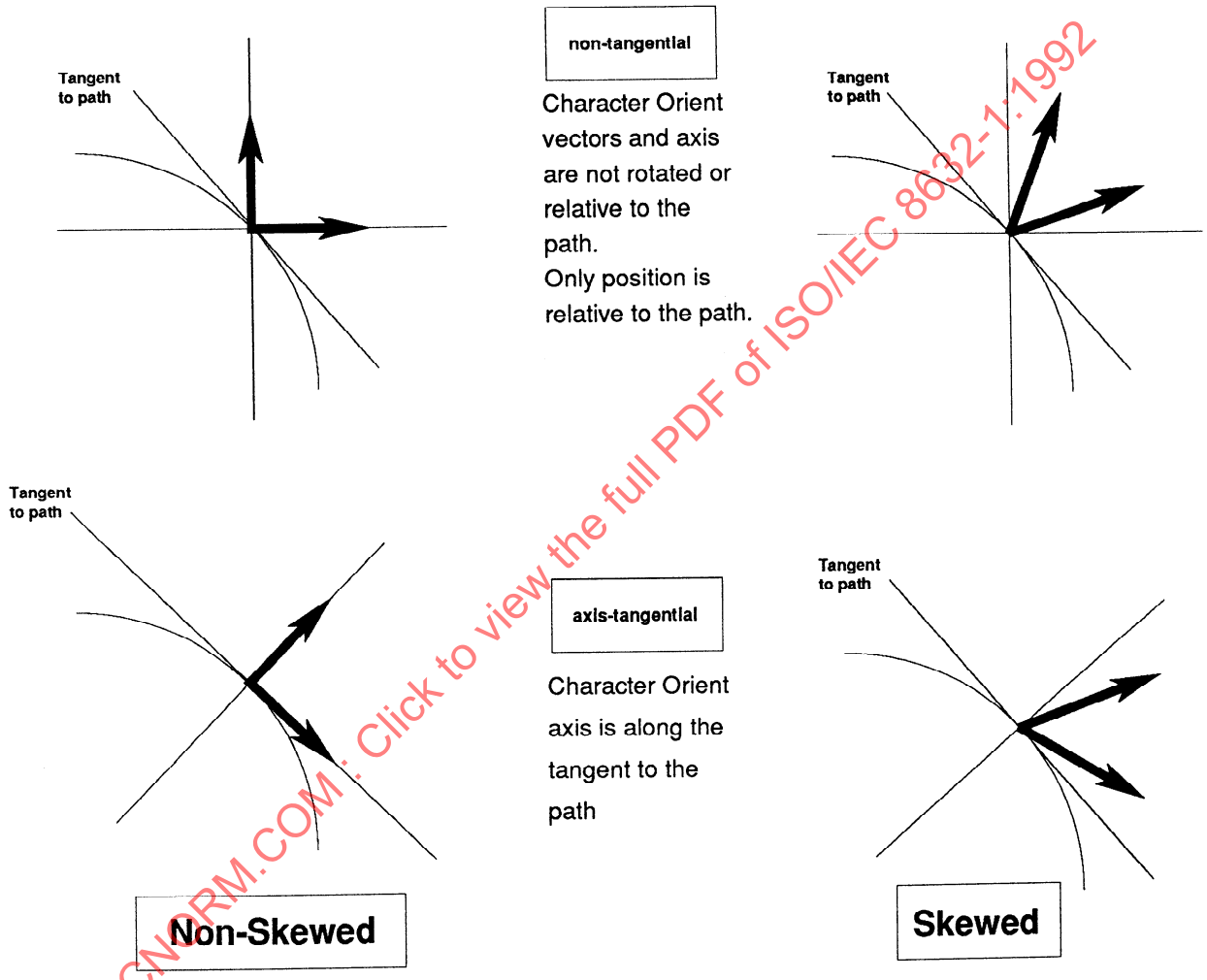


Figure 21 — Illustration of the modes of GENERALIZED TEXT PATH MODE

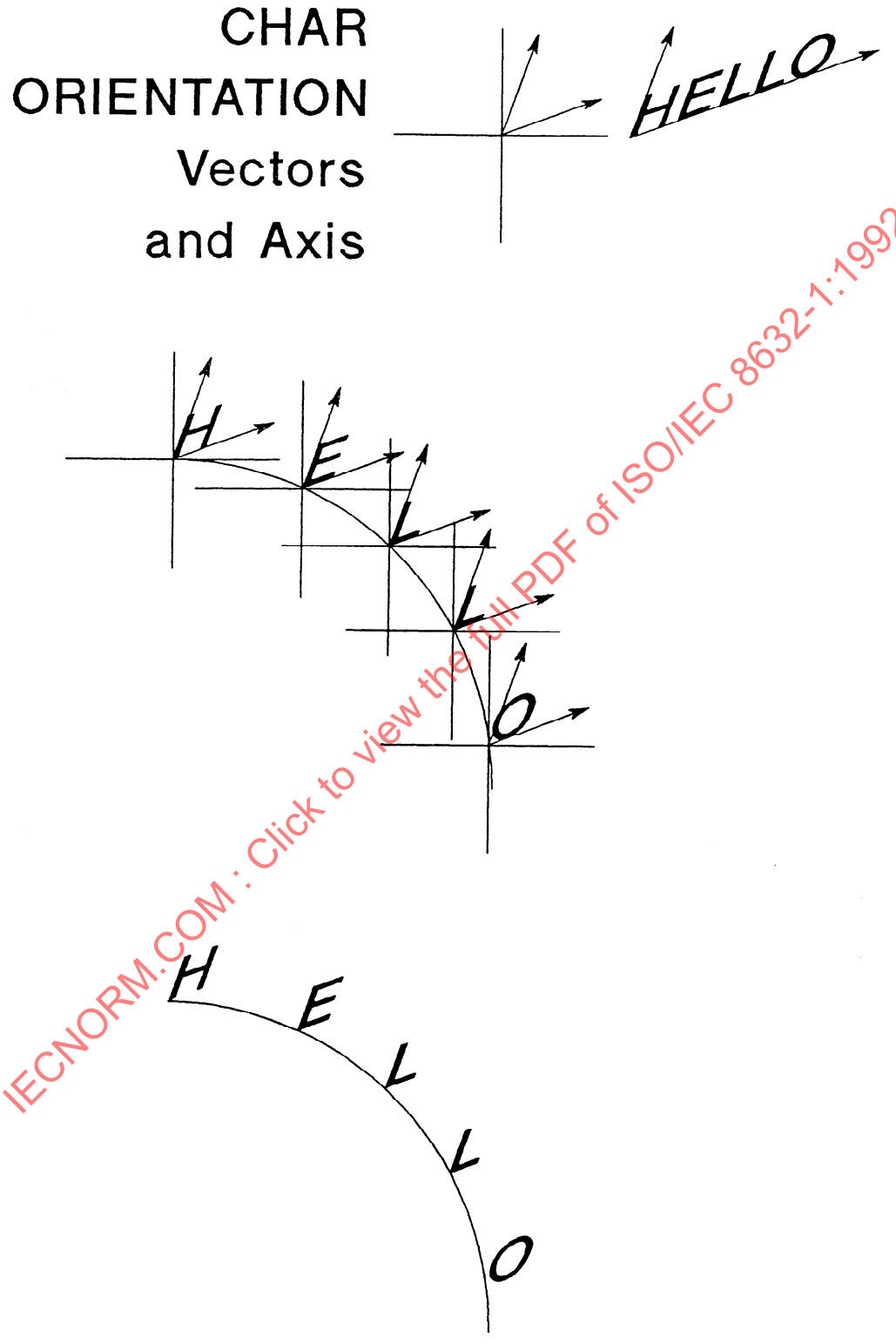


Figure 22 — Examples of GENERALIZED TEXT PATH MODE, non-tangential

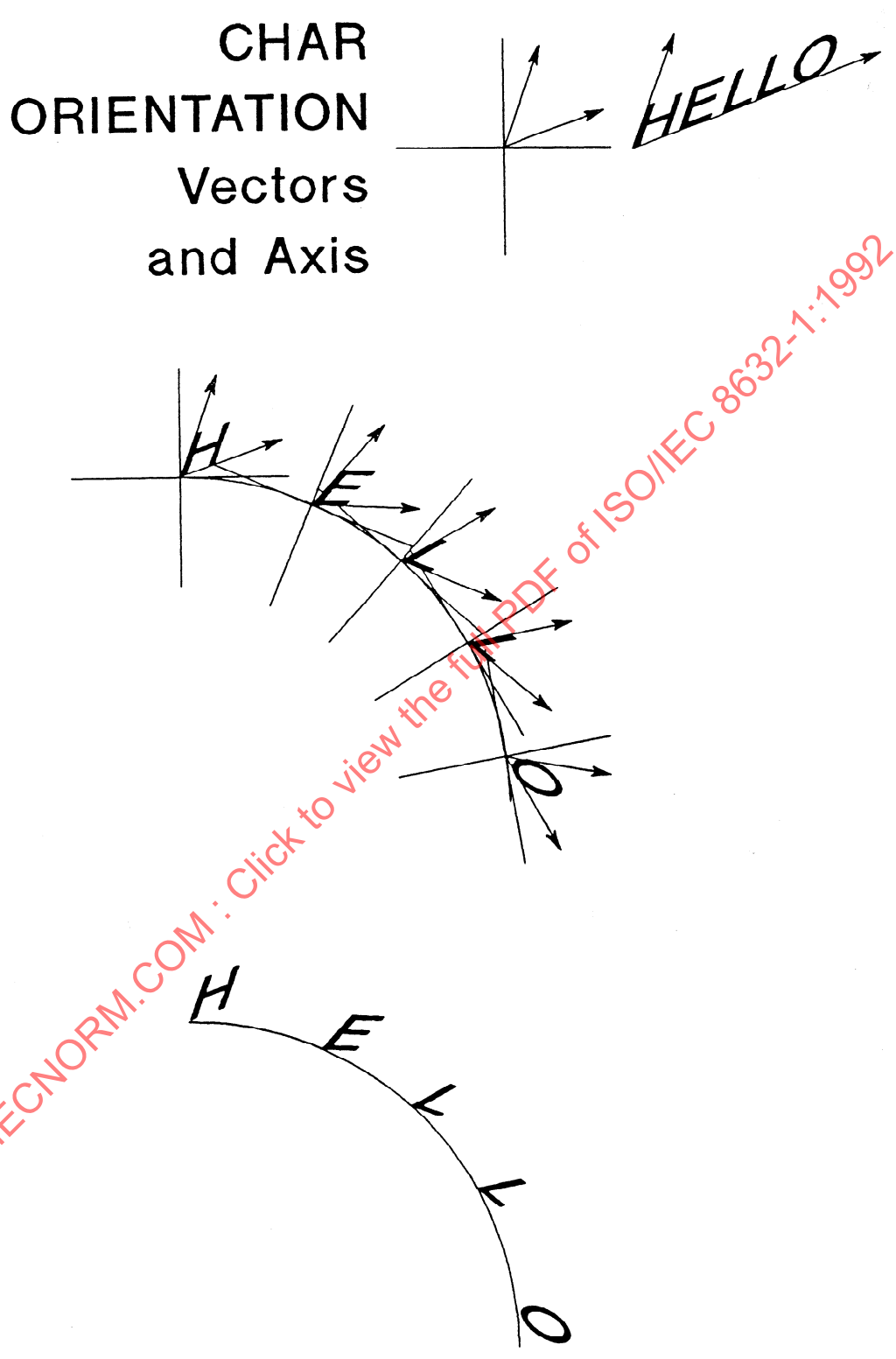


Figure 23 — Examples of GENERALIZED TEXT PATH MODE, axis-tangential

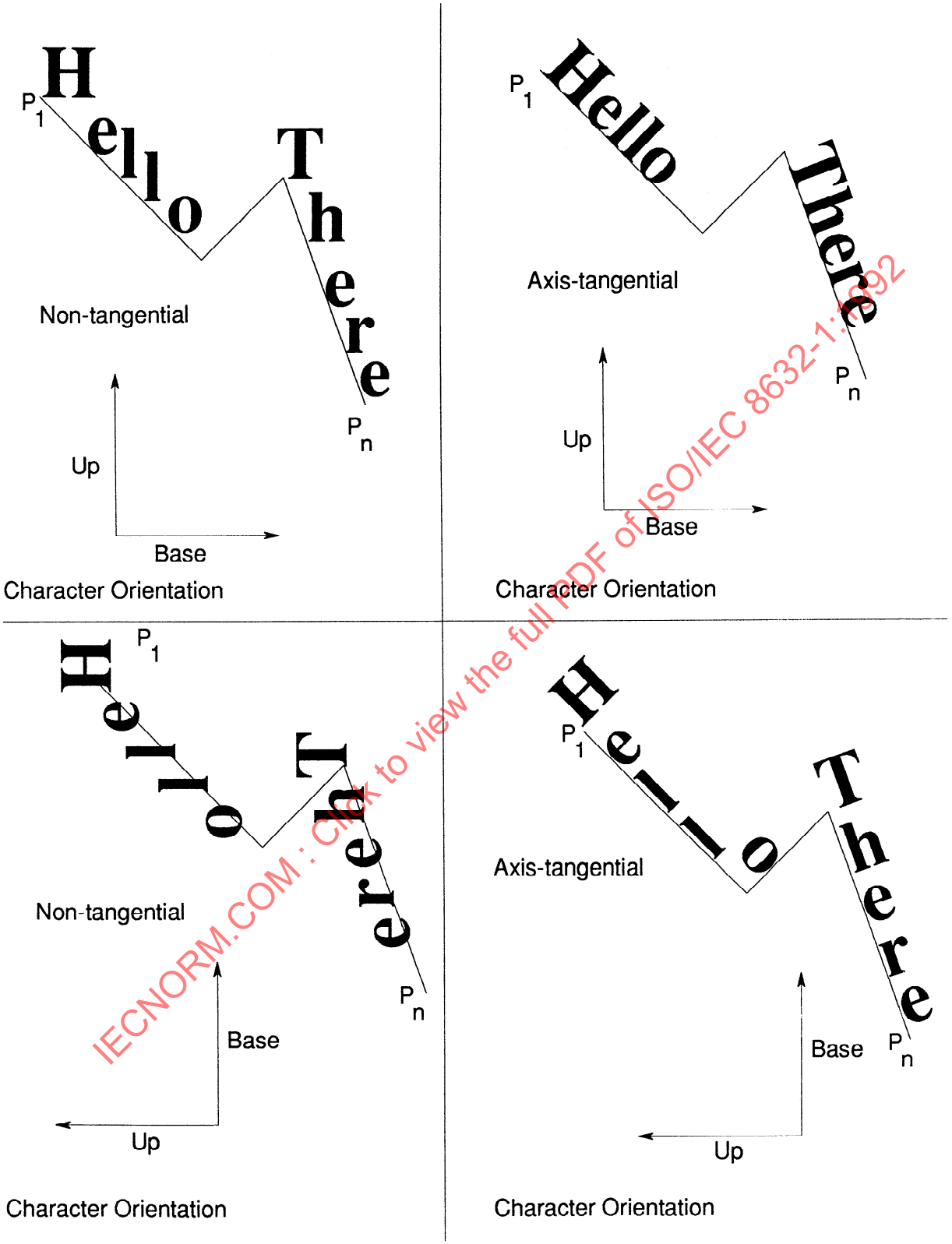


Figure 24 — GENERALIZED TEXT PATH MODE and unsmooth paths

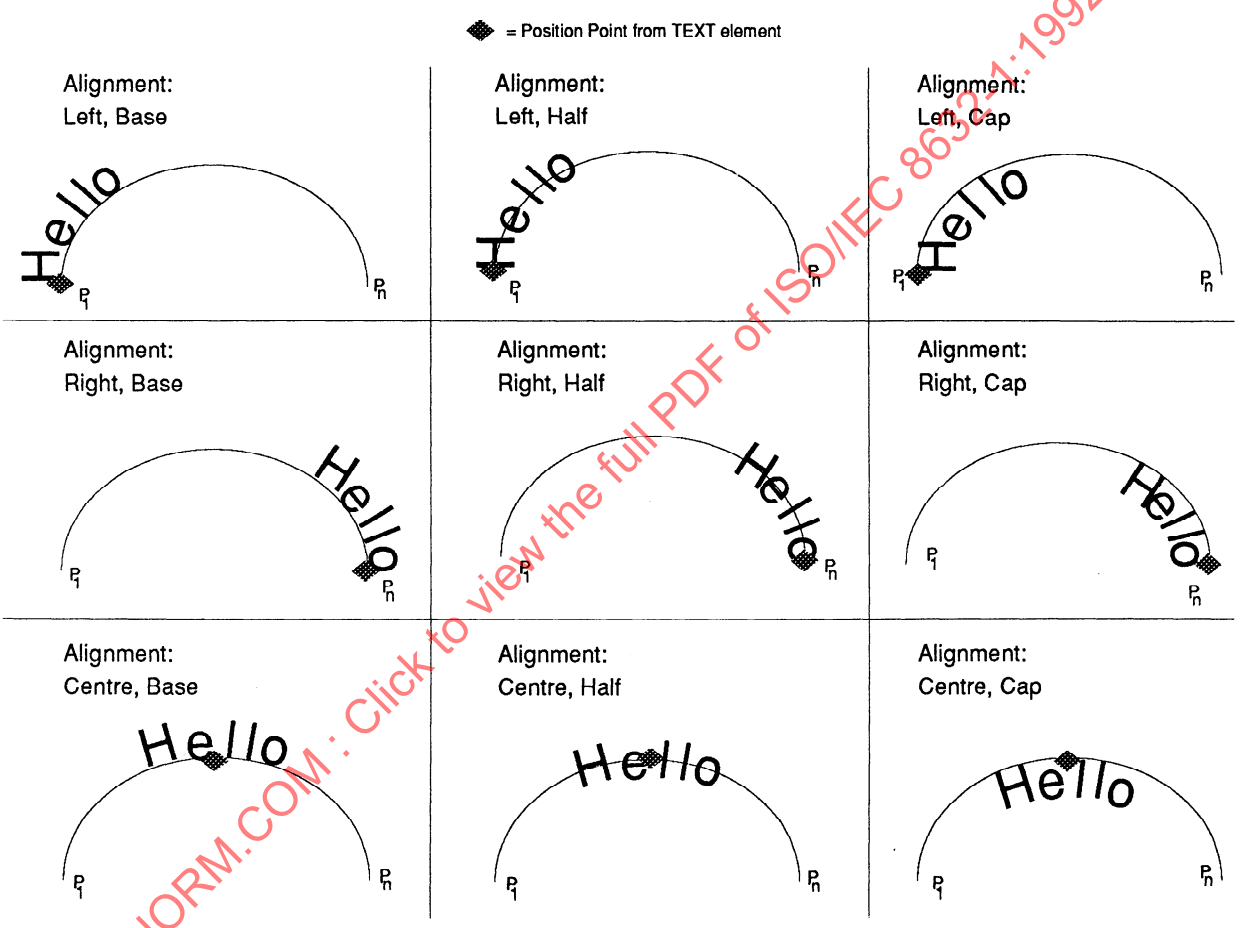


Figure 25 — Effect of TEXT ALIGNMENT and path text

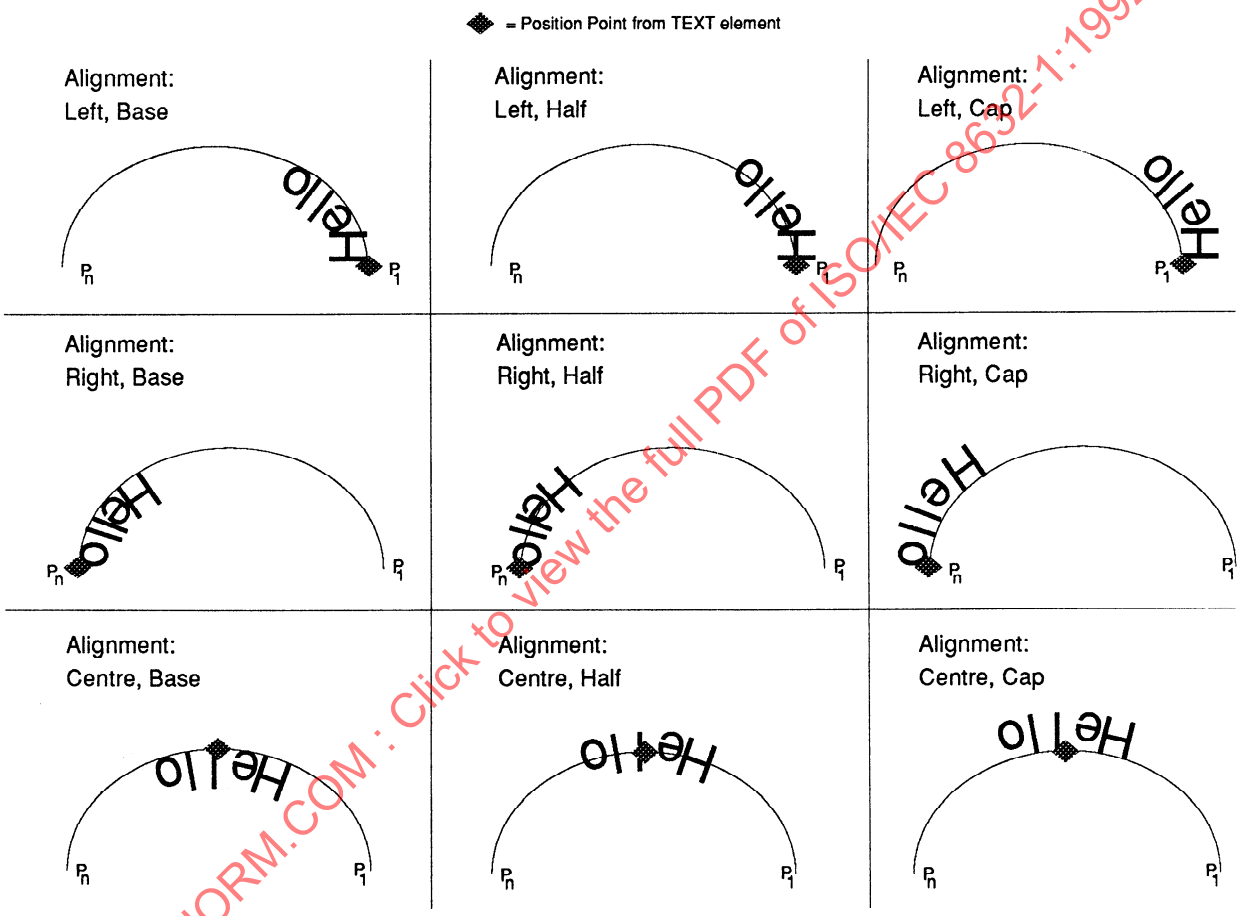


Figure 26 — Effect of path direction and TEXT ALIGNMENT with path text

4.7.4 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.

4.7.4.1 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.

4.7.4.2 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.

4.7.4.3 Individual fill attributes and usage of fill attributes

4.7.4.3.1 *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.

4.7.4.3.2 *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.

4.7.4.3.3 *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.

4.7.4.4 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.

4.7.5 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 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.

4.7.6 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 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 CCIR601 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.

NOTES

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.

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.

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, and 3, as well

as in the Picture Descriptor for Version 2 and Version 3 metafiles.

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). The primitive is bound to the definition of the colour index at the time that it is bound to the index.

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.

4.7.7 Pick identifier

The pick identifier is associated with graphical primitive elements within segments (see 4.10). 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.

4.7.8 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 1 Using paths with sharp curvature is likely to cause overlapping tops or bottoms of characters unless care is taken in adjusting the text attributes.

4.7.9 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.

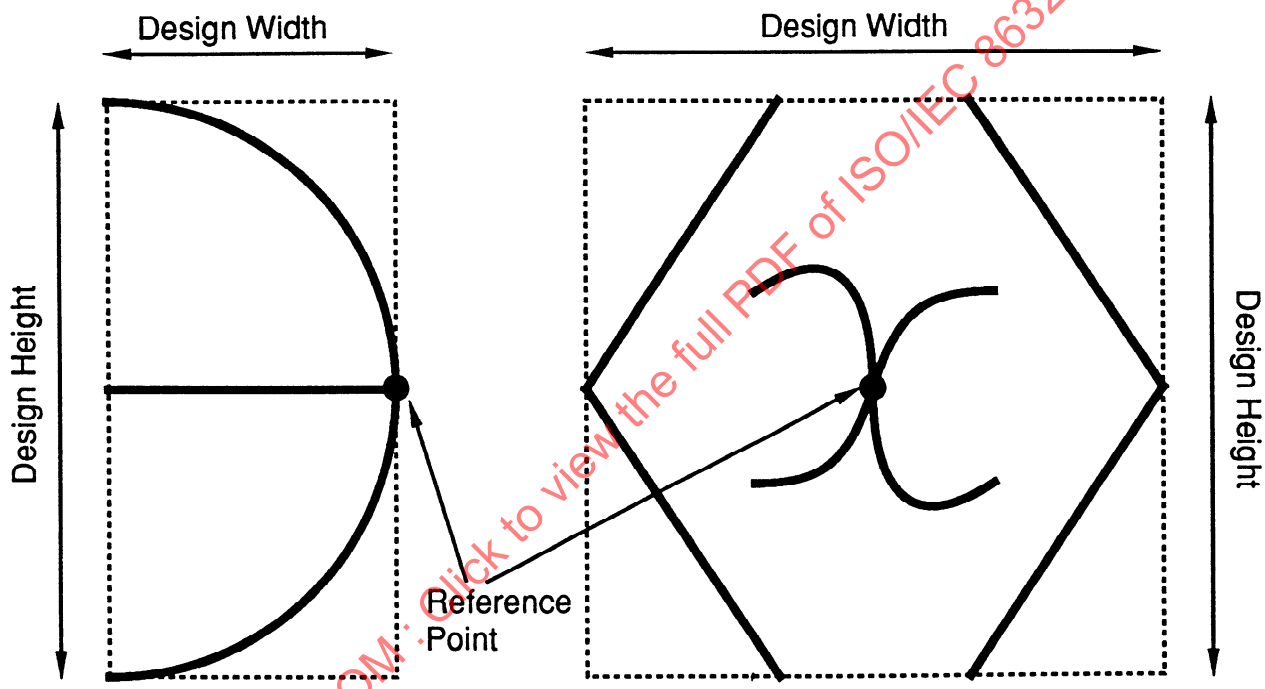


Figure 27 — Symbol coordinate system

4.8 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.

4.9 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.

4.10 Segment elements

4.10.1 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.

4.10.2 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.

4.10.2.1 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 4.10.5) element. The COPY SEGMENT element incorporates the segment into the open picture in the same way for both local and global segments.

4.10.2.2 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.

4.10.2.3 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.

4.10.2.4 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 6 of this part of ISO/IEC 8632) or explicit (that is, by values set in the Metafile Defaults Replacement).

4.10.3 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.

4.10.4 Segment attributes

4.10.4.1 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

4.10.4.2 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.

4.10.4.3 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.

4.10.4.4 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.

4.10.4.5 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.

4.10.5 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

Table 6 — Inheritance filter selection names for attributes (concluded).

Attribute Group Name	Individual Attribute Name
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
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)
```

Segment elements	Concepts
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'.

4.11 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 9.

The states in which each element is allowed for Version 1, Version 2, and Version 3 metafiles are described in table 8. This table also shows 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 5 for the minor metafile states (see 5.1).

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 8 — CGM Elements by their allowed states

CGM Element	ver (1)	CGM Major States						
		PCS	MDS	DR (3)	GSS, DSS	PDS	POS	LSS
		v1(2)	v1	v1	v2	v1	v1	v2
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(7), v2	2		X				X	
BEGIN SEGMENT, v3	2		X			X	X	
END SEGMENT	2				X			X
BEGIN FIGURE	2				X		X	X
END FIGURE	2							
BEGIN PROTECTION REGION	3				X		X	X
END PROTECTION REGION	3							
BEGIN COMPOUND LINE	3				X		X	X
END COMPOUND LINE	3							
BEGIN COMPOUND TEXT PATH	3				X		X	X
END COMPOUND TEXT PATH	3							
BEGIN TILE ARRAY	3						X	
END TILE ARRAY	3							
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					

Table 8 — CGM Elements by their allowed states (continued)

CGM Element	ver (1)	CGM Major States						
		PCS	MDS	DR (3)	GSS, DSS	PDS	POS	LSS
		v1(2)	v1	v1	v2	v1	v1	v2
SCALING MODE	1			X		X		
COLOUR SELECTION MODE(7), v1	1			X		X		
COLOUR SELECTION MODE, v2/3	1			X	X	X	X	X
LINE WIDTH SPECIFICATION MODE(7), v1	1			X		X		
LINE WIDTH SPECIFICATION MODE, v2/3	1			X	X	X	X	X
MARKER SIZE SPECIFICATION MODE(7), v1	1			X		X		
MARKER SIZE SPECIFICATION MODE, v2/3	1			X	X	X	X	X
EDGE WIDTH SPECIFICATION MODE(7), v1	1			X		X		
EDGE WIDTH SPECIFICATION MODE, v2/3	1			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
LINE AND EDGE TYPE DEFINITION	3			X		X		
HATCH STYLE DEFINITION	3			X		X		
GEOMETRIC PATTERN DEFINITION	3			X		X		
VDC INTEGER PRECISION	1			X	X		X	X
VDC REAL PRECISION	1			X	X		X	X
AUXILIARY COLOUR	1			X	X		X	X
TRANSPARENCY	1			X	X		X	X
CLIP RECTANGLE	1			X	X		X	X
CLIP INDICATOR	1			X	X		X	X
LINE CLIPPING MODE	2			X	X		X	X
MARKER CLIPPING MODE	2			X	X		X	X
EDGE CLIPPING MODE	2			X	X		X	X
NEW REGION	2							
SAVE PRIMITIVE CONTEXT	2				X		X	X
RESTORE PRIMITIVE CONTEXT	2				X		X	X
PROTECTION REGION INDICATOR	3			X	X		X	X
GENERALIZED TEXT PATH MODE	3			X	X		X	X
MITRE LIMIT	3			X	X		X	X
TRANSPARENT CELL COLOUR	3			X	X	X	X	X

Table 8 — CGM Elements by their allowed states (continued)

CGM Element	ver (1)	CGM Major States						
		PCS	MDS	DR (3)	GSS, DSS	PDS	POS	LSS
		v1(2)	v1	v1	v2	v1	v1	v2
POLYLINE	1				X		X	X
DISJOINT POLYLINE	1				X		X	X
POLYMARKER	1				X		X	X
TEXT	1				X		X	X
RESTRICTED TEXT	1				X		X	X
APPEND TEXT	1							
POLYGON	1				X		X	X
POLYGON SET	1				X		X	X
CELL ARRAY	1				X		X	X
GDP	1				X		X	X
RECTANGLE	1				X		X	X
CIRCLE	1				X		X	X
CIRCULAR ARC 3 POINT	1				X		X	X
CIRCULAR ARC 3 POINT CLOSE	1				X		X	X
CIRCULAR ARC CENTRE	1				X		X	X
CIRCULAR ARC CENTRE CLOSE	1				X		X	X
ELLIPSE	1				X		X	X
ELLIPTICAL ARC	1				X		X	X
ELLIPTICAL ARC CLOSE	1				X		X	X
CIRCULAR ARC CENTRE REVERSED	2				X		X	X
CONNECTING EDGE	2							
HYPERBOLIC ARC	3				X		X	X
PARABOLIC ARC	3				X		X	X
NON-UNIFORM B-SPLINE	3				X		X	X
NON-UNIFORM RATIONAL B-SPLINE	3				X		X	X
POLYBEZIER	3				X		X	X
POLYSYMBOL	3				X		X	X
BITONAL TILE	3							
TILE	3							
LINE BUNDLE INDEX	1			X	X		X	X
LINE TYPE	1			X	X		X	X
LINE WIDTH	1			X	X		X	X
LINE COLOUR	1			X	X		X	X
MARKER BUNDLE INDEX	1			X	X		X	X
MARKER TYPE	1			X	X		X	X
MARKER SIZE	1			X	X		X	X
MARKER COLOUR	1			X	X		X	X
TEXT BUNDLE INDEX	1			X	X		X	X
TEXT FONT INDEX	1			X	X		X	X
TEXT PRECISION	1			X	X		X	X
CHARACTER EXPANSION FACTOR	1			X	X		X	X
CHARACTER SPACING	1			X	X		X	X
TEXT COLOUR	1			X	X		X	X
CHARACTER HEIGHT	1			X	X		X	X
CHARACTER ORIENTATION	1			X	X		X	X

Table 8 — CGM Elements by their allowed states (continued)

CGM Element	ver (1)	CGM Major States						
		PCS	MDS	DR (3)	GSS, DSS	PDS	POS	LSS
		v1(2)	v1	v1	v2	v1	v1	v2
TEXT PATH	1			X	X		X	X
TEXT ALIGNMENT	1			X	X		X	X
CHARACTER SET INDEX	1			X	X		X	X
ALTERNATE CHARACTER SET INDEX	1			X	X		X	X
FILL BUNDLE INDEX	1			X	X		X	X
INTERIOR STYLE	1			X	X		X	X
FILL COLOUR	1			X	X		X	X
HATCH INDEX	1			X	X		X	X
PATTERN INDEX	1			X	X		X	X
EDGE BUNDLE INDEX	1			X	X		X	X
EDGE TYPE	1			X	X		X	X
EDGE WIDTH	1			X	X		X	X
EDGE COLOUR	1			X	X		X	X
EDGE VISIBILITY	1			X	X		X	X
FILL REFERENCE POINT	1			X	X		X	X
PATTERN TABLE(7), v1	1			X			X	
PATTERN TABLE, v2/3	1			X		X	X	
COLOUR TABLE(7), v1	1			X			X	
COLOUR TABLE, v2/3	1			X		X	X	
ASPECT SOURCE FLAGS	1			X	X		X	X
PICK IDENTIFIER	2			X	X		X	X
LINE CAP	3			X	X		X	X
LINE JOIN	3			X	X		X	X
LINE TYPE CONTINUATION	3			X	X		X	X
LINE TYPE INITIAL OFFSET	3			X	X		X	X
TEXT SCORE TYPE	3			X	X		X	X
RESTRICTED TEXT TYPE	3			X	X		X	X
INTERPOLATED INTERIOR	3			X	X		X	X
EDGE CAP	3			X	X		X	X
EDGE JOIN	3			X	X		X	X
EDGE TYPE CONTINUATION	3			X	X		X	X
EDGE TYPE INITIAL OFFSET	3			X	X		X	X
SYMBOL LIBRARY INDEX	3			X	X		X	X
SYMBOL COLOUR	3			X	X		X	X
SYMBOL SIZE	3			X	X		X	X
SYMBOL ORIENTATION	3			X	X		X	X
ESCAPE	1	X	X	X	X	X	X	X
MESSAGE	1	X	X	X	X	X	X	X
APPLICATION DATA	1	X	X	X	X	X	X	X
COPY SEGMENT	2				X		X	X
INHERITANCE FILTER	2			X	X		X	X
CLIP INHERITANCE	2			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

Table 8 — CGM Elements by their allowed states (continued)

CGM Element	CGM Minor States				
	FOS v2	TOS v1	CPS v3	PRS v3	TAS v3
BEGIN METAFILE END METAFILE BEGIN PICTURE BEGIN PICTURE BODY END PICTURE					
BEGIN SEGMENT END SEGMENT BEGIN FIGURE END FIGURE BEGIN PROTECTION REGION	X				
END PROTECTION REGION BEGIN COMPOUND LINE END COMPOUND LINE BEGIN COMPOUND TEXT PATH END COMPOUND TEXT PATH			X(5) X(6)	X	
BEGIN TILE ARRAY END TILE ARRAY METAFILE VERSION METAFILE DESCRIPTION VDC TYPE					X
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					

Table 8 — CGM Elements by their allowed states (continued)

CGM Element	CGM Minor States				
	FOS v2	TOS v1	CPS v3	PRS v3	TAS v3
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 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 VDC INTEGER PRECISION	X		X	X	
VDC REAL PRECISION AUXILIARY COLOUR TRANSPARENCY CLIP RECTANGLE CLIP INDICATOR	X X X	X X	X	X	
LINE CLIPPING MODE MARKER CLIPPING MODE EDGE CLIPPING MODE NEW REGION SAVE PRIMITIVE CONTEXT	X			X	
RESTORE PRIMITIVE CONTEXT PROTECTION REGION INDICATOR GENERALIZED TEXT PATH MODE MITRE LIMIT TRANSPARENT CELL COLOUR					

Table 8 — CGM Elements by their allowed states (continued)

CGM Element	CGM Minor States				
	FOS v2	TOS v1	CPS v3	PRS v3	TAS v3
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	
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					

Table 8 — CGM Elements by their allowed states (concluded)

CGM Element	CGM Minor States				
	FOS v2	TOS v1	CPS v3	PRS v3	TAS v3
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					
COLOUR TABLE					
ASPECT SOURCE FLAGS	X				
PICK IDENTIFIER					
LINE CAP					
LINE JOIN					
LINE TYPE CONTINUATION					
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
COPY SEGMENT					
INHERITANCE FILTER					
CLIP INHERITANCE					
SEGMENT TRANSFORMATION					
SEGMENT HIGHLIGHTING					
SEGMENT DISPLAY PRIORITY					
SEGMENT PICK PRIORITY					

Notes on the state tables:

1. The "ver" column in the tables indicates the lowest metafile version in which the element may appear.
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 and Version 3 metafiles); "v2" indicates that the state is defined for Version 2 metafiles (hence also for Version 3); and "v3" indicates that the state is defined only for Version 3 metafiles.

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.
4. The Metafile Closed State is not included in this table — BEGIN METAFILE is the only element allowed in this state.
5. END COMPOUND LINE is only allowed in Compound Path State when that state was entered by the BEGIN COMPOUND LINE element.
6. END COMPOUND TEXT PATH is only allowed in Compound Path State when that state was entered by the BEGIN COMPOUND TEXT PATH element.
7. 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 and Version 3 metafiles.

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

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	GSS,LSS,POS	FOS
END FIGURE	FOS	previous state
TEXT (not-final)	GSS,LSS,POS	TOS
RESTRICTED TEXT (not-final)	GSS,LSS,POS	TOS
APPEND TEXT (final)	TOS	previous state
BEGIN COMPOUND LINE	GSS,LSS,POS	CPS
END COMPOUND LINE	CPS	previous state
BEGIN COMPOUND TEXT PATH	GSS,LSS,POS	CPS
END COMPOUND TEXT PATH	CPS	previous state
BEGIN PROTECTION REGION	GSS,LSS,POS	PRS
END PROTECTION REGION	PRS	previous state
BEGIN TILE ARRAY	POS	TAS
END TILE ARRAY	TAS	previous state

4.12 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 TYPE	HATCH STYLE
COLOUR CALIBRATION	LINE CAP	EDGE TYPE
FONT PROPERTIES	LINE JOIN	EDGE CAP
GLYPH MAPPING	LINE TYPE CONTINUATION	EDGE JOIN
SYMBOL LIBRARY LIST	MARKER TYPE	EDGE TYPE CONTINUATION
BITONAL TILE	TEXT SCORE TYPE	GENERALIZED DRAWING PRIMITIVE
TILE	RESTRICTED TEXT TYPE	ESCAPE

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).

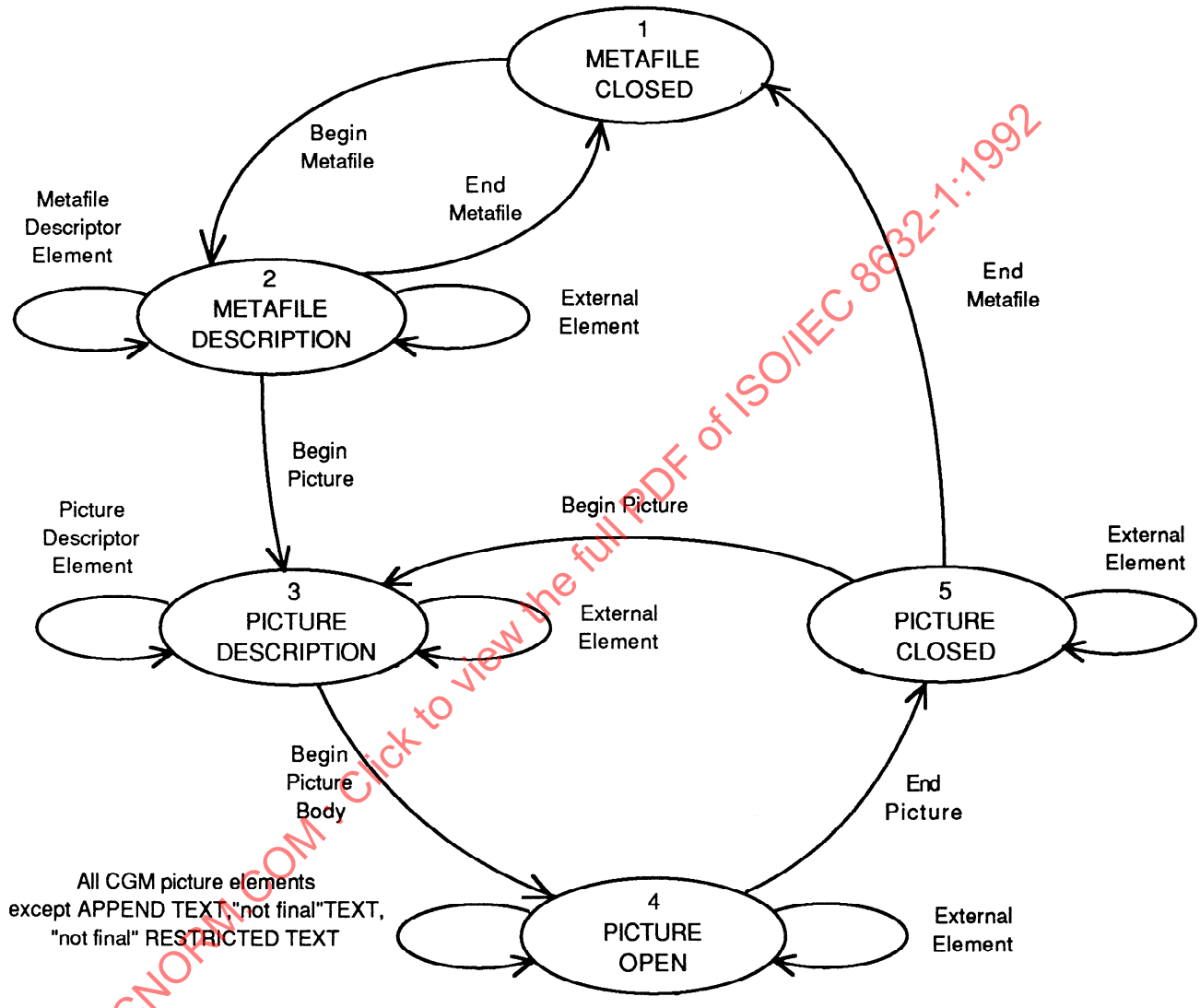


Figure 28 — State diagram for Version 1 metafiles

NOTE 1 "External Element" in figure 28 denotes the set containing both the Escape Elements and the External Elements.

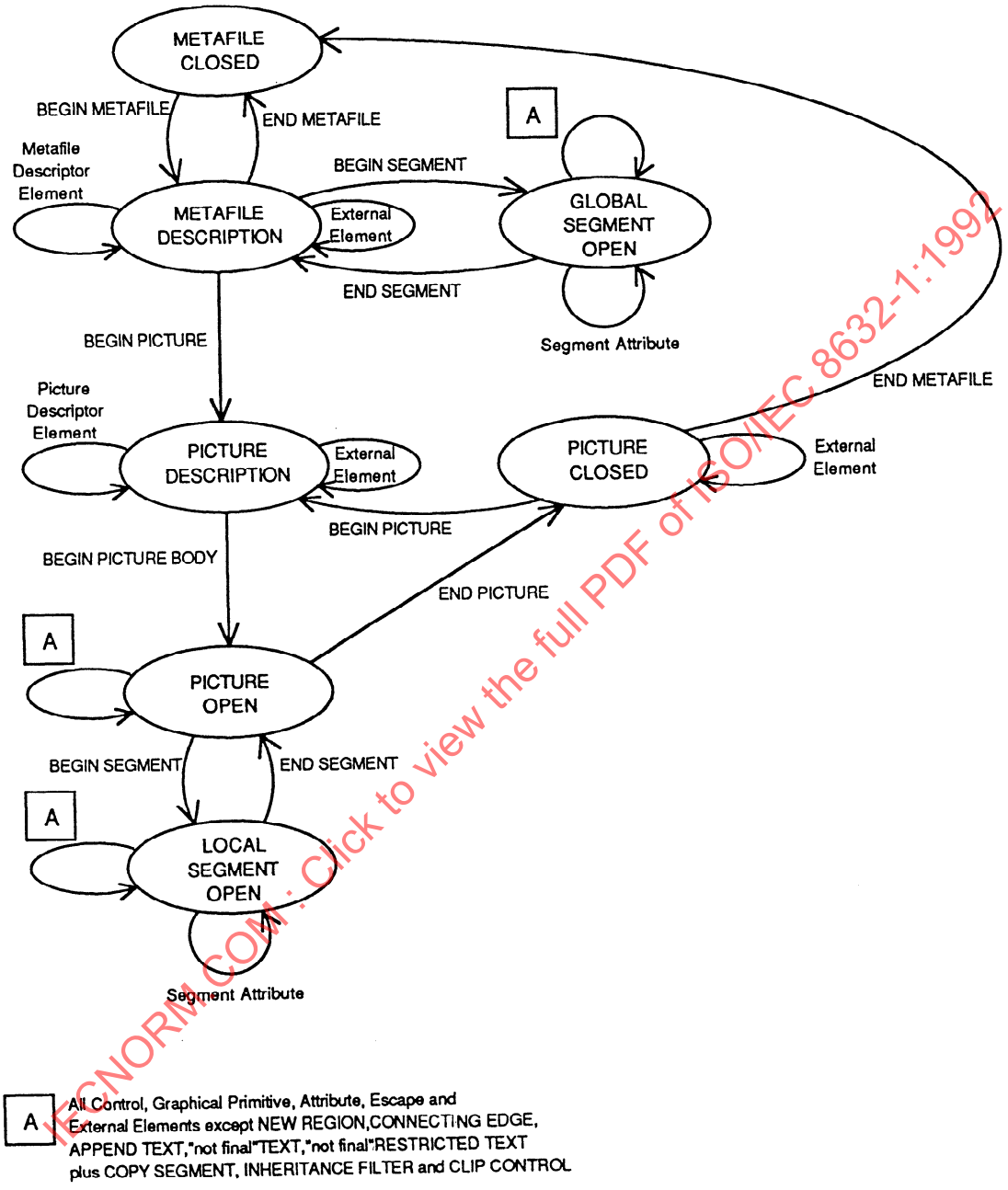


Figure 29 — State diagram for Version 2 metafiles.

NOTE 2 "External Element" in figure 29 denotes the set containing both the Escape Elements and the External Elements.

5 Abstract specification of elements

5.1 Introduction

The metafile elements are discussed in this clause.

The Delimiter Elements (see 5.2) delimit significant structures within the Metafile.

The Metafile Descriptor Elements (see 5.3) describe the functional content, default conditions, identification, and characteristics of the CGM.

The Picture Descriptor Elements (see 5.4) define the extent of VDC space and declare the parameter modes of attribute elements for the entire picture.

The Control Elements (see 5.5) specify size and precision of coordinate space and format descriptions of the CGM elements.

The Graphical Primitive Elements (see 5.6) describe geometric objects in the CGM.

The Attribute Elements (see 5.7) describe the appearance of graphical primitive elements.

The Escape Elements (see 5.8) describe device- and system-dependent elements used to construct a picture.

The External Elements (see 5.9) communicate information not directly related to the generation of a graphical image.

The Segment Elements (see 5.10) 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 8. 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 9). 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 10.

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, 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 4.3.4.5).

Combinations of simple types are 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' SS applies to such metafile aspects as line width and marker size. See table 11 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.
UI32	32-bit Unsigned Integer	An unsigned integer in the range 0..(2^{31} - 1), represented in each of the encodings with a precision equivalent to 32 binary bits.

Table 10 — Data type definitions and abbreviations (concluded)

Data Types		Meaning
BS	Bitstream	A binary data object, given an encoding-dependent representation in each of the encodings (part 2, 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 responses.

5.2 Delimiter elements

5.2.1 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:

4.2

5.2.2 END METAFILE

Parameters:

None

Description:

This is the last element of a metafile.

END METAFILE shall occur exactly once in a metafile.

References:

4.2

5.2.3 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:

4.2

5.2.4 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:

4.2

D.4.1

5.2.5 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:

4.2
D.4.1

5.2.6 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:

4.2
4.10.3

5.2.7 END SEGMENT

Parameters:

None

Description:

This is the last element of a segment. Subsequent elements will no longer belong to a segment.

References:

4.2
4.10.3

5.2.8 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:

4.2
4.6.11

5.2.9 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:

4.2
4.6.11

5.2.10 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.

References:
4.2
4.5.2
4.5.4

5.2.11 END PROTECTION REGION

Parameters:

None

Description:
This is the last element of a protection region definition.

References:
4.2
4.5.2
4.5.4

5.2.12 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:
4.2
4.6.1.2

5.2.13 END COMPOUND LINE

Parameters:

None

Description:
This is the last element of a compound line definition.

References:
4.2
4.6.1.2

5.2.14 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 4.

References:
4.2
4.5.5
4.7.3.2
4.7.8

5.2.15 END COMPOUND TEXT PATH

Parameters:

None

Description:
This is the last element of a compound text path definition.

References:
4.2
4.5.5
4.7.3.2
4.7.8

5.2.16 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:

4.2
4.6.5.2
D.4.5.13

5.2.17 END TILE ARRAY

Parameters:

None

Description:

This is the last element of a tile array definition.

Delimiter elements **Abstract specification of elements**

References:

4.2
4.6.5.2
D.4.5.13

5.3 Metafile descriptor elements

5.3.1 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:

4.3.1

5.3.2 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:

4.3.1

5.3.3 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:

4.3

5.3.4 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:

4.3

5.3.5 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:

4.3

5.3.6 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:

4.3

5.3.7 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:

4.3

5.3.8 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:

4.3

5.3.9 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:

4.3

5.3.10 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 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 and Character 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.

NOTES

1 Version 1 and Version 2 metafiles support only RGB.

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 $0 \le L^* \le 100.0$, $-128 \le a^* \le 127$, $-128 \le b^* \le 127$.

References:

4.7.6

5.3.11 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
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 4.3.2.

NOTE — The information carried by this element can be used by the interpreters to determine the maximum facilities necessary for interpreting the metafile.

References:

4.3.2

5.3.12 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 6 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 6, 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 and Version 3 metafiles.

References:
4.3.3

5.3.13 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. An amendment to this part of this International Standard, Rules for Profiles, currently at the stage of Draft Amendment, contains specifications for effective utilization of the FONT LIST element.

References:
4.3.4
4.7.3.2

5.3.14 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),
designation sequence tail (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 *designation sequence tail* parameter. The type specifies which type of character set is being declared (that is, which type of ISO 2022 designating escape sequence is associated with that character set). The 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:

4.3.4
4.7.3.2

5.3.15 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:

4.7.3.2

5.3.16 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:

4.3

5.3.17 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:

4.3
4.4.4

5.3.18 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:

4.3
4.10.4

5.3.19 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:

4.7.6

5.3.20 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 *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 *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.

NOTES

3 ISO/IEC 8613 mandates value 9 (and only value 9) for legal calibration selection value.

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:
4.7.6

5.3.21 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: font 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. This depends upon what kind of change is made to ISO/IEC 9541.

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 *Courier*.

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 *class*, the typeface *subclass*, and the typeface *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:
4.3.4.1

5.3.22 GLYPH MAPPING

Parameters:

character set index (IX)
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 5.3.14). 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 5.3.14).

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 *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 *octets 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.

NOTES

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.

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:
4.3.4.4

5.3.23 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.

NOTES

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.

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:
4.6.12

5.4 Picture descriptor elements

5.4.1 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:

4.4.1

5.4.2 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 and Version 3 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:

4.4.2
4.7.6

5.4.3 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 4 for a description of the meanings of the four styles. The modes 'fractional' and 'mm' are only supported in Version 3 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 and Version 3 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:

4.4.3
4.7.5

5.4.4 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 4 for a description of the meanings of the four styles. The modes 'fractional' and 'mm' are only supported in Version 3 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 and Version 3 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:

4.4.3
4.7.5

5.4.5 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 4 for a description of the meanings of the four styles. The modes 'fractional' and 'mm' are only supported in Version 3 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 and Version 3 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:

4.4.3
4.7.5

5.4.6 VDC EXTENT

Parameters:

first corner (P)
second corner (P)

Description:

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 4.4.4 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.

NOTE — 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:
4.4.4
4.4.5

5.4.7 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.

References:
4.4.6

5.4.8 DEVICE VIEWPORT

Parameters:

first corner (VP)
second corner (VP)

Description:
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.

NOTE — 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:
4.4.7

5.4.9 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 5.4.8). 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:
4.4.7

5.4.10 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:

4.4.7

5.4.11 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:

4.4.8

5.4.12 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:

4.4.8

5.4.13 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:
4.4.8

5.4.14 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 metafiles.

References:
4.4.8

5.4.15 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" parmeter 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:

4.4.8

5.4.16 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 4 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 may appear only in Version 3 metafiles.

References:

4.6.4.7
4.7.5

5.4.17 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 I_1 units, gap for I_2 units, solid for I_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:

4.4.9
4.7.1.2
4.7.4.4

5.4.18 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 4. 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:

4.7.4.3

5.4.19 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.

Picture descriptor elements **Abstract specification of elements**

References:
4.7.4.3
D.4.3.2

5.5 Control elements

5.5.1 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:

4.5.1

5.5.2 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:

4.5.1

5.5.3 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

5.5.4 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

5.5.5 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:

4.5.2
D.4.4

5.5.6 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:

4.5.2

5.5.7 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:

4.5.2

4.6.1.6
D.4.4

5.5.8 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:
4.5.2
4.6.2.3
D.4.4

5.5.9 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:
4.5.2
4.6.4.5
D.4.4

5.5.10 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:

4.6.11

5.5.11 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 may be saved by SAVE PRIMITIVE CONTEXT and restored by RESTORE PRIMITIVE CONTEXT are:

LINE BUNDLE INDEX
LINE TYPE
LINE WIDTH (note 1)
LINE COLOUR (note 1)
LINE CLIPPING MODE
LINE CAP
LINE JOIN
LINE TYPE CONTINUATION
LINE TYPE INITIAL OFFSET
MARKER BUNDLE INDEX
MARKER TYPE
MARKER SIZE (note 1)
MARKER COLOUR (note 1)
MARKER CLIPPING MODE
TEXT BUNDLE INDEX
TEXT FONT INDEX
TEXT PRECISION
CHARACTER EXPANSION FACTOR
CHARACTER SPACING
TEXT COLOUR (note 1)
CHARACTER HEIGHT
CHARACTER ORIENTATION
TEXT PATH
TEXT ALIGNMENT
CHARACTER SET INDEX
ALTERNATE CHARACTER SET INDEX
INTERIOR STYLE
FILL COLOUR (note 1)
HATCH INDEX
PATTERN INDEX
INTERPOLATED INTERIOR
EDGE BUNDLE INDEX
EDGE TYPE
EDGE WIDTH (note 1)
EDGE COLOUR (note 1)
EDGE VISIBILITY
EDGE CLIPPING MODE
EDGE CAP
EDGE JOIN
EDGE TYPE CONTINUATION
EDGE TYPE INITIAL OFFSET
FILL REFERENCE POINT (note 2)
PATTERN SIZE
PICK IDENTIFIER
SYMBOL LIBRARY INDEX
SYMBOL COLOUR
SYMBOL SIZE
SYMBOL ORIENTATION
CLIP INDICATOR
CLIP RECTANGLE (note 2)
PROTECTION REGION INDICATOR (note 3)
AUXILIARY COLOUR (note 1)

TEXT SCORE TYPE	TRANSPARENCY
RESTRICTED TEXT TYPE	MITRE LIMIT
FILL BUNDLE INDEX	ASPECT SOURCE FLAGS

NOTES

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 4.5.3).

2 The VDC precisions in effect when these values are saved is also recorded.

3 The currently defined regions are saved/restored as well.

References:
4.5.3

5.5.12 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:
4.5.3

5.5.13 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:
4.5.4

5.5.14 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:

4.5.5

5.5.15 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 4 for a description of the effect of mitre limit.

Valid values of the *mitre limit* parameter are non-negative reals.

References:

4.5.6

5.5.16 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 4. See clause 4 also for additional restrictions on transparent cells capability defined by this element.

References:

4.5.7

5.6 Graphical primitive elements

5.6.1 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:

4.6
4.6.1
4.7.1

5.6.2 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:

4.6
4.6.1
4.7.1

5.6.3 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 and Version 3 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 4.5.2 and 4.6.2.3.

References:

4.6
4.6.2
4.7.2

5.6.4 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 4.3.4.3 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:

4.3.4.3
4.6
4.6.3
4.7.3

5.6.5 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 4.3.4.3 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 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 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 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:

4.3.4.3
4.6
4.6.3
4.7.3
D.4.5.2

5.6.6 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

See 4.3.4.3 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:

4.3.4.3
4.6
4.6.3
4.7.3
D.4.5.1

5.6.7 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 4.6.4.4.

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:

4.6
4.6.4
4.7.4

5.6.8 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 4.6.4.4) 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:

4.6.4
4.7.4
D.4.5

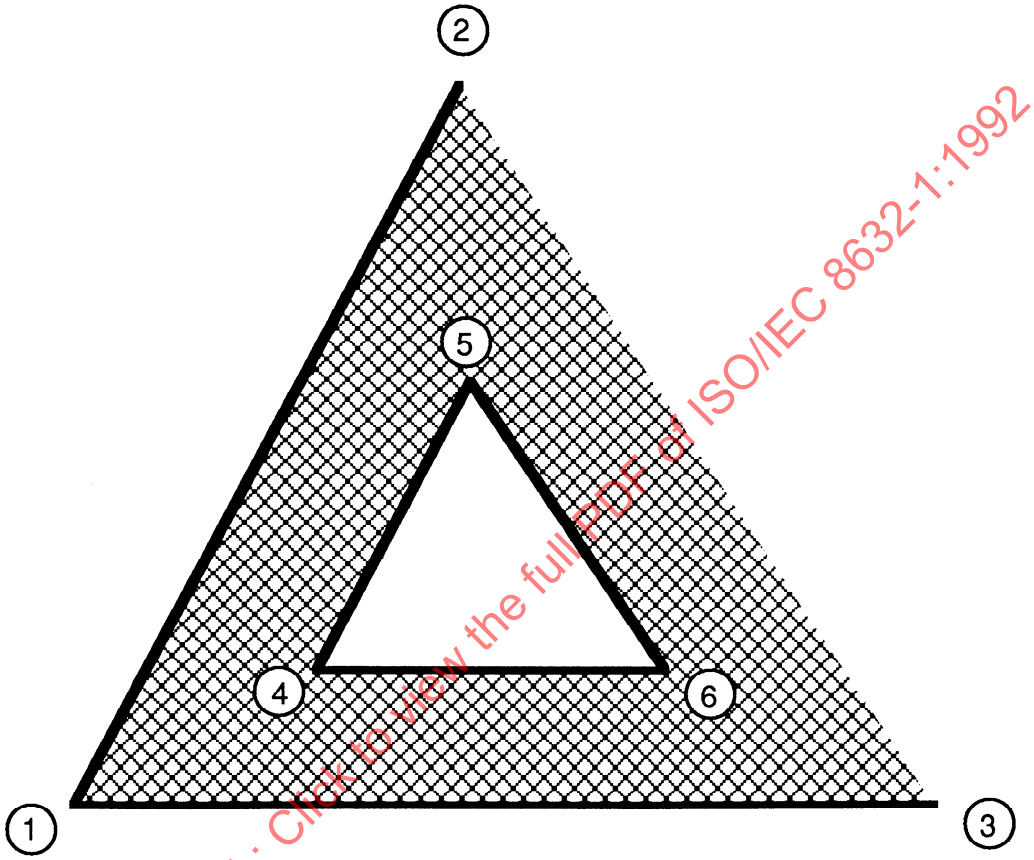


Figure 30 — Example for POLYGON SET

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

5.6.9 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·ny*CO)

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·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·ny* identical cells. The colour list consists of *nx·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 *nx*th 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:

4.6
4.6.5.1
D.4.5

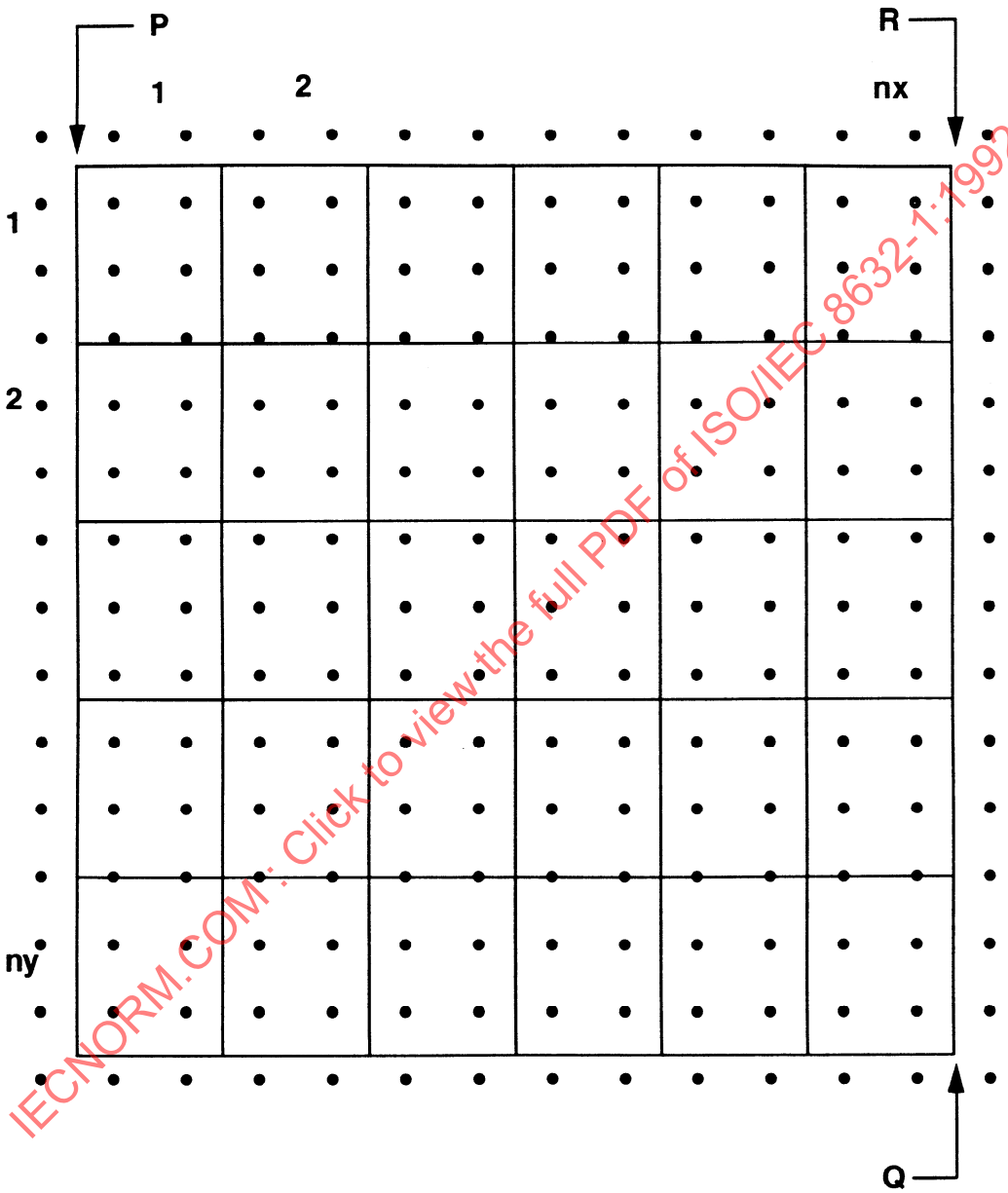


Figure 31 — Nx-by-ny CELL ARRAY rectangle mapped onto display surface

5.6.10 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* paramter 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 *ref* 471.16 regarding the format of the *data record* parameter.

NOTES

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.

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:

4.6
ref 471.16

5.6.11 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 4.6.4.4) 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:

4.6
4.6.4

4.7.4

5.6.12 CIRCLE

Parameters:

centrepoint (P)
radius (VDC)

Description:

The *radius* and *centrepoint* parameters specify respectively the radius and center of a circle, to be displayed with interior (see 4.6.4.4) 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:

4.6
4.6.4
4.7.4

5.6.13 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:

4.6
4.6.1
4.6.6
4.7.1
D.4.5

5.6.14 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 4.6.4.4) 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:

4.6
4.6.4
4.6.6
4.7.4
D.4.5

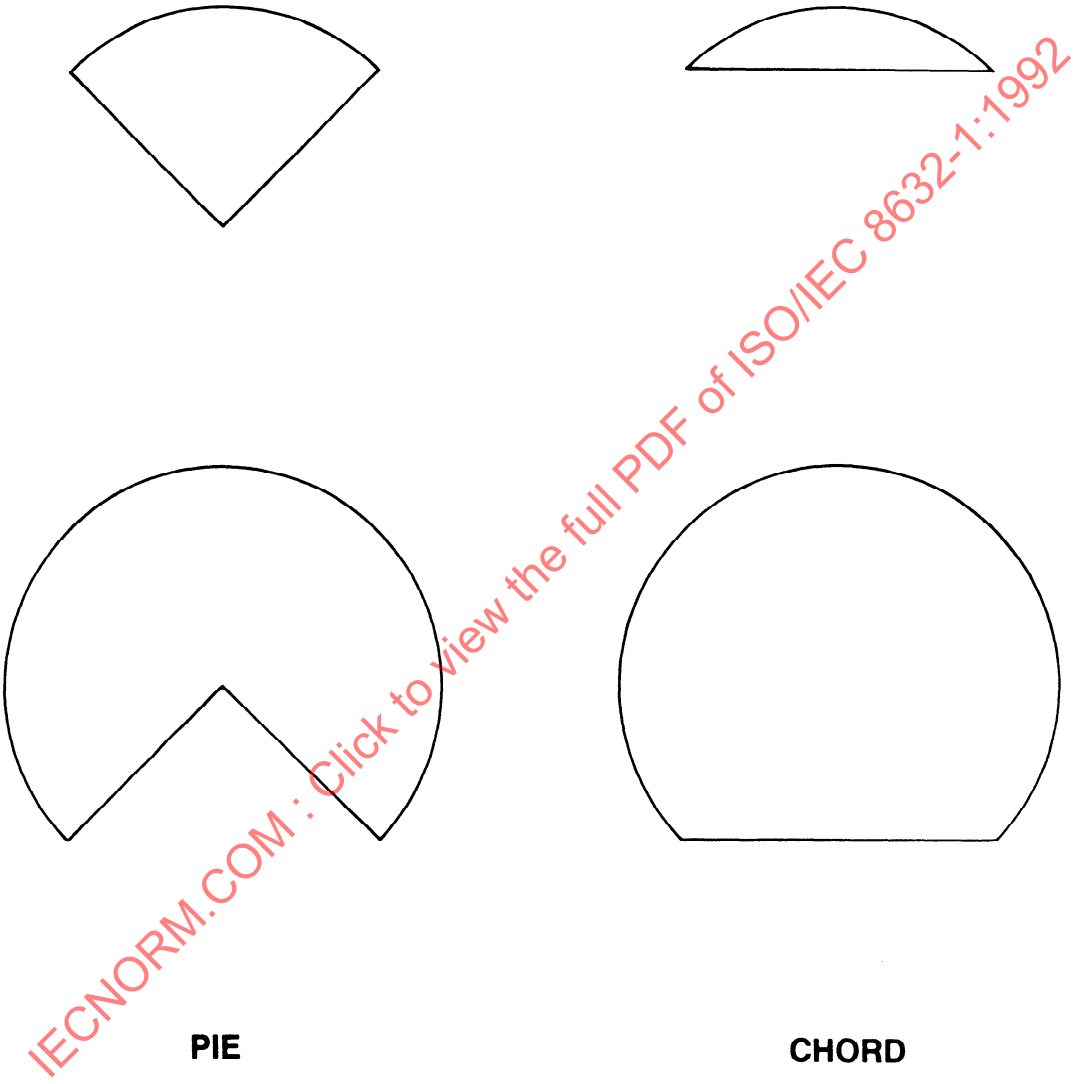


Figure 32 — CIRCULAR ARC 3 POINT CLOSE specifications with 'pie' and 'chord'.

5.6.15 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:

4.6
4.6.1
4.6.6
4.7.1
D.4.5

5.6.16 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 arc and 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 4.6.4.4) 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:

4.6
4.6.4
4.6.6
4.7.4
D.4.5

5.6.17 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 4.6.4.4) 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:

4.6

4.6.4
4.6.7
4.7.4
D.4.5

5.6.18 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:

4.6
4.6.1
4.6.7
4.7.1
D.4.5

5.6.19 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 4.6.4.4) 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:

4.6
4.6.4
4.6.7

4.7.4
D.4.5

5.6.20 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:

4.6
4.6.1
4.6.6
4.7.1
D.4.5

5.6.21 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:
4.6.11

5.6.22 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 endpoint* 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 4 for further discussion of the geometric significance of the parameterization and details of rendering of hyperbolic arcs.

References:
4.6
4.6.1
4.6.8
4.7.1

5.6.23 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 4 for

further discussion of the geometric significance of the parametcrization and details of rendering of parabolic arcs.

References:

4.6
4.6.1
4.6.9
4.7.1

5.6.24 NON-UNIFORM B-SPLINE

Parameters:

spline order (I)
number of control points (I)
control points (nP)
list of knots ((m+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 4). The *number of control points* parameter value shall bc 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 bc 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:

4.6
4.6.10
4.6.10.1

5.6.25 NON-UNIFORM RATIONAL B-SPLINE

Parameters:

spline order (I)
number of control points (I)
control points (nP)
list of knots ((m+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 4). 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:

4.6
4.6.10
4.6.10.1

5.6.26 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:

4.6
4.6.10
4.6.10.2

5.6.27 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:

4.6
4.6.12

5.6.28 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' 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)

NOTES

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.

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 Sub-

committee for *Computer Graphics*, the compression method value will be assigned by the Registration Authority.

References:

4.6
4.6.5
4.6.5.2
D.4.6

5.6.29 TILE

Parameters:

compression type (IX)
row padding indicator (I)
cell colour precision (I)
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 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*. If the value of the *cell colour precision* parameter is *m*, then 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 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)

NOTES

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.

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.

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 Sub-committee for *Computer Graphics*, the compression method value will be assigned by the Registration Authority.

References:

4.6
4.6.5
4.6.5.2
D.4.6

5.7 Attribute elements

5.7.1 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 4.6 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:

4.6.1
4.7.1.1
D.4.6

5.7.2 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 4.6 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.

NOTES

1 Line type continuity is addressed in Version 3 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.

2 Whether or not line type is maintained continuously across the segments of DISJOINT POLYLINE is not addressed by ISO/IEC 8632.

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:

4.6.1
4.7.1.1
D.4.6

5.7.3 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 4.6 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:

4.4.3
4.6.1
4.7.1.1
4.7.5
D.4.6

5.7.4 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 4.6 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:

4.6.1
4.7.1.1
4.7.6
D.3.2

5.7.5 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:

4.6.2
4.7.2.1
D.4.6

5.7.6 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 Sub-committee for *Computer Graphics*, the marker type value will be assigned by the Registration Authority.

References:

4.6.2
4.7.2.1
D.4.6

5.7.7 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:

4.4.3
4.6.2
4.7.2.1

4.7.5
D.4.6

5.7.8 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:

4.6.2
4.7.2.1
4.7.6
D.3.2

5.7.9 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 4.6 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:

4.6.3
4.7.3
D.4.6

5.7.10 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 4.6 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.

NOTES

1 Metafile generators should ensure that the selected character set and text font are compatible.

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:

4.6.3
4.7.3
D.4.6

5.7.11 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 4.6 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:
4.6.3
4.7.3
D.4.6

5.7.12 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 4.6 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:
4.6.3
4.7.3
D.4.6

5.7.13 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 4.6 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:

4.6.3
4.7.3
D.4.6

5.7.14 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 4.6 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:

4.6.3
4.7.3
4.7.6
D.3.2

5.7.15 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 4.6 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:

4.6.3
4.7.3.2
D.4.6

5.7.16 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 4.6 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 4.7.3.2.

References:

4.6.3
4.7.3.2
D.4.6

5.7.17 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 4.6 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:

4.6.3
4.7.3.2
D.4.6

5.7.18 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 4.6 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:

4.6.3
4.7.3.2
D.4.6

5.7.19 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 4.6 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:

4.6.3
4.7.3.2
D.4.6

5.7.20 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:

4.6.3
4.7.3.2

5.7.21 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 4.6 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:

4.6.4
4.7.4.1
4.7.4.3
D.4.6

5.7.22 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 4.6 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. (Scc 4.7.4.3 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:

4.6.4
4.7.4.1
4.7.4.3
D.4.6

5.7.23 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 4.6 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:

4.6.4
4.7.4.1
4.7.4.3
4.7.6
D.3.2

5.7.24 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 4.6 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 Sub-committee for *Computer Graphics*, the hatch index value will be assigned by the Registration Authority.

References:

4.6.4
4.7.4.1
4.7.4.3
D.4.6

5.7.25 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 4.6 for a list of filled-area elements. The pattern index is a pointer into the pattern tables.

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:

4.6.4
4.7.4.1
4.7.4.3

5.7.26 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 4.6 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:

4.6.4
4.7.4.1
4.7.4.3
D.4.6

5.7.27 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 4.6 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.

NOTES Edge type continuity is addressed in Version 3 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. 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 Sub-committee for *Computer Graphics*, the edge type value will be assigned by the Registration Authority.

References:

4.6.4
4.7.4.1
4.7.4.3
D.4.6

5.7.28 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 4.6 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:

4.4.3
4.6.4
4.7.4.1
4.7.5
4.7.4.3
D.4.6

5.7.29 EDGE COLOUR

Parameters:

edge colour specifier (CO)

Description:

The edge colour index or edge colour value is set as specified by the parameter(s).

When the EDGE COLOUR ASF is 'individual' and EDGE VISIBILITY is 'on', the edge of filled-area elements are displayed with this colour. See 4.6 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:

4.6.4
4.7.4.1
4.7.6
4.7.4.3
D.3.2

5.7.30 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 4.7.4.3 for the distinction between the edge and the boundary of a filled-area element. See 4.6 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:

4.6.4
4.7.4.3

5.7.31 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 that have the same hatch index and which abut, have a visually continuous interior rendering across all of the

filled areas.

References:

4.6.4
4.7.4.3

5.7.32 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:

4.6.4
4.7.6
4.7.4.3

5.7.33 PATTERN SIZE

Parameters:

pattern size specifier (4SS)

Description:

The pattern size is set to the values specified by the parameters.

When the INTERIOR STYLE is set to 'pattern' or 'geometric pattern,' subsequent filled-area elements are displayed using this pattern size. See 4.6 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 direction of the height vector. In this way, each of the *nx·ny* colour array elements is associated with one of the *nx·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 *nx*th 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 deefines the interior style for the filled-area element being displayed.

References:

4.6.4
4.7.4.3
D.4.6

5.7.34 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.

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:

4.7.6
D.3.2

5.7.35 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:

4.7

5.7.36 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:

4.7.7

5.7.37 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:

4.7.1.2

5.7.38 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:

4.7.1.2

5.7.39 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".

NOTES The styles 'restart' and 'adaptive' are likely to yield poor results if the drawn lines consist of many short segments. 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:

4.7.1.2

5.7.40 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:

4.7.1.2

5.7.41 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:

4.7.3.2

5.7.42 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 4.

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:

4.6.3.2
4.7.3.2

5.7.43 INTERPOLATED INTERIOR

Parameters:

style (IX)
reference geometry (2nSS)
number of stages (I)
list of stage designators (mR)
list of reference colour specifiers ((m+1)CO)

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 4.

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:

4.7.4.3

5.7.44 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 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:

4.7.4.4

5.7.45 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:

4.7.4.4

5.7.46 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'.

NOTES The styles 'restart' and 'adaptive continue' are likely to yield poor results if the drawn lines consist of many short segments. 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:

4.7.4.4

5.7.47 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 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:

4.7.4.4

5.7.48 SYMBOL LIBRARY INDEX

Parameters:

symbol library index (IX)

Description:

The value of the *symbol library index* parameter selects 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:

4.6.12

5.7.49 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:

4.6.12
D.4.6

5.7.50 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 clause 4 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:

4.6.12

5.7.51 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 4.6 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:

4.6.12

5.8 Escape elements

5.8.1 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.

NOTES ESCAPE is designed for access to non-standardized control features of graphics devices, as opposed to non-standardized geometric primitives. The GENERALIZED DRAWING PRIMITIVE element is designed for specification of non-standardized primitives. 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:

4.8
D.4.7

5.9 External elements

5.9.1 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:

4.9

5.9.2 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.

NOTE — 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:

4.9
D.4.8

5.10 Segment elements

5.10.1 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 4.10.5. The way in which clipping is applied to primitives within a copied segment is controlled by CLIP INHERITANCE (see 4.10.5).

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:

4.10.1
4.10.5

5.10.2 INHERITANCE FILTER

Parameters:

filter selection list (list of elements or groups from:

LINE BUNDLE INDEX	ALL

Abstract specification of elements	Segment elements
LINE TYPE	LINE TYPE ASF
LINE WIDTH	LINE WIDTH ASF
LINE COLOUR	LINE COLOUR ASF
LINE CLIPPING MODE	MARKER TYPE ASF
MARKER BUNDLE INDEX	MARKER SIZE ASF
MARKER TYPE	MARKER COLOUR ASF
MARKER SIZE	TEXT FONT INDEX ASF
MARKER COLOUR	TEXT PRECISION ASF
MARKER CLIPPING MODE	CHARACTER EXPANSION FACTOR ASF
TEXT BUNDLE INDEX	CHARACTER SPACING ASF
TEXT FONT INDEX	TEXT COLOUR ASF
TEXT PRECISION	INTERIOR STYLE ASF
CHARACTER EXPANSION FACTOR	FILL COLOUR ASF
CHARACTER SPACING	HATCH INDEX ASF
TEXT COLOUR	PATTERN INDEX ASF
CHARACTER HEIGHT	EDGE TYPE ASF
CHARACTER ORIENTATION	EDGE WIDTH ASF
TEXT PATH	EDGE COLOUR ASF
TEXT ALIGNMENT	LINE ASFS
FILL BUNDLE INDEX	MARKER ASFS
INTERIOR STYLE	TEXT ASFS
FILL COLOUR	FILL ASFS
HATCH INDEX	EDGE ASFS
PATTERN INDEX	ALL ASFS
EDGE BUNDLE INDEX	MITRE LIMIT
EDGE TYPE	LINE CAP
EDGE WIDTH	LINE JOIN
EDGE COLOUR	LINE TYPE CONTINUATION
EDGE VISIBILITY	LINE TYPE INITIAL OFFSET
EDGE CLIPPING MODE	TEXT SCORE TYPE
FILL REFERENCE POINT	RESTRICTED TEXT TYPE
PATTERN SIZE	INTERPOLATED INTERIOR
AUXILIARY COLOUR	EDGE CAP
TRANSPARENCY	EDGE JOIN
LINE ATTRIBUTES	EDGE TYPE CONTINUATION
MARKER ATTRIBUTES	EDGE TYPE INITIAL OFFSET
TEXT REPRESENTATION AND PLACEMENT ATTRIBUTES	
TEXT PLACEMENT AND ORIENTATION ATTRIBUTES	
FILL ATTRIBUTES	
EDGE ATTRIBUTES	SYMBOL LIBRARY INDEX
PATTERN ATTRIBUTES	SYMBOL COLOUR
OUTPUT CONTROL	SYMBOL SIZE
PICK IDENTIFIER	SYMBOL ORIENTATION
ALL ATTRIBUTES AND CONTROL	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:

4.10.5

5.10.3 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 4.10.5).

The description of clipping in this subclause also applies to the protection region used for clipping and shielding of arbitrary areas.

References:

4.10.5

5.10.4 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.

References:

4.10.4.2

5.10.5 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.

References:

4.10.4.3

5.10.6 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.

References:

4.10.4.4

5.10.7 SEGMENT PICK PRIORITY

Parameters:

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.

Segment elements **Abstract specification of elements**

References:
4.10.4.5

6 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 5.3.12 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:	0.0,1.0,0.0,1.0
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:	1; default VDC extent
CLIP RECTANGLE:	VDC EXTENT
CLIP INDICATOR:	on
LINE CLIPPING MODE:	locus
MARKER CLIPPING MODE:	locus
EDGE CLIPPING MODE:	locus
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

Metafile defaults

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)

Metafile defaults

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 — 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

7 Conformance

7.1 Forms of conformance

ISO/IEC 8632 specifies functionality and encodings of Computer Graphics Metafiles. A metafile may conform to ISO/IEC 8632 in one of two ways. Full conformance occurs when the metafile is functionally conforming and additionally conforms to one of the encodings specified in ISO/IEC 8632-2, ISO/IEC 8632-3, or ISO/IEC 8632-4. Functional conformance occurs when the content of the metafile corresponds exactly to the Functional Specification given in this part of ISO/IEC 8632, but a private encoding is used. These rules are expanded in the following sub-clauses.

NOTE 1 Work is in progress on an amendment to this International Standard, "Rules for Profiles", which will comprise a new clause 8. Conformance of metafile generators and interpreters is defined in this new clause as well.

7.2 Functional conformance of metafiles

A metafile is functionally conforming to ISO/IEC 8632 if it is functionally conforming to one of the versions defined in ISO/IEC 8632-1. The defined versions of ISO/IEC 8632-1 are Version 1, Version 2, and Version 3. A metafile is functionally conforming to a version of ISO/IEC 8632 if the following conditions are met

a) The metafile contains exactly one METAFILE VERSION element whose value defines the conformance version.

b) All graphical elements contained therein match the functionality of the corresponding elements of ISO/IEC 8632-1 for that version. The metafile shall conform to the relationships defined in the formal grammar for that version, the state tables, and all other syntactic and semantic requirements for that version.

c) The sequence of elements in the metafile conforms to 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, include exactly one metafile descriptor at the beginning which contains at least all the required elements.

d) 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 element or the external elements APPLICATION DATA and MESSAGE.

7.3 Full conformance of metafiles

A metafile is said to be fully conforming to ISO/IEC 8632 if the following conditions are met.

a) The metafile is functionally conforming, as specified above.

b) The metafile is encoded in conformance with one of the standardized encodings specified in ISO/IEC 8632-2, ISO/IEC 8632-3, or ISO/IEC 8632-4.

7.4 Conformance of other encodings

A functionally conforming metafile may use a private encoding. While it is beyond the scope of ISO/IEC 8632 to standardize rules for private encodings, annex E suggests minimum criteria that private encodings should meet.

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>
```

Detailed grammar **Formal grammar of the functional specification of version 1 metafiles**

```
<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>
                                       <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>*
```

Detailed grammar **Formal grammar of the functional specification of version 1 metafiles**

<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>

Formal grammar of the functional specification of version 1 metafiles **Detailed grammar**

```
<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>

Terminal symbols **Formal grammar of the functional specification of version 1 metafiles**

<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>

<picture content> ::= <picture element>
|

<picture element> ::= <control element>
| <graphical element>
| <closed figure>
| <primitive attribute element>
| <pattern table element>
| <colour table element>
| <specification element>
| <segment control element>
| <extra element>

 ::= <BEGIN SEGMENT>
<segment identifier>
<segment attribute element>*
<eligible picture element>*
<END SEGMENT>

<segment identifier> ::= <name>

<eligible picture element> ::= <control element>
| <graphical element>
| <closed figure>
| <primitive attribute element>
| <specification element>
| <segment control element>
| <extra element>

B.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>
                                      <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>

<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>
                                |   <MAXIMUM VDC EXTENT>
                                      <point> (2)
                                |   <SEGMENT PRIORITY EXTENT>
                                      <minimum extent>
                                      <maximum extent>
                                |   <segment>
                                |   <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>

<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>
```

Detailed grammar **Formal Grammar of the functional specification of version 2 metafiles**

```
<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>
                                  | <SEGMENT>

<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>
```


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.