EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN-DD/89/17

May 11, 1989

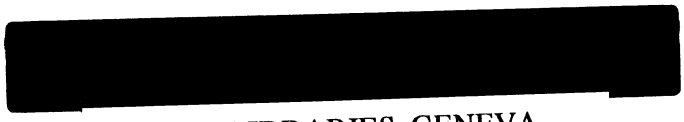
PAW, a general-purpose portable software tool for data analysis and presentation.

René Brun, Olivier Couet, Carlo E. Vandoni, Pietro Zanarini

Data Handling Division
CERN
1211 Geneva 23 SWITZERLAND

Abstract

During the last twenty years, CERN has played a leading role as the focus for development of packages and software libraries to solve problems related to High Energy Physics (HEP). The results of the integration of resources from many different Laboratories can be expressed in several million lines of code written at CERN during this period of time, used at CERN and distributed to collaborating Laboratories. Nowadays, this role of software developer is considered very important by the entire HEP community. In this paper a large software package, where man-machine interaction and graphics play a key role (PAW-Physics Analysis Workstation), is described. PAW is essentially an interactive system which includes many different software tools, strongly oriented towards data analysis and data presentation. Some of these tools have been available in different forms and with different human interfaces for several years.



CERN LIBRARIES, GENEVA

CM-P00059918

Presented at the conference on Computing in High-Energy Physics
Oxford, April 1989

1. Introduction

Data analysis packages have been available to the HEP community for more than 25 years. Data presentation packages have been also available since a long time. In the past, the two functions were separated, as in any case the media available for presenting data in visual form were rather limited (for instance, for many years the only output medium normally available to produce a scatter plot was the line printer). However, it was recognized very early on that the provision of graphics facilities could be an important factor in helping the physicist in the analysis of experimental data and in the production of the results in graphical form. Development in the area of computer graphics started very early; operational systems were already available to the physics community already in the late 1960s, [1]. Nowadays, the availability of modern workstations, offering an adequate computing power and high quality graphics capabilities at an affordable cost, have had a major impact in the field of data analysis and presentation packages.

In this paper a large software package where man-machine interaction and graphics play a key role (PAW[1]) is described, and its origins, motivations and main characteristics are presented.

2. PAW

PAW is conceived as a general-purpose instrument to assist physicists in the analysis and presentation of their data. A number of special-purpose systems of this kind have been developed in the past. It is worth mentioning here the MIDAS system [2] implemented at ESO, GEP[3], IDA[4], PV-WAVE[5], POL [6], and PUNCH [7] by RAL. PAW provides interactive statistical or mathematical analysis working on objects familiar to physicists like histograms, event files (Ntuples), vectors etc., coupled with powerful and sophisticated interactive graphical presentation facilities, allowing the production of graphical output of publication quality. PAW is primarily intended to be the last link in the analysis chain of experimental data. Some parts of PAW are also useful for graphical representations of simple data provided by the user, for the study of mathematical functions, and it may also be used as a data presentation package in a data-acquisition environment. It has, therefore, access facilities to experimental data, to HBOOK [8] data, or to any FORTRAN data, in addition to those generated by PAW itself, in numerical or graphical form. Powerful facilities for mathematical manipulation or combination of data objects (i.e. for data analysis), are also provided. As concerns data presentation, a large set of graphic functions is provided, easing the production by the user of graphs of high quality. This graphical part of PAW is essentially based on the features provided by HPLOT [9], coupled with the facilities provided by the new HIGZ [10] package.

2.1 Data analysis

The main objects onto which PAW is capable of operating can be categorized as in the following.

- Files: It is of vital importance for a package of this kind to have access to experimental data, no matter where the data reside. Therefore, facilities to retrieve data and to access them on the various machines available to the user are obviously part of the package. Files are, by their nature, non-volatile, and reside normally on disk areas.

- Histograms: Data resulting from HEP experiments are often presented in the form of histograms. The facility for producing this particular kind of graph is so important in HEP that the original semantics of the word "histogram" has been altered. In HEP jargon "histogram" means the complex of numeric and non-numeric information, enabling the data presentation package to build-up a histogram in graphical form. Histograms are volatile items, but they can

---

1 **Paw:** the foot of an animal having claws | to stroke with the hand | to feel or touch clumsily, indelicately, or rudely, etc.., (Webster's Third New International Dictionary of the English Language, Merriam Webster, 1981).

In this context: PAW – Physics Analysis Workstation.

be stored and retrieved by appropriate commands. Needless to say, all the functionality available under the well–known packages HBOOK and HPLOT is now available under PAW. In particular, this includes facilities for the creation of one and two–dimensional histograms, operations between histograms, projections and comparisons of histograms, etc.

- N–tuples: Ntuples can be considered as large named two–dimensional arrays. The Ntuples can be seen from the physics point of view as event files. The user can access the ntuple as a whole or single columns, or even single components. Columns can be identified by a name or by an index. A rather complete set of operators is available to deal with ntuples – these include capabilities to apply "cuts" or selection criteria to the ntuple data, using a notation where arithmetic and boolean operators and mathematical functions can be freely used. A powerful facility exists ('mask'), enabling the user to select an ntuple subset which has particular characteristics, allowing in this way very fast access to the data subset.

- Vectors: Vectors are in fact one to three–dimensional arrays of volatile nature. They can be created during a session, operated upon using the full functionality of SIGMA, used to produce graphics output, and, if necessary, stored away on disk files for further usage.

- Pictures: These are collection of graphical primitives or macroprimitives. They can be used to produce off–line hard copies but, more important, can be manipulated by the graphic editor available under HIGZ.

2.2 Data presentation

In the past the preparation of graphs of publication quality has always been considered a tedious job and very often this was delegated to draftsmen. The graphics facilities offered by PAW, linked with the availablity of top quality hard copiers (like modern laser printer/plotters), permit even the inexperienced user to produce graphs ready for publication. Examples are given in the following sections.

3. The components: HBOOK, HPLOT, SIGMA, COMIS, MINUIT, ZEBRA, KUIP, HIGZ

As mentioned above, PAW integrates most of the functionality of older packages such as HBOOK[8], HPLOT[9], SIGMA[11], COMIS[12], MINUIT[13]. Several years of experience and hundreds of thousands of hours spent by users with these packages proved that the approach was a sound one. While integrating these packages, we managed to add functionality, to provide a full integration of the features offered, to permit full interchangeability of data and objects as well as full access to data bases.

A detailed description of PAW appears in [14]. Figure 1 outlines the various components of PAW and their mutual relationships. It is important to underline here the fact that these modules, although integrated in PAW, can be used autonomously or integrated in other packages. A typical example is the use of KUIP, ZEBRA[15] and HIGZ in GEANT, [16]. There is no space in this paper to describe even superficially all the components of PAW. They are completely documented in the manuals quoted in the references. A general description of the package is also given in [17], while a brief history of its development and goals is given in [18].

The two modules KUIP and HIGZ deserve a more detailed description.

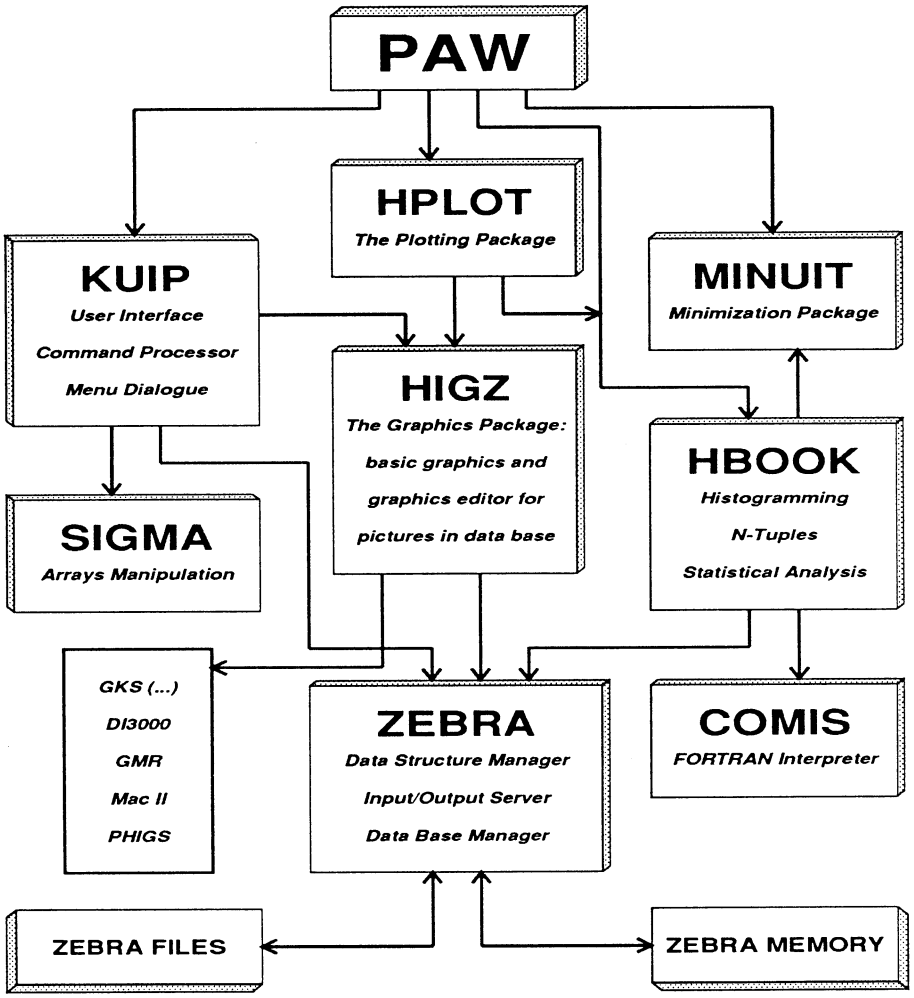


Figure 1: The PAW structure.

3.1 KUIP

KUIP (Kit for a User Interface Package) [19] represents a new approach to the relatively old problem of general–purpose User Interface systems. The basis of KUIP is the so–called Command Definition File (CDF) which constitutes a very concise formal description of the command structure. An appropriate module, the KUIP Compiler, generates from this file a set of FORTRAN subroutines which are then compiled and linked with the interactive application.

The dialogue between the user and the system is handled either by typing command lines or by choosing alphanumeric or graphical menus. The different dialogue modes available in KUIP are illustrated in Figure 2.
In order to avoid verbosity in typing command lines, often useless and annoying for the experienced user, command abbreviations are possible as long as they do not produce ambiguities. The user is

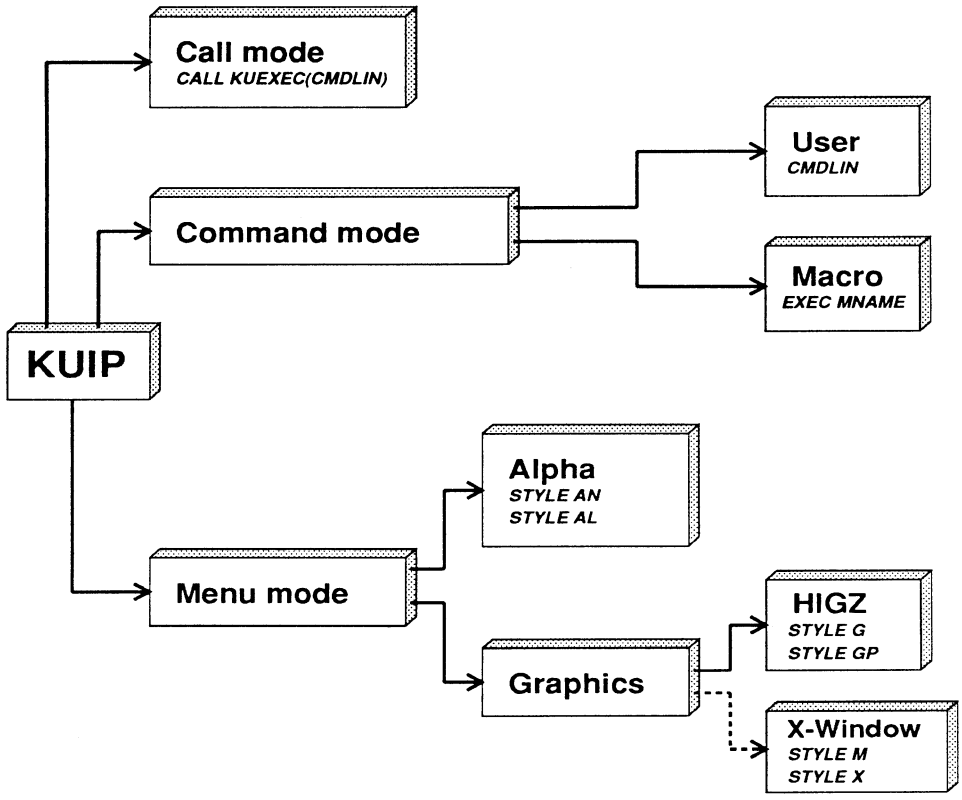


Figure 2: The KUIP dialogue modes.

able to switch from one dialogue mode to the other at any moment. Furthermore, menus do not need any special additional programming; they are automatically derived from the command structure as described in the CDF. Needless to say, it is possible to group commands in 'macro' files and invoke these sets of commands for repeated execution. Assignment and control statements may be used inside macros. KUIP keeps track of all command lines entered (independently of the dialogue mode), which are automatically recorded in a macro file, and may therefore be edited and re–executed as with normal macros. An Unix–like history mechanism is also available.

An on–line help facility is integral part of the system. This can be considered as a good step in the direction of the "system to be used without a manual". The importance of proper documentation should never be underestimated: it is vital to make available users' documentation since the release of the first prototype of the package (documentation which could possibly be produced automatically by the system itself). It is also very important to provide different levels of documentation: beginners' guide, users' guide, reference manual, each one having a different target and different users. One of the major problems normally encountered in this area is how to keep the documentation up–to–date. KUIP solved this problem by introducing a feature in the system which permits the automatic generation of the documentation (already in text processing format). As an additional benefit, as the documentation is an integral part of the language description system, the updating of the documentation

is automatic. To our knowledge these two features (automatic production of documentation and automatic update) are features unique to PAW.

The system has to be considered completely open–ended, in the sense that the user is able not only to personalize his own version, for instance using the mechanism of the KUIP macros, but he is also able to create his own interactive language adapted to his particular application, by using the Command Definition facility.

3.2 HIGZ

The use of graphics packages like GKS is becoming increasingly important for the provision of a standard interface between user programs and devices. The use of only one such package, GKS, provides portability of application programs between systems on which GKS is installed, and makes the application programs largely device–independent. These packages, however, have limitations. They do not provide the high level functions (axes, graphs, logarithmic scales, etc.) necessary for a data presentation system. There are always (sometimes minor) differences in the actual implementations on different computers (the so called "installation–dependent–parameters"). They do not foresee an acceptable way of recording large volumes of graphical information in compact form with a convenient access method for later manipulation. The GKS metafile is conceived as a vehicle to communicate series of pictures between computers but not for their later manipulation.

The package HIGZ (High level Interface to Graphics and ZEBRA) [10] is an interface package between the user program and an underlying graphics package. It provides an interface to a standard data structure management system (ZEBRA) and through it a mechanism to store graphics data in a way which makes their organization and subsequent editing possible and easy. The picture data base is highly condensed and fully transportable. A picture editor is part of the package, allowing merging of pictures, editing of basic graphics primitives, operations onto HIGZ structures, etc. The level of HIGZ was deliberately chosen to be close to GKS and as basic as possible. This makes the interface to GKS a very simple one, and preserves full compatibility with the most important underlying graphics packages. HIGZ does not introduce new basic graphics features and does not duplicate GKS functions. On the other hand, some graphics macro–primitives are implemented, providing very frequently used functions such as histograms, full graphs, circles, bars and pie charts, axes, etc.. HIGZ is presently interfaced to several versions of GKS, to GMR, to DI3000 and very soon to PHIGS. The underlying graphics package is chosen at compilation time. HIGZ provides both 2D and 3D graphics features. In addition, a PostScript driver and communication facilities (TELNETG)[2] are provided.

The structure of the package is shown in Figure 3.

---

[2] The TELNETG program permits the running of an application based on HIGZ on a remote computer (mainframe) with graphic input/output on a local computer (workstation), combining the power of the CPU of the mainframe and the flexible graphics facilities of the workstation.

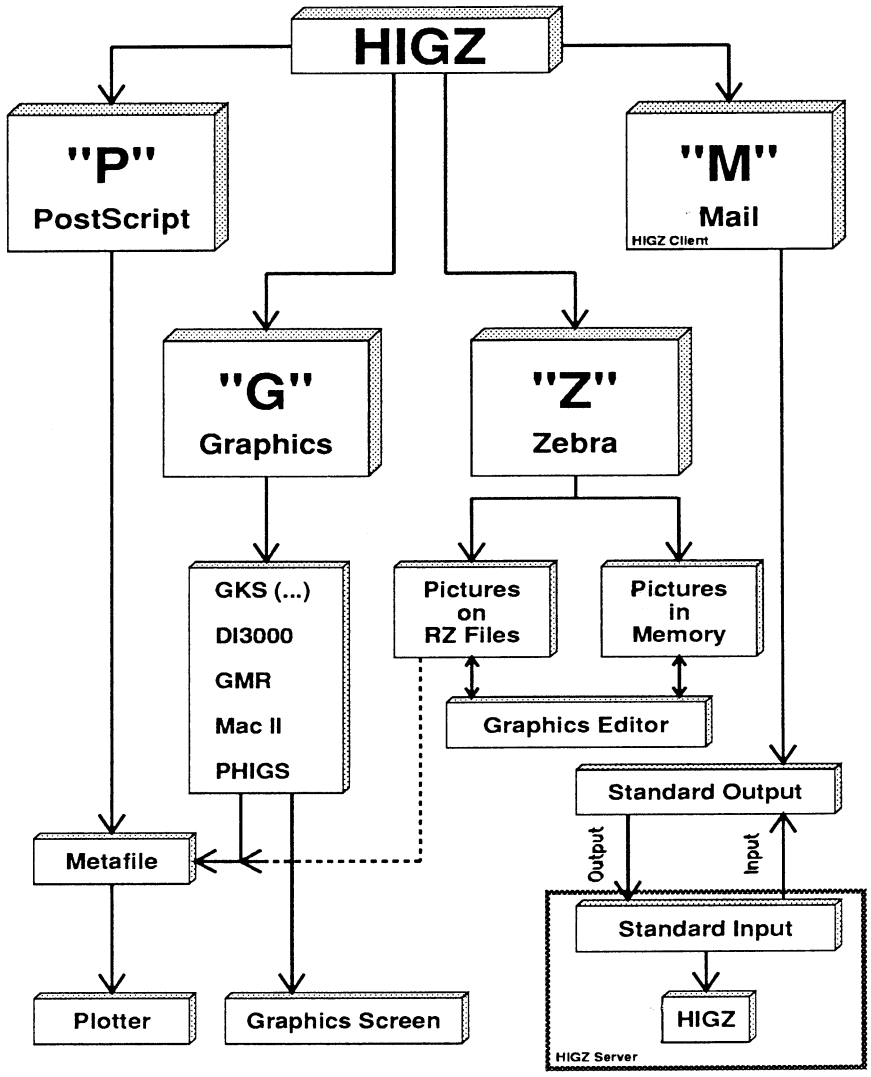


Figure 3: The HIGZ structure.

4. Distributed PAW

Another unique characteristic of PAW is its ability to run in a distributed fashion. In practice, the user sitting at his workstation is capable of accessing files on remote computers, even if the data representation is different. This is achieved thanks to the object-oriented system ZEBRA[15] and a Remote Procedure Call facility built on top of TCP/IP or equivalent transport protocol. In an homogeneous environment, a PAW user can also access data coming from a different process via a memory mapping technique (i.e. global sections on VAX/VMS). The facilities have proved to be very useful in data acquisition systems and will become vital in a world of distributed computing.

5. Users' experience and feedback

When developing this kind of system, sometimes the users' needs and wishes may not coincide with those of the system designers. On the other hand, users' feedback is invaluable during the design and implementation phases. The most effective way to obtain such a feedback is to make available to a few selected users a prototype, even incomplete, of the system, with the clearly declared purpose of permitting experimentation and of obtaining feedback. The designer must be prepared to discard completely the prototype and to start afresh the programming of the future production version of the system. This philosophy was adopted during the development of PAW and proved to be quite successful. In the same way, the designer must be prepared to revise the basic design; we want here to quote an example: the basic syntax of PAW was originally designed on a "verb/predicate" structure. During the development, we quickly reckoned that this structure was orthogonal to a good human interface when in a menu-mode. Therefore we decided to reverse the syntactical definition to a "predicate/verb" structure. Thanks to the command-definition-file facility of KUIP, this total re-definition was an extremely easy job.

6. Usage of PAW

6.1 Size of the package

In order to quantify in some way the complexity of the system we could just quote that the total number of lines of code of the seven modules composing PAW is well over 100,000. This figure does not include the underlying graphics package, the mathematical routines or the data structures management system. All the components of PAW are written in FORTRAN 77.

6.2 Where is PAW available

PAW has been installed today in many laboratories in the world. It is almost impossible to say how many users exist today but our estimation is that their number is well above 1000. At CERN a recent measurement has shown 300 users and 8000 sessions per month on the IBM system only. An equal number of users are estimated to use regularly the VAX and Apollo systems.

The packages ZEBRA, HBOOK, HPLOT, MINUIT and HIGZ are implemented on many computer systems. PAW has been implemented on the following systems: IBM (VM/CMS and MVS/TSO), VAX (VMS), Apollo (Aegis and UNIX) and SUN (UNIX).

A partial implementation also exists for the Macintosh II and a version for the CRAY is in preparation.

7. Acknowledgments

The authors wish to thank Paolo Zanella and Michael Metcalf for their continuous support and encouragement, and all the USERS of PAW for their patience and invaluable feedback! We also recognize the role of R. Bock in the original concept of PAW.

Particular thanks go to Jackie Turner, for invaluable help in the final preparation of the paper.

8. References

[1] P.M.Blackall, J.C.Lassalle, C.E. Vandoni and A.Yule, *High Energy Physics Applications in Computer Graphics-Techniques and Applications*, edited by R.D. Parslow (Plenum Press, London, 1969), pp. 149-160.

[2] European Southern Observatory, Image Processing Group, *MIDAS Users Guide*, ESO, Garching bei Munchen, January 1988.

[3] E. Bassler, *GEP User Manual*, DESY 1985.

[4] T. Burnett, *IDA Interactive Data Analysis*, SLAC MAC-III memo 1/83-6.

[5] *PV-WAVE Precision Visuals Workstation Analysis and Visualization Environment-Introduction*, Precision Visuals, Inc., June 1988.

[6] N.Armenise, G.Zito, A.Silvestri, E.Lefons,M.T.Pazienza and F.Tangora, *POL: An Interactive System to Analyze Large Data Sets*, Computer Physics Communications 16 (1979) pp. 147-157.

[7] R.G.Parry, K.J.Knowles, C.Moreton-Smith, R.T.Lawrence, W.C.A. Pulford and M.A.Sturdy, *PUNCH User Guide*, Rutherford Appleton Laboratory, RAL-88-109, December 1988.

[8] R.Brun and D.Lienart, *HBOOK users guide*, CERN program library Y250.

[9] R.Brun, N.Cremel and O.Couet, *HPLOT users guide*, CERN program library Y251.

[10] R.Bock, R.Brun, O.Couet, R.Nierhaus, N.Cremel, C.E.Vandoni and P.Zanarini, *HIGZ Users Guide*, CERN program library Q120.

[11] R. Hagedorn, J. Reinfelds, C.E. Vandoni and L. Van Hove, *SIGMA, A New Language for Interactive Array-oriented Computing*, CERN Report 73-5 (1973). also CERN Report 78-12 (1978).

[12] V.Berezhnoi, R.Brun, S.Nikitin, Y.Petrovykh and V.Sikolenko, *COMIS Users Guide*, CERN program library L210.

[13] F.James and M.Roos, *MINUIT, Function Minimization and Error Analysis*, CERN program library D506.

[14] R.Brun, O.Couet, N.Cremel, C.Vandoni and P.Zanarini, *PAW-Physics Analysis Workstation*, CERN program library Q121.

[15] R.Brun and J.Zoll, *ZEBRA Users Guide*, CERN program library Q100.

[16] R.Brun, F.Bruyant, M.Maire, A.C.McPherson and P.Zanarini, *GEANT3*, CERN Data Handling Division, DD/EE/84-1, September 1987.

[17] R.Bock, R.Brun, O.Couet, J.C.Marin, R.Nierhaus, L.Pape, N.Cremel-Somon, C.Vandoni and P.Zanarini, *PAW-Towards a Physics Analysis Workstation*, Computer Physics Communications 45 (1987).

[18] C.E. Vandoni, *Development of a large graphics-based application package*, Invited paper, Third International Yugoslav Conference on Computer Graphics, Dubrovnik, Yugoslavia, 22-24 June 1988.

[19] R.Brun and P.Zanarini, *KUIP Users Guide*, CERN program library I202.