Graphics standards in High Energy Physics

O. Couet CERN, Geneva, Switzerland

Abstract

This article presents the various graphics standards used in High energy physics software, in which context they are used, and an idea of what will be the next generation of graphics packages in view of LHC experiments.

Keywords: Graphics standards; GKS; PHIGS; OpenGL; PostScript.

1 Why graphics standards ?

In the past, in High Energy Physics (HEP), graphics standards (such as GKS) were mainly used to develop portable graphics programs which are independent of the graphics hardware and operating system. In future HEP should use appropriate industry graphics standards, if they can be provided at a reasonable cost, because the industry effort is much greater than HEP effort for software developments, HEP does not have special requirements in many computing areas, and finally the manpower for central support is limited (even diminishing), at least for CERN. For all these reasons HEP should in the future concentrate the effort on specific HEP software.

2 The various graphics layers

A graphics package can be divided into several layers.

Low level graphics: This level is the closest to the hardware. It allows the drawing of points, lines, polygons and text but only in pixel coordinates on screen or on paper (printer). This layer is machine dependent and is often called a "Device driver".

Window management: This level is not really a "graphics" one because it doesn't produce any picture. Nevertheless it is mandatory in almost all the graphics applications and is often bundled with the low level graphics layer. This level essentially provides graphics window manipulation (move, resize, request the size, etc.). X-Windows is the de-facto standard on UNIX platforms.

Basic graphics: This layer performs graphics in user defined coordinates which are independent of the final hardware output required. It gives also the possibility to create graphics objects such as GKS segments, PHIGS structures, etc. which can be manipulated interactively. This level is machine independent.

High level graphics: This layer provide functions able to produce sophisticated graphics primitives (macro primitives) such as axes, iso-contours, vector fields, etc. It should provide all the graphics tools specific to a given environment. For example, for HEP applications, this layer should provide all the necessary functions to produce log scales, error bars, lego plots, etc.

3 Where do we need graphics in HEP ?

In HEP all levels of graphics are used, ranging from the low level 2D drawing to the high level 3D graphics. The following list describes the various areas where graphics is used in HEP.

- *Data analysis and presentation:* The main tool used for Physics analysis and data presentation is PAW. In this kind of system 2D and 3D views are used depending on the kind of data set represented. A high quality paper output is required for publications.
- *Simulation:* GEANT is the system used to simulate the experiments at the physics level. GEANT uses 2D and 3D graphics. 3D graphics was mainly wireframe but some high level

rendering algorithms including “ray tracing” or “Gouraud shading” are used more and more.

- *Event display:* Physics experiments typically have at least one “Event Display” program. These kind of programs give reconstructed views of the interactions in the detector. 2D and 3D graphics are used. 3D is mainly wireframe. It is evolving and some graphics rendering algorithms are now used. Speed is needed to rotate the 3D volumes in real time.
- *Data acquisition Monitoring, Control:* This is often called *online software*. It requires fast 2D graphics and deported output. There is no real need for high quality paper output.
- *CAD:* 2D and 3D views are used. It requires high quality paper output, often in large formats such as A0.
- *Virtual reality:* Virtual reality is used to do navigation into the accelerators and detectors. It requires fast high level 3D rendering. These are real time programs.

4 Who is using what ?

For each kind of system listed in the previous section we shall now describe which graphics system is used.

- *Data analysis and presentation:* As we said before, this task is mainly covered with PAW. The graphics is provided by the HIGZ library. The X11 driver of the library covers almost all the need of the screen output. The paper output need is completely covered by the PostScript driver.
- *Simulation:* The simulation is done with GEANT-3. GEANT-3 uses HIGZ 2D. The 3D transformations are done by GEANT itself. Because of the large number of volumes manipulated and the fact that GEANT needs to have the knowledge of these volumes (for tracking), GEANT does not use the graphics structures provided by systems such as PHIGS.
- *Event display:* This is the area where the PHIGS system is most widely used because it provides a standard way to create and manipulate 3D graphics structures.
- *Data acquisition Monitoring, Control:* In this domain, X11 is often used directly. Also HIGZ and HPLOT are used (with the X11 driver). PHIGS or GKS are little used.
- *CAD:* The graphics package used comes with the CAD system itself. Often the user does not even know which one is used.
- *Virtual reality:* Vitual reality programs run on dedicated hardware (mainly SGI). The graphics system used are special VR systems built on top of GL or OpenGL.

5 Yesterday in HEP (before 1992)

The first graphics (ISO) standard used in HEP was the Graphical Kernel System (GKS). Below are listed the main advantages (+) and the disadvantages (-) of this system:

+ It is an (ISO) standard which provides portability and independence from the hardware.
+ It produced true PostScript output.
+ It was simple to use.
+ A lot of device drivers were available (Tektronix, Falco, Megatek, etc.).

But:
- It was only 2D graphics. GKS 3D has immediately been superseded by PHIGS.
- The X11 device driver was not well supported (at least in the GTSGRAL version).
- No scientific macro primitives were available.

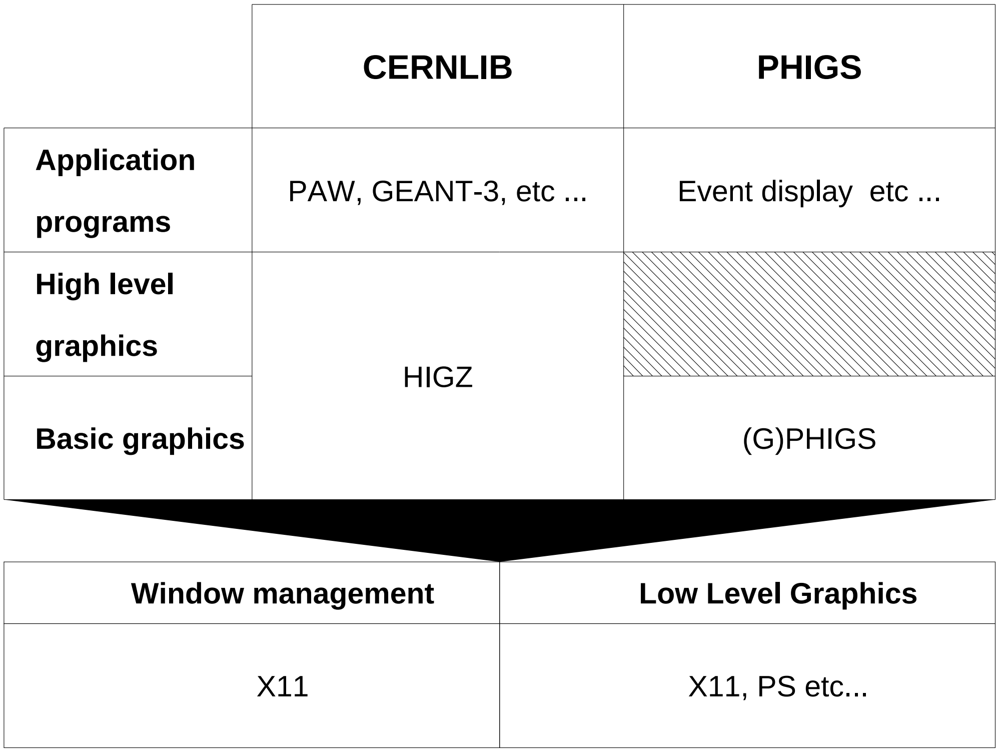
No new application uses GKS and it is now virtually abandoned.

6 Today in HEP

Today in HEP, two main lines are used to produce graphics:

1. the “CERNLIB” one, using the HIGZ package developed at CERN.
2. the PHIGS environment.

The CERNLIB and PHIGS environment are shown in figure 1.



	CERNLIB	PHIGS
Application programs	PAW, GEANT-3, etc ...	Event display etc ...
High level graphics	HIGZ	▨
Basic graphics		(G)PHIGS

Window management	Low Level Graphics
X11	X11, PS etc...

Figure 1: The CERNLIB and PHIGS graphics environments.

GPHIGS is the PHIGS implementation of G5G/TGS. This is the main machine-independent implementation available today (since 1990). Note that not all the HEP experiments use GPHIGS (CMS also uses HP native PHIGS).

The following list described the main advantages (+) and the disadvantages (-) of those two lines:

HIGZ: + HIGZ is tuned for HEP. It provides the basic functions needed for physics data presentation: macro primitives, axes, log scales, errors bars, etc.
+ It produces true PS output. We will see later that some systems are only able to produce bitmap Postscript output which is not good for publication.
+ It is simple to use.
+ The X11 driver is well supported.
But:
- It is not a 3D package (2D 1/2).
- Only a few drivers are supported.
- It is not an industry standard. This is likely to give problems for long term support.

PHIGS: + PHIGS is an ISO standard.
+ Graphics structures are supported
+ It provides many drivers.
+ It provides true PS output.
+ It is fully 3D.

+ The X11 driver is supported.

But:

- It is complex to use (especially PHIGS+).

- It does not provide scientific macro primitives (although G5G has developed PHIGURE).

7 What are the industry standards today ?

Nowadays graphics developments are driven more and more by industries such as movies, games, design, etc., in which the quality and the realism of the pictures produced is essential. OpenGL and Open Inventor meet these requirements.

- *OpenGL* is a software interface for applications to generate interactive 2D and 3D graphics.
- *Open Inventor* is an object oriented 3D toolkit which simplifies graphics programming. It includes a rich set of objects such as cubes, polygons, text, materials, cameras lights etc . Open Inventor is built on top of OpenGL. It defines a standard file format for 3D data interchange.

Open Inventor/OpenGL are the emerging industry standards for screen output:

- Microsoft provides OpenGL for free on Windows.
- A public domain OpenGL has been built by Gnu (YGL).
- 35 vendors have already adopted OpenGL (IBM, HP, SunSoft, Microsoft, DEC, etc.).
- In HEP, the BaBar experiment, DESY, and Fermilab have already acquired licenses for Open Inventor (and Iris Explorer).

8 Tomorrow in HEP ?

The Open-Inventor/OpenGL environment is shown in figure 2. Note that OpenGL does not provide window management facilities, as do X11, SGI/GL, Apollo/GPR, etc. The native window manager has to be used to manipulate windows (move, resize, request the size, etc.). The OpenGL task is “only” to produce graphics.

The following list described the main advantages (+) and the disadvantages (-) of Open Inventor / OpenGL:

+ It Works on PCs. This is the first system widely available on both workstations and PCs. This is a great asset because the PC will be an increasingly important platform for HEP computing in the future.

+ It is the emerging de facto industry standard.

+ OpenGL manages the screen output only (the native window manager must also be used).

+ OpenGL is bundled with NT.

+ VRML: Virtual Reality modeling Language is very close to the Open Inventor file format.

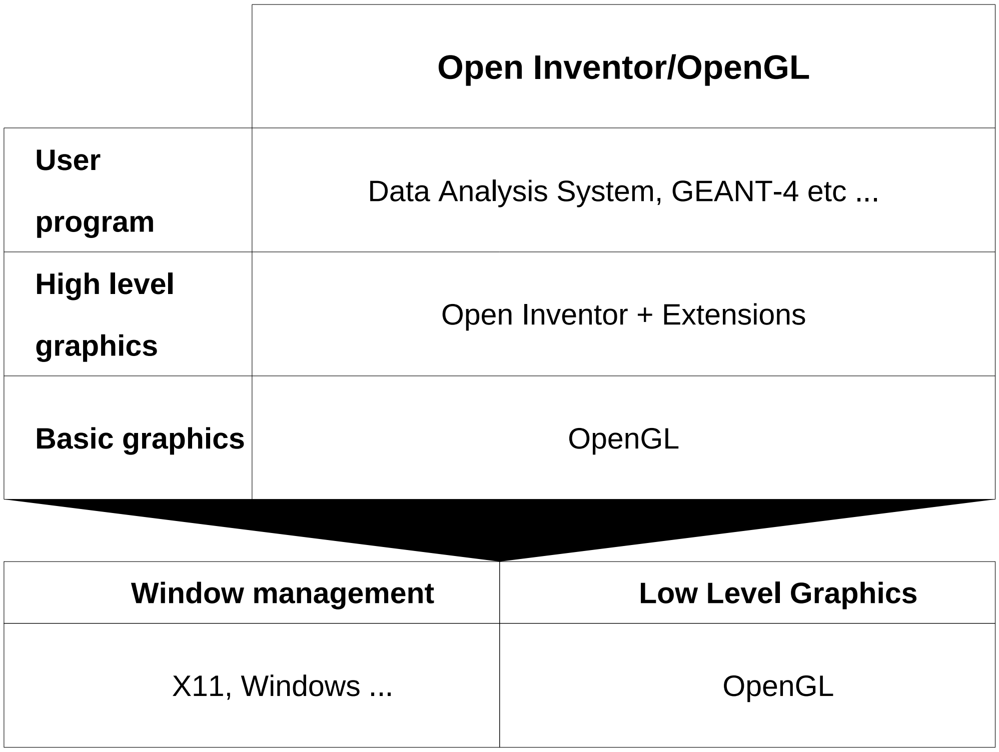
But:

- No true PostScript output is provided in Open Inventor (G5G/TGS has developed one).

- No scientific macro primitives are provided (but likely to come: G5G/TGS extensions).

9 Conclusions

- After a long period, during which several standards for screen ouput and printing were available (Tektronix, Megatek, etc.), the *Low level graphics* has now standardized on **X-Windows** for the screen output and **Postscript** for printing. These two “standards” are in fact “de-facto standards” and have not been initiated by any committee.
- The *Basic Graphics* and the *High level graphics* do not have standards yet. GKS is now abandoned and PHIGS will certainly be replaced by **OpenGL**.



	Open Inventor/OpenGL
User program	Data Analysis System, GEANT-4 etc ...
High level graphics	Open Inventor + Extensions
Basic graphics	OpenGL

Window management	**Low Level Graphics**
X11, Windows ...	OpenGL

Figure 2: The Open-Inventor/OpenGL graphics environments.

- OpenGL is the first “standard” which covers the *Low level graphics* and the *Basic Graphics*.
- **OpenInventor**, with the G5G/TGS extensions, should cover the *High level graphics*.
- Graphics is now more and more integrated with the Web (VRML, Java). New products such as COSMO 3D from SGI are now integrating 3D graphics capabilities with the Web. The SGI web pages say COSMO 3D is: “ a powerful set of advanced technologies bringing interactive multimedia and 3D graphics to the World Wide Web.”