15

Standardisation in computer graphics: an introduction to GKS

K. W. Brodlie
Department of Computer Studies, University of Leeds

15.1 Introduction

One of the major developments in computer graphics in the 1980s has been the emergence of the Graphical Kernel System, or GKS, as an International Standard—ISO 7942 (ISO 1985).

Why was GKS developed? To answer this question, it is necessary to return to the mid-1970s. At that time there was a large number of basic graphics packages—for example, GINO-F and GHOST were prominent in the UK, while other packages were more popular in other countries. This growth in basic graphics packages was a hindrance, not a help, to computer graphics, because it prevented the development of portable graphical applications software. Suppose someone wanted to develop some contouring software. He would write this in terms of his favourite local package—say GHOST. But should he subsequently wish to transfer this software to another site, then it would only run unchanged at another site which supported GHOST; if another basic package was supported, then a certain amount of editing, possibly even redesign, would be required.

So it was decided in the mid-1970s that computer graphics as a discipline had reached a mature state, and that its development was being hampered by lack of standards. Accordingly national activities were set in motion to create proposals for a graphics standard. Two major offerings were forthcoming: the CORE system from the ACM SIGGRAPH group in the US, and GKS from a DIN graphics working party in West Germany. An ISO working group was formed in 1978 to work towards a single international standard. GKS was chosen, mainly on the basis of its relative compactness as compared with the CORE. There began an intensive technical review of GKS, in which the US, British, Dutch and French standards bodies played major rôles as well as the Germans. A Draft ISO standard was agreed in 1982, and the fully-fledged ISO standard followed in 1985.

It has already made a significant impact. Several good implementations of GKS are now available, and the two leading computer manufacturers—IBM and DEC—both offer a GKS implementation on their principal ranges of computers. Thus it is now possible to recommend to users of computer graphics in application areas such as archaeology to write their graphical programs in terms of GKS—in the confidence that such programs can be transported to colleagues anywhere in the world and have a realistic chance of finding a GKS implementation on which to run.

The idea of this paper is to introduce archaeologists to the main concepts of GKS, and to encourage them to find out more about GKS so they can use it in their computing work.

15.2 GKS

GKS is a functional specification of a kernel system for computer graphics. Key words in that sentence are 'functional' and 'kernel'. By 'functional', it is meant that GKS describes a set of functions for graphics programming—functions that are independent of any particular programming language. There is a separate standard which will specify the mapping of the GKS functions to a number of common programming languages: the FORTRAN binding, as it is called, is almost settled (ISO 1986a). In this way, the same concepts will apply in all graphics work, independently of the programming language being used.

By 'kernel', it is meant that GKS just provides those functions necessary for the drawing of pictures: any functions which can be built from simpler functions are therefore not included. For example, GKS includes a function for drawing lines (POLYLINE), but not one for drawing axes, or contour maps, because these can be drawn from more elementary functions. GKS is concerned only with 2D graphics, but an extension to 3D, called GKS-3D, will soon appear as a standard (ISO 1986b).

The fundamental principles of GKS can be described quite simply. A user creates a picture using a set of building blocks, known as output primitives. There are six output primitives in GKS: polyline, polymarker, text, fill area, cell array and GDP—these are described in more detail in the next section. These primitives are defined in world coordinates—a coordinate system chosen by the user to be appropriate to the problem. For example, a geographer might choose to define his picture in National Grid coordinates. The appearance of the output is controlled by a number of attribute functions, such as character height.

Output in GKS is displayed on workstations, which is the GKS term for a graphical device. The process of transforming from world coordinates to device coordinates is done in two steps. In the first step, the world coordinates are normalised: the user defines a window, which is a rectangular region in world coordinates containing the picture to be displayed; the user also defines a corresponding viewport, which is a rectangle within a unit square. Points within the window are mapped to corresponding points in the viewport. The user may define several window-viewport mappings, and so put together a composite picture in the unit square. By default, this unit square is mapped automatically to the largest square area on the workstation display surface.

GKS has a number of additional facilities. Primitives can be grouped together in segments; segments can be highlighted, deleted or transformed—allowing a form of animation on suitable devices. There are a number of graphical input functions, such as locator which allows coordinate positions to be identified on the display surface and returned to the user program. There is a facility for storing graphical data in a file—called a graphical metafile—for long term storage, or transfer to another site.

15.3 GKS output facilities

As mentioned above, pictures in GKS are composed from six output primitives. These primitives are as follows:

POLYLINE Draws a connected sequence of lines through a given set of points.

POLYMARKER Draws a marker symbol at a given set of points.

TEXT Draws a string of characters at a given position.

FILL AREA Draws a polygonal area defined by a given set of points.

CELL ARRAY Draws an image made up of a grid of coloured rectangles.

GDP Draws a generalised drawing primitive—an implementation-dependent facility allowing access to special hardware features, such as circle drawing.

The actual appearance of the output primitives is controlled by a number of attributes. Polyline, polymarker, text and fill area each have their own set of attributes. A novel concept in GKS is the idea of bundled attributes. Consider, for example, line drawing. GKS recognises three aspects of lines: linetype, linewidth and line colour. But it is realised that different devices have different capabilities for displaying lines—for example, a monochrome device such as a Tektronix 4010 can use different linetypes, but not different line colours, while on colour raster displays, colour and linetype can be used, but possibly not linewidth. A user may prefer therefore not to describe different lines by these aspects explicitly, but leave it to GKS to choose the best representation on a particular workstation. Thus GKS allows a user to associate a polyline index with each polyline: this acts as an index into a bundle table (one for each workstation) which holds different representations of lines suitable for the workstation concerned. Similarly, the appearance of the polymarker and fill area primitives is controlled by assigning a polymarker or fill area index. Text also has an index, but in addition there are a number of other text attributes—controlling for example the character height and orientation. These other attributes are essentially geometric, and properly part of the description of the picture, rather than the representation of the picture on a particular workstation.

15.4 GKS transformations

The output primitives describe a picture in world coordinates. GKS provides a number of transformation functions which allow the world coordinate picture to be displayed on the surface of any particular workstation. This is done in a two-stage process, using a normalised space as an intermediary: this normalised space is the unit square, and its coordinates are referred to as normalised device coordinates, or NDC for short.

The user defines a window, which is a rectangle in world coordinates containing the picture; and he also defines a corresponding viewport, which is a rectangle in NDC onto which the window is to be mapped. This is known as a normalisation transformation. A GKS program can use several normalisation transformations, mapping different windows onto different viewports and building up a composite picture in NDC.

By default, the whole of NDC space is mapped to the largest square area on a workstation's display surface. But, if required, a workstation transformation can be defined to give user control over this mapping, on a workstation-by-workstation basis.

15.5 GKS input facilities

Input devices in GKS are divided into six classes, according to the type of information they return. These classes are

LOCATOR Returns a coordinate position; typically this will be implemented in practice by a cursor which is moved by the operator to a required position and then triggered. The point located in this way is transformed back to world coordinates and its position returned to the user program.

STROKE	Returns a sequence of coordinate positions.
VALUATOR	Returns a single real value—typically this will be implemented by displaying a scale on the display screen, with an arrow indicating the current value. The operator moves the arrow up and down the scale using a mouse, say.
CHOICE	Returns a single integer value.
STRING	Returns a string of characters input by the operator.
PICK	Described in section 15.6 on segments.

Notice that the input devices are classified, not by their physical type such as cursor, keyboard, etc., but by the form of data they return. For this reason, they are often referred to as logical input devices.

Input devices can be used in three different ways, but most GKS implementations only support one mode, known as REQUEST mode. There are six functions: REQUEST LOCATOR, REQUEST STROKE, etc. To execute REQUEST LOCATOR, a GKS program halts, and waits until the operator has triggered the locator input before continuing—a good analogy is a FORTRAN READ statement where a program waits for data to be entered, and continues when the 'carriage return' trigger is hit.

15.6 Segments

It is sometimes useful to be able to group a number of output primitives together as a single unit, so that some operation may be performed on the unit as a whole—for example, in a picture composed of various archaeological exhibits, each exhibit might be stored as a single entity, or 'segment' in GKS terminology. Each segment can have a number of attributes: for example, a segment can be highlighted, or not; a segment can be visible, or invisible. A particularly useful attribute is a segment's transformation matrix: this is by default the identity matrix, but can be changed to allow the scale, position and orientation of a segment to change dynamically.

Another important attribute is segment detectability. It is often useful to select from the picture on a graphics display a particular segment of output—in the example above, to select a particular exhibit. GKS has a class of input device known as PICK: this is typically implemented as a cursor which the operator uses to identify a particular segment, by placing the cursor over some primitive belonging to that segment. In a cluttered picture, the operator can be helped by making only relevant segments detectable; a REQUEST PICK operation will return the name of a segment which is detectable, and 'closest' to the cursor position.

15.7 Graphical metafiles

Another aspect of computer graphics which has been addressed by the standards bodies is the storage of pictorial information. GKS has a special type of workstation known as metafile output: primitives sent to this workstation are stored on a file on disc. There is a corresponding metafile input workstation which allows metafiles to be read back into GKS and their contents displayed. Note however that GKS does not prescribe the format of metafiles; this is considered in a separate standard known as the Computer Graphics Metafile (ISO 1986c), which is close to being ratified.

15.8 Conclusions

This paper has given only a brief introduction to GKS. There are two excellent books on GKS which are recommended as further reading. The first is *Introduction to the Graphical Kernel System (GKS)* (Hopgood *et al.* 1986), which gives an easily read overview of GKS; the other is *Computer Graphics Programming: Graphical Kernel System* (Enderle *et al.* 1986), which is better used as a reference work.

A number of implementations of GKS are already available. Of particular interest to the academic community is GKS-UK, an implementation written jointly by the SERC at the Rutherford Appleton Laboratory and by ICL. This is being distributed to UK University Computer Centres by a GKS Support Team, based at the Universities of Edinburgh, Leicester and Salford. The implementation is also being marketed commercially, and further details are available from the author.

It is important to emphasise that GKS is a kernel system. Some people are disappointed to find that GKS does not have the facilities they are accustomed to in the package they are currently using, such as axes-drawing for example. For this reason, those developing graphical software in various application areas such as archaeology may be deterred from using GKS as their base system. It becomes vital that there is a widely available library of graphical routines based on GKS and containing routines for axes drawing, curve and function drawing, contouring and other common, inter-disciplinary operations. This already exists in a limited way in the NAG Graphical Supplement, which can be used in conjunction with a number of different basic graphics packages including GKS. It is possible that a revised version of the Supplement, based purely on GKS, will be developed by NAG.

References

ENDERLE, G., K. KANSY, & G. PFAFF 1986. *Computer Graphics Programming: GKS—The Graphics Standard,* Springer-Verlag, Berlin.

HOPGOOD, F. R. A., D. A. DUCE, J. R. GALLOP, & D. C. SUTCLIFFE 1986. *Introduction to the Graphical Kernel System (GKS),* Academic Press, London.

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

ISO 1986a. *Information processing systems—Computer graphics—Graphical Kernel System (GKS) language bindings—Part 1: FORTRAN,* ISO DIS 8651/1—1986, ISO.

ISO 1986b. *Information processing systems—Computer graphics—Graphical Kernel System for three dimensions (GKS-3D) functional description,* ISO DP 8805. 2, ISO.

ISO 1986c. *Information processing systems—Computer graphics—Metafile for the storage and transfer of picture description information.,* ISO DIS 8632, ISO.