KEK 83-10
July 1983
E

OFF-LINE SOFTWARE FOR LARGE EXPERIMENTAL SETUPS

F. BRUYANT

NATIONAL LABORATORY FOR
HIGH ENERGY PHYSICS

KEK Reports are available from

Technical Information Office
National Laboratory for High Energy Physics
Oho-machi, Tsukuba-gun,
Ibaraki-ken, 305
JAPAN

Phone: 0298-64-1171
Telex: 3652-534 (Domestic)
(0)3652-534 (International)
Cable: KEKOHO

OFF-LINE SOFTWARE FOR LARGE EXPERIMENTAL SETUPS(*)

F. Bruyant(**)

National Laboratory for High Energy Physics
Oho-machi, Tsukuba-gun, Ibaraki-ken, 305, Japan

ABSTRACT

The purpose of this report is to emphasize the importance of Off-line software for large experimental setups in High Energy Physics. Simple notions of program structuring, data structuring and software organization are discussed in the context of the software developped for the European Hybrid Spectrometer.

(*) The talk at the TRISTAN VENUS meeting

(**) On the leave from European Organization for Nuclear Research CERN/EP 1211 Geneva 23 Switzerland

KEYWORDS: OFF-LINE SOFTWARE, HIGH ENERGY PHYSICS, LARGE EXPERIMENT.

I. Introduction and Acknowlegements

Strictly speaking, the off-line software in high energy physics is the interface between the data acquisition and the experimental results.

In the period covering approximately the years 1960 to 1970 off-line software was not usually considered as a major task and was not given too much 'a priori' thought. Data reduction was one of the many activities of a physicist, not separated from data analysis. During that period however the huge development of Bubble Chamber techniques has been mostly responsible for a change of attitude in the High Energy Physics community about matters related to off-line software. The use of bubble chambers as facilities by many research institutions from different countries and the increasing complexity of the detectors (Gargamelle, Mirabelle, BEBC,..) made it clear that the tasks of off-line software were not compatible any longer with a 'Do it yourself' approach. Regarding the evolution in Europe, it is one of the merits of Prof. Ch. Peyrou (CERN, TC) during those days to have given to some members of his staff as much freedom and as much support as possible to think about new concepts for data handling and to propose tools which could assist the physicists in the design and the development of large size programs. PATCHY[1] and HYDRA[2] were conceived during that period.

In the years 1970-1980, more and more physicists (not all of them!) have become convinced of the importance of off-line software and are aware of the special care it requires. Far from considering it as an auxiliary task they rather feel it should be the heart of the experiment. However, because of budget and man power problems, very often these good feelings do not go beyond the declaration of intention. This is most unfortunate.

I am very glad that you asked me today at KEK to discuss some aspects of off-line software as it shows clearly your concern about this matter, and I hope you will succeed in the realisation of the TRISTAN projects.

I would like also to take this opportunity to express my gratitude to Prof. T. Nishikawa, Director General of KEK, for his invitation and generous support,

To Prof. I. Kita, Tokyo University of Agriculture and Technology, for his most appreciated kindness in organizing my trip to Japan and for his continuous help,

To Profs. T. Yamagata and T. Hirose, Tokyo Metropolitan University, and to the physicists and students there, for very friendly welcome and care. I am especially indebted to Drs. S. Kitamura (TMU) and T. Emura (TUATT) for guiding my first steps in the discovery of Japan.

This talk whose purpose is to emphasize the importance of off-line software, is divided in three parts:

- General considerations for off-line software,
- The EHS setup and off-line software organisation,
- Some comments about data structures.

At the end, as an introduction to the discussion that will follow, I will come back to those points which seem essential for the success of future experiments.

II. General Considerations for Off-Line Software

The approach to off-line software depends closely on the answers given to simple questions like,

what does one expect from a software team?

what does a software team expect from others?

and at first, is the concept of a software team relevant at all?

The opinions expressed here, based on past experience, reflect of course personal feelings and are of limited value. However the questions themselves are interesting and should be examined with care by every person involved in an experiment, in particular by the project leader.

The answer to the last question, as you may guess, is YES. The variety and complexity of the detectors used in high energy physics

nowadays require that many people contribute to the implementation of the software. They have to share the work and to make sure that their efforts are convergent. This is the definition of a team. For a large experimental setup, there is a need for a strong software team, with well defined and well accepted rules. Let's recall briefly the main characteristics of a team and the minimal set of rules:

- A team is not the simple addition of independent people.
- A team needs some continuity, the permanence of some of its members.
- The activities within a team have to be coordinated by one person.
- The coordinator has to encourage the delegation of responsibilities.
- No member should start a major task without 'a priori' consultations, at least with the coordinator.
- Every member is urged to bring suggestions.
- Every member should feel free to make criticisms.

Two remarks:

- These rules do not imply that all people have to work in the same place. In practice however, as long as powerful networks are not available, the absence of frequent personal contacts is a real inconvenience.
- There is obviously no intellectual hierarchy between the members of a team. However, in case of conflict, the coordinator should be given the authority to decide one way or another and has to assume his responsibility.

Coming to the first question "what does ONE expect from a software team?", the answer depends on who is ONE.

<u>ONE</u> is a physicist in charge of the construction of some hardware component:

In the early phase the constructor may need

- Help for the design specifications (acceptance, multi-track performance, integration to the setup...) at the time of the preparation of the proposal.
- Guidance for the definition of some tolerances (required survey accuracy, default of alignment...).

Later the constructor might expect

- Feed back for the evaluation of the quantitative and qualitative performance of the hardware.

ONE is a physicist interested in the analysis of the program results (refered to as physicist):

The physicist is usually asking for too much, but could reasonably expect,

- Reliability and operational simplicity of the programs.
- Good documentation.
- Ease of result interpretation.
- Completeness of the production chain.
- Correctness of the results, which means no wrong interpretation or, in case of doubt, access to the alternative solutions.
- Accuracy of the results, which means for instance the best estimate of the kinematical parameters and of their errors.

The physicist is also expecting an easy adaptation of the software to changes in the hardware or in the running conditions.

Remark: The above classification is purely artificial. In practice it is frequent that the constructor and the physicist are the same person, that the physicist and the programmer are the same person. It turns out to be usually much better if it is so.

Now, the second question "what does a software team expect from OTHERS?" The answer depends on who the OTHERS are:

OTHERS is the project leader:
The software team (at least its coordinator) expects

- A strong overall coordination with estimate of the time schedule and of the priorities.
- Then, confidence and support.

OTHERS are constructors:
The software team expects

- Information about the details of construction (including the way the survey will be done).
- Access to the drawings.
- Information about calibration and monitoring procedures (the calibration usually requires special runs, the monitoring is a quasi continuous process during the data taking).

OTHERS are those responsible for data taking:
In the early phase of the project, the software team expects

- A two-way communication with the on-line experts to prepare a clean interface, which should also contain the key for an unambiguous understanding of the data (correspondance between a logical component and the physical component used at any one time, read-out conventions..).
- The guarantee that data will be available for all required special studies.

And, during the run,

- Some control of quality of the data recorded (this being partly also an off-line responsibility).
- An accurate report of any action taken deliberately, or of the occurence of any abnormal events, which could affect the interpretation of the data (automatic log book)

OTHERS are physicists:
The software team expects from them

- As much light as possible on the physics aims (when known!).
- Some patience when the production starts, as it is very usual that things look very bad at first and improve considerably, within a few days, once the programs are tuned.

After this short review of the connections between the different poles of an experiment it might be easier to appreciate the importance of software organization.

I would like to add one more comment: A software team should impose on itself another constraint dictated by pure aesthetic considerations. Given the painful and long effort required for the development of large programs it is important, as for any human enterprise, to consider the intrinsic value of the work, the beauty of the construction. Maybe the best comparison is with the realisation of a Japanese garden. It requires, I guess, a lot of imagination, a lot of work, a lot of time. It presents both a fantastic diversity and a harmonious unity. It offers here or there a few highlights linked together through a network of paths, none of them useless, and ... it has a few unaccessible areas which might hide a confused jungle!

III The EHS Setup and Off-Line Software Organization

The European Hybrid Spectrometer (EHS) is a facility to study multihadron events in high energy interactions.

It is a two lever arm spectrometer proposed by a large european collaboration and approved by the CERN-SPSC between 1975 and 1980.

It is hybrid as it combines bubble chamber and counter techniques.

Nearly completed today (Fig. 1), the EHS consists of:

- A vertex detector. Alternatively, RCBC (a 80 cm hydrogen bubble chamber inside the vertex magnet M1) used mainly for soft hadron physics (NA23, NA22), or HOLEBC (a high resolution 10 cm hydrogen bubble chamber upstream of M1). In this second configuration the space inside M1 is filled with high accuracy devices like a mini drift chamber (MDC) and a proportional inclined chamber (PIC)
- A beam lever arm with two 5-plane proportional wire chambers
- An interaction/event trigger
- A first lever arm with a large 6-plane proportional wire chamber (W2) and three large 4-plane drift chambers (D1, D2, D3)
- A second lever arm, downstream of the auxiliary magnet M2, with three large 4-plane drift chambers (D4, D5, D6)
- A quasi-complete set of particle identifiers. In the first lever arm, in addition to the bubble chamber itself, a silica aerogel Cerenkov detector (SAD) and a large drift volume with 320 sense wires for ionization measurement by sampling (ISIS). In the second lever arm a high temperature Cerenkov counter (FC) and a transition radiation detector (TRD).
- Two lead glass electromagnetic calorimeters: An intermediate gamma detector (IGD) in the first lever arm and a forward gamma detector (FGD) in the second lever arm.
- Two neutral calorimeters(INC and FNC).

In June 1980 a running-in experiment (NA16) was made with an incomplete setup, a provisional high resolution 20 cm bubble chamber (LEBC) and a provisional magnet (M1') downstream of M1.(3)

Charm particles were scanned and fully reconstructed within a year and final results were presented at the 1982 Paris conference.(4)

Three years ago several Japanese universities joined the collaboration

and actively participated in the experiment NA23 and NA27. Dr. R. Hamatsu (TMU) is coordinating the work in Japan for those two experiments and for two years Dr. S. Kitamura (TMU) has been a most efficient collaborator in the software team. The NA16 results, together with preliminary results from NA27, experimentally confirm the expected performance of the setup for its most critical components:

Holebc: Bubbles 20 μm diameter
space reconstruction 10 μ (in main deflection plane)

Spectrometer: $\Delta p/p \sim 1\%$ for the whole range of momenta,
mass resolution: 2.5 MeV/c $\sim$ for K^0, 1.1 MeV/c $\sim$ for Λ^0

Gamma detectors: $\Delta E/E \sim 0.15/\sqrt{E} + 0.02$ in the range 2 to 40 GeV
for the IGD and the FGD.

Isis: Geometrical information for through tracks:
position <0.2 cm, direction <0.5 mrd.
Ionization measurement $\Delta I/I \sim 7\%$ FWHM.

The dimension of the EHS, the use of a large variety of detectors with alternative configurations and the physics requirement for great accuracy made 'a priori' the conception and the realisation of the software a rather attractive job. The developments which have taken place from 1975 to 1980 have all been oriented towards the same goal, how to best satisfy the tasks reviewed in section II.

This has required many iterations with the constructors, the survey group, the on-line team and the physicists. The successive steps have been:

- the first generation of simulation programs for the preparation of the proposal, with special studies of acceptance and performance of the magnets and spectrometer.(5)
- The optimization of the drift chamber geometry.(3)
- The composition of the team, the definition of the program main components and the sharing of responsibilities (about four people full time at CERN, including the coordinator, and about eight people part-time outside in five different laboratories).
- The choice of a programming system, PATCHY/HYDRA, which was considered at that time (and still is today, if not tomorrow) the only system able to accommodate the complexity of the EHS data handling.
- The agreement on the concept of detector structure to carry all

geometrical and physical characteristics of the different detectors.

- The interface on-line/off-line with several types of records (KEY, EVENT, STATUS, monitoring ...).
- The definition of the user data substructures and of their relationship (Input, Output and Intermediate structures).
- A pool of 'title' factory programs for the tuning of some critical parameters (→ ACCURACY).
- The second generation of simulation programs with simulation of data for all devices, including bubble chamber film measurement, with different models of event generators (→ CORRECTNESS).
- A coherent production chain, with the successive program steps PRECIS (formating), GEOHYB (BC & spectrometer), GAMIN (electromagnetic showers), CALOR (hadron showers), PARTID (particle identification), KINEHS (kinematics) and more recently the development of rescue programs and, for the charm experiments, the use of graphic technics (→ COMPLETNESS).
- A scheme for automatic control of those parameters which may vary during the run, the Title Data Base which gives both RELIABILITY and SIMPLICITY of operation.
- A pool of programs to control the quality of reconstruction and to study the residual systematic effects (→ ACCURACY).
- A pool of programs for the control of data taking, by sampling events and processing them at the central computer through the CERNET link, with display of the results in the counting house.

Most of these developments have been done in the same frame of work, with access to unified data structures. This makes the EHS software somewhat unique for its coherence and its power, at least on paper. Fig. 2 shows the flow chart of the program chain. It waa designed in 1976 and the execution of the work has followed without any significant change. In practice, the maintenance, not to say the development, of the software has been affected by a degradation in the available manpower and by a lack of continuity and the situation today is far to be idyllic.

It is instructive to try 'a posteriori' to understand why a few things went wrong, and to do it in fairness to collaborators whose tasks were often complex and who have worked very hard. Having been in charge of the coordination of the software from 1975 to 1981 I am of

course responsible for many weaknesses. But it is difficult to judge objectively one's own work.

Most often the grey or black areas correspond to developments which have been done without obeying the team rules. There are for instance at least two occasions where products have been put on the market without 'a priori' discussions and this has caused some difficulties to integrate them efficiently. In other situations, parts of programs have been written for a specific application without any effort to generalization and later the job had to be redone in a slightly different context. The lack of continuity in the team has been most damaging, and this was enhanced by the lack of good documentation.

Other sources of trouble are probably due to weaknesses in the overall project coordination: gadgets impossible to survey when put in their final position, lousy hardware read-out conventions, parts of on-line/off-line interface not defined early enough, no serious checklist procedure when starting the runs, reluctance of the physicists to execute special calibration tests, errors in the design of the setup ... .

The last, but not least, source of problems has been the occurence of hardware failures. These are most likely unavoidable and the software should be designed with enough flexibility to absorb smoothly the most probable illness conditions but this is always painful and sometimes impossible.

IV Some Comments about Data Structures

When the variety and the complexity of a setup increases, many people have to contribute to the software. Each one should be given as much freedom as reasonable. The need to assemble at some point in time the different contributions requires a great modularity of the programs and well defined interfaces.

Furthermore, changes in the hardware main components, or simply the addition of a new component, and changes in the running conditions are very frequent with such large detectors. This is a second plea for modularity.

The best way to implement modularity, if not the only way, is by program structuring and data structuring.

What does program structuring mean? How is it achieved?

It is a very natural concept, at least for cartesian minds. In simple words, the cartesian approach *consists of dividing a complex* problem into more simple ones ... and to iterate. Applied to software *it requires to identify functions which correspond to more and more* elementary tasks and to precisely specify their relationships. In practice it leads to the definition of a pool of utility routines and of a pool of programs, each program being made up of one or several 'modules', each module dealing with one or several elementary tasks.

PATCHY with its PAM/PATCH/DECK structure but mainly with the possibility it offers to plug-in alternative and/or optional source code is a very useful tool for program structuring. The concept of PROCESSOR within HYDRA has been especially designed for that purpose and is also very helpful.

What does data structuring mean? How is it achieved?

Programs, or modules, are not independant objects. They have to share information, they have to talk to each other. Data structuring consists of defining those data which go logically together and of fixing the rules which permit access to them, so as to enable the communication between modules and between programs.

Data structuring can be achieved with ordinary Fortran, by playing with arrays and pointers, or with specific memory managers as those proposed by the programming *systems* HYDRA and ZBOOK(6). There is no fundamental difference between both approaches, the use of a memory manager is simply more powerful and more friendly, and at the end more economical. In any case, unfortunately, the most difficult part of the job has to be done by the user, namely to classify the information in the appropriate way and to see clearly the relationship between the different sets of data.

In the past, large programs have been written in Fortran with a high degree of memory dynamicity, like the latest version of the bubble chamber geometry program THRESH.

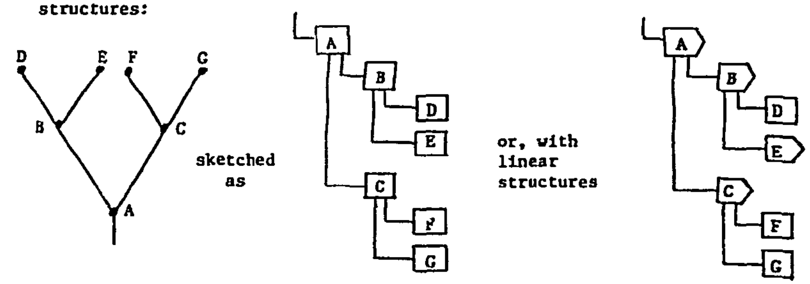
Today, for the complex setups with which we are concerned, it might still be possible to use ordinary Fortran but most likely it would quickly become unmanageable, because of the lack of flexibility and generality and also because of the amount of overhead in the user code.

New languages have been developped (PASCAL, ADA...) which offer more possibilities than Fortran for data structuring but so far none of them has been adopted by the high energy physics community.

The advantage of a memory manager is to provide the user with the tools needed to organize and to control the data but again, it does not tell the user which elementary data should be grouped together. The groups of data are refered to as data banks. They are stored in a dynamic core where they can be retrieved by simple addressing. They can be dropped when not wanted any longer, thus releasing space for other banks. They are usually chained together in data structures.

The simplest data structure is the linear one which permits easy access from any one bank to the next of same type:

Other structures can be developped which emphasize the filiation between banks, such as the tree structures, with or without linear structures:



Trees with linear structures have hardly their equivalent in nature. I leave it to Japanese people, whose expertise in growing trees has become an art, to think about it!

Quite like the cutting of a main tree prong, dropping a mother bank implies dropping of all banks which are hanging to it. Note however that any bank within a linear structure can be dropped separately as there is usually no filiation between those banks. If wanted, all banks of a linear structure can be dropped at once.

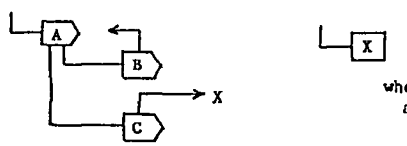
The links which appear in the above structures are called structural links. For the reason just mentioned, the links between banks of a linear structure have a specific nature. They are usually refered to as horizontal links, the others being the vertical links. Note that the linear structures in ZBOOK are pseudo str. ures as there is no explicit horizontal link in the banks themselves.

In many applications there is a real benefit to consider another type of link. For instance, some bank within a structure may have to make reference to other banks from the same structure or from other data structures, without implying any filiation. This type of link is called a reference link.

When dropping a bank, any bank which can be accessed from it through a reference link has to stay alive of course, unless it also has some structural connection with the mother bank dropped.

Hydra is the only system whose memory manager can handle, with some generality, banks which contain structural and reference links.

A data structure with both types of links can be sketched as follows:



where X belongs to another structure

Nature provides us with trees which exhibit about the same features: the Bunian, a superb Indian tree, with its apparent multi-tree configuration and its network of aerian 'roots' might be the best candidate!

The last part of this section will be devoted to three examples of structures used in the EHS software:

The Detector structure (Fig. 3)

All components of the setup are classified in Detector SETS, each containing one or several MODULES, each containing zero, one or several ELEMENTS. The corresponding DS/DM/DE structure contains the description of the geometrical and physical characteristics of the setup.

There is one set for each of the following components: Upstream lever arm, bubble chamber, magnet M1, first spectrometer lever arm, magnet M2, second spectrometer lever arm, MDC, PIC, SAD, ISIS, IGD, INC, FC, TRD, FGD and FNC.

Extension banks DSX and DMX, respectively for the DS and for the DM, have been introduced to give more flexibility. They are identified by their first word of data which contains a unique type for information

retrieval. When needed, a DS can make reference to a bank RTP which contains the transformation parameters between the proper DS reference system and any system chosen as a basic reference system. Secondary transformations from one DS to another one (both expressed in non-basic reference systems) can be stored in RTS banks. The DS link to the RTP is a reference link, just for reasons of convenience. It could as well have been a structural link.

The Event structure (Fig. 4)

This holds the whole information concerning an event: The input formated raw data or eventually preprocessed data, the event reconstruction results with detailed information on the spectrometer tracks which have been successfully linked to the bubble chamber tracks ('hybridized' tracks), on the left-over spectrometer tracks ('hanging' tracks) and on the electromagnetic and hadronic showers. For simulation events it also gives the original event characteristics and details of the simulation processes. This is most useful for program debugging to be able to step by step compare the behaviour and performance of the reconstruction programs with events whose characteristics are well known. The more sophisticated the simulation can be the greater is the benefit.

From the start of the tree, the HT bank, one can access:

- A LOG chain where the history of each program step can be stored
- The bubble chamber film measurement tree (VW chain).
- The formated raw data (RE chains) for the proportional wire chambers and the drift chambers, for the electromagnetic and hadronic calorimeters, for the particle identifiers.
- The preprocessed data from the ISIS pattern recognition program SPIRES.
- The event geometrical and kinematical results (PF chain).
- The spectrometer hybridized tracks (DPF/DTF chain).
- The spectrometer hanging tracks (DTF chain).
- The impacts in the PWC's and DC's, both the ones used in the reconstructed tracks and the unused ones (EM chain).
- The reconstructed showers (SHW, SHWN chains).

And finally, for simulated data, the description of the generated event (PF chain), the details of track following (UFO chain) and the simulated impacts in the PWC's and DC's (EM chain).

The DTF substructure (Fig. 5)

This is a typical, somewhat unorthodox, example of an elementary data structure which permits a great program modularity. The mother bank, the DTF, describes a track in the spectrometer. In addition to the horizontal link, which gives access to the other tracks, it has two vertical links which hold the EMA chain and the DTFX chain. Note that the EMA linear structure is made of banks referencing other banks which differ both in nature and in number. The EMA banks describe the structural components of the current track. Each one is identified by the first word of data which contains a unique type. The EMA of type 1, for instance, gives the details of the space impacts and of the individual hits used by the track. This is why it makes reference to intermediate EM banks where the space impacts are stored and, through the EM banks, to the RE banks which contain the raw data, the original wire hits. The impacts in the EM banks are obtained in a stand-alone preliminary step before they are combined to form track segments. Their definition and coordinates may differ from those of the corresponding impact as used in a track context. In this last case the impact definition is more precise, permitting for instance resolution of an original left-right ambiguity for a hit with a small drift time in a given chamber plane. The contents of the EM bank should not be altered as the same impact could be used again in an attempt to reconstruct other tracks. Instead, the characteristics of the impacts used for a given track are stored in the EMA bank. The same remarks apply to the other types of EMA banks, for instance the one used to establish the connection with ISIS or, when the particle gives a shower, for the connection with the Gamma Detectors.

The DTFX chain contains all kinds of intermediate results which one wants, or needs, to transmit from module to module and from program to program. The DTFX banks can be considered as dynamic extensions of the DTF. As the EMA banks, they are unambiguously identified by the first word of data which contains a unique type. As examples of DTFX let us mention the DTFX of type 2 which contains the results of the spectrometer track fit and the DTFX of type 3 which contains the so-called long term track predictions, which permit the correlation with the detectors around the spectrometer like ISIS or the Gamma Detectors.

As one can see, the DTF structure contains a lot of information, accumulated at different stages of the definition of a track. When transmitted on output tape, that information can be processed through

special control programs in order to study the systematic effects, for instance by histogramming the differences between predicted and observed values of some parameter. Such an operation cannot be done in-line in the reconstruction program as the relevant part of coding will most likely be entered not only for the good tracks but also for candidates which happen to be eliminated later. The extraction of the information can take place only at the end when the best final selection has been made. This is a justification for the somewhat complex contents of these data banks.

The analysis of NA16 has given many illustrations of the validity of this approach. One example is the detection of a survey error for the location of one chamber. Another example is the correction brought to T^o, the drift chamber 'zero drift time'. Initially obtained from a separate study of the through beam tracks, T^o turned out to be affected by a widening of the pulse shape due to space charge in the beam region.

V <u>Conclusion</u>

There is little doubt that off-line software should be considered as one of the corner stones of future experiments. It requires a lot of thought, a lot of effort, a lot of support. Of course the ambitions have to be adapted to the means available, but in the early phase one should try hard to adapt the means to the ambitions!

There is no chance of real success without a team and without a complete consensus between its members.

Program structuring and data structuring seem to be necessary. The definition of data structures and the contents of banks require special care. But there are other critical problems, uneasy to solve, like the best way to handle parameters which vary during the run.

There is a great benefit to develop, in the same frame of work, the simulation programs, the reconstruction programs and the graphics analysis programs.

A good documentation of the software is essential. May be it should be circulated before the programs are released and updated every time a new program version is dispatched.

These are simple conclusions. They don't have the enlightenment nor the permanence of the teaching of Buddha! They might nevertheless have some validity for the organization of the software in future large experimental setups.

Figure captions

1. Present version of EHS with HOLEBC or RCBC
2. EHS off-line program chain
3. The Detector structure
4. The Event structure
5. The DTF substructure

References

1. PATCHY Manual. J. ZOLL et al. CERN Program Library.
2. HYDRA Topical Manuals. J. ZOLL et al. CERN Program Library.
3. The European hybrid spectrometer - A facility to study multihadron events produced in high energy interactions.
   M. Aguilar-Benitez et al., Nuclear Inst. Meth. 205(1983) 79-97
4. - Lifetime measurement of charm mesons produced in π^-p and pp interactions at 360 GeV/c.
   M. Aguilar-Benitez et al., Phys. Lett. 122B (312-316)
   - Charm D-meson production in 360 GeV/c π^-p interactions - evidence for leading quarks.
   M. Aguilar-Benitez et al., Phys. Lett. 123B (98-102)
   - Charm D-meson production in 360 GeV/c pp interactions - comparison with π^-p at the same energy.
   M. Aguilar-Benitez et al., Phys. Lett. 123B (103-107)
5. Proposal P42 to SPSC - L. Montanet CERN/EP
6. ZBOOK Manual. R. Brun et al. CERN Program Library.

EHS OFF-LINE PROGRAM CHAIN

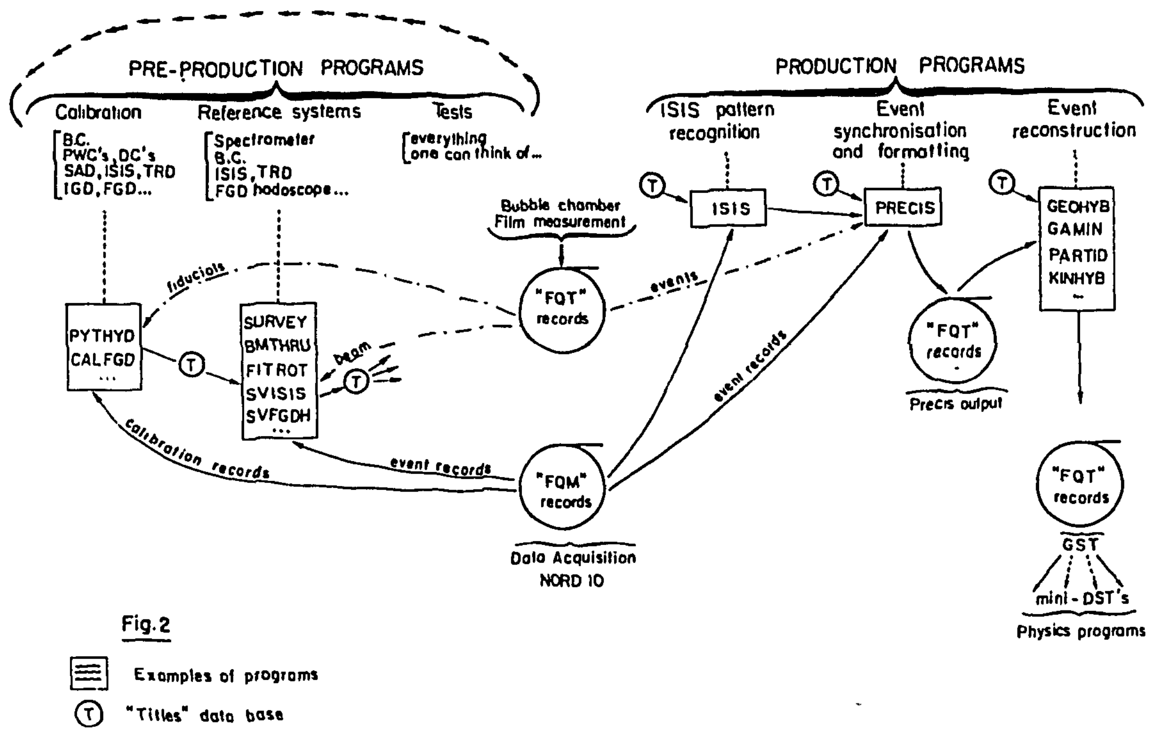
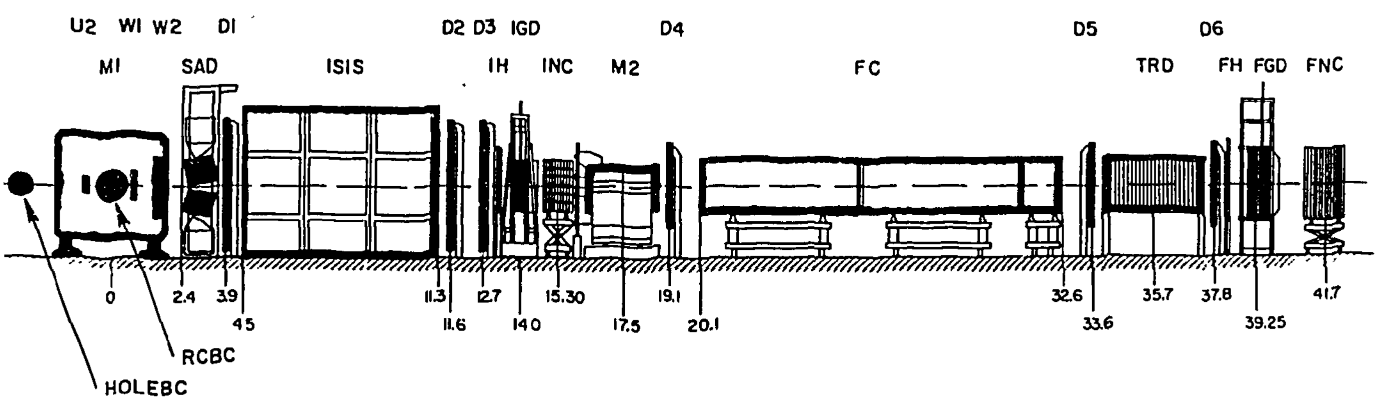


Fig. 2



PRESENT VERSION OF EHS + HOLEBC OR RCBC.

THE DETECTOR STRUCTURE

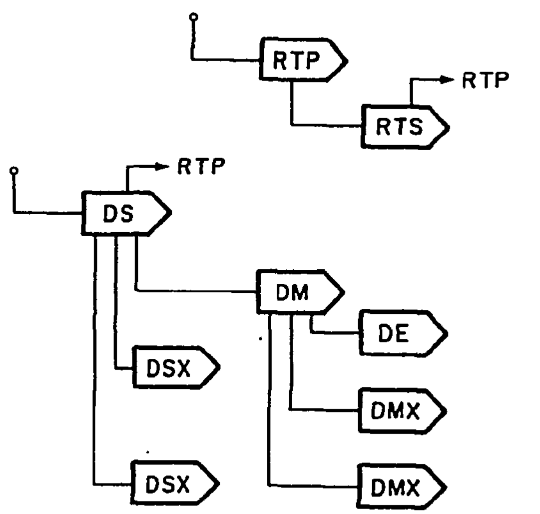


Fig. 3

THE EVENT STRUCTURE

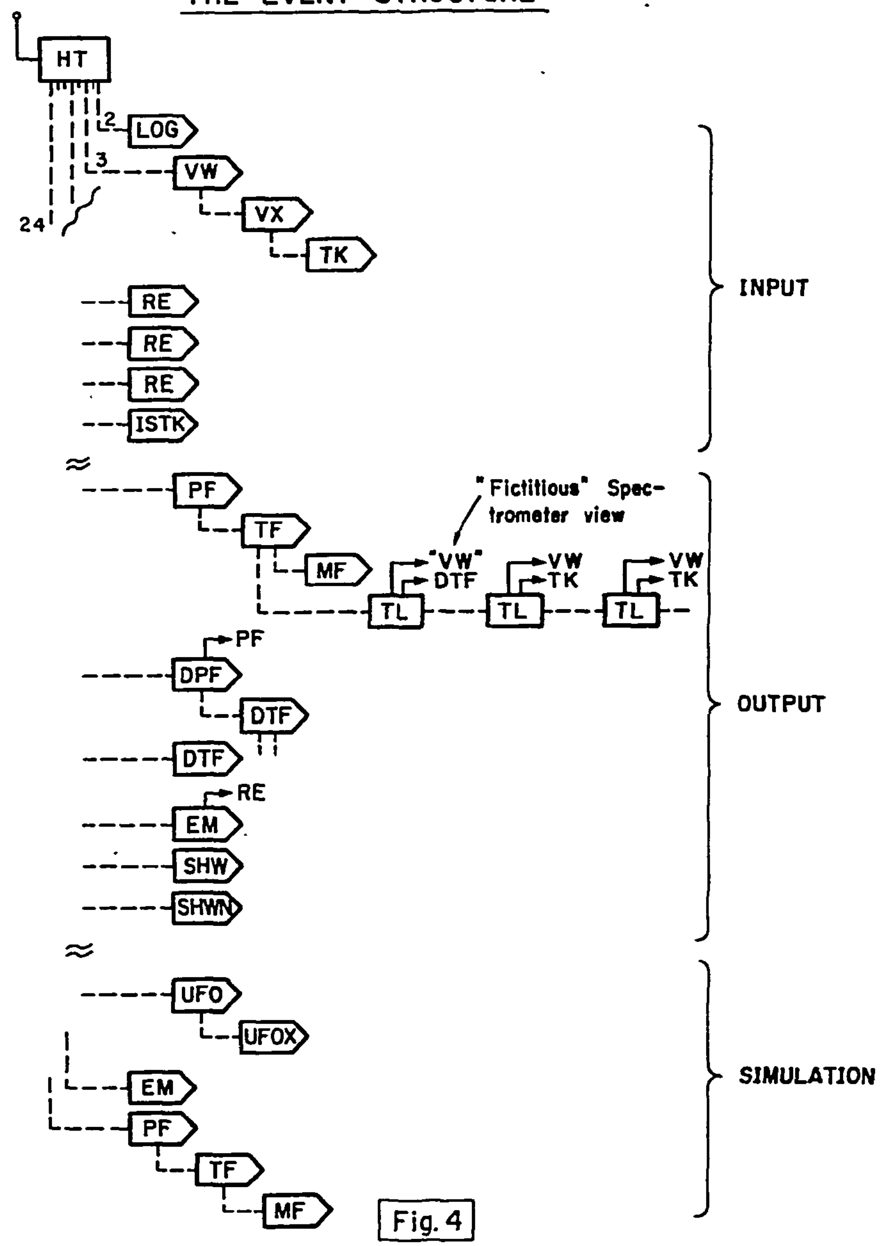


Fig. 4

THE DTF SUBSTRUCTURE

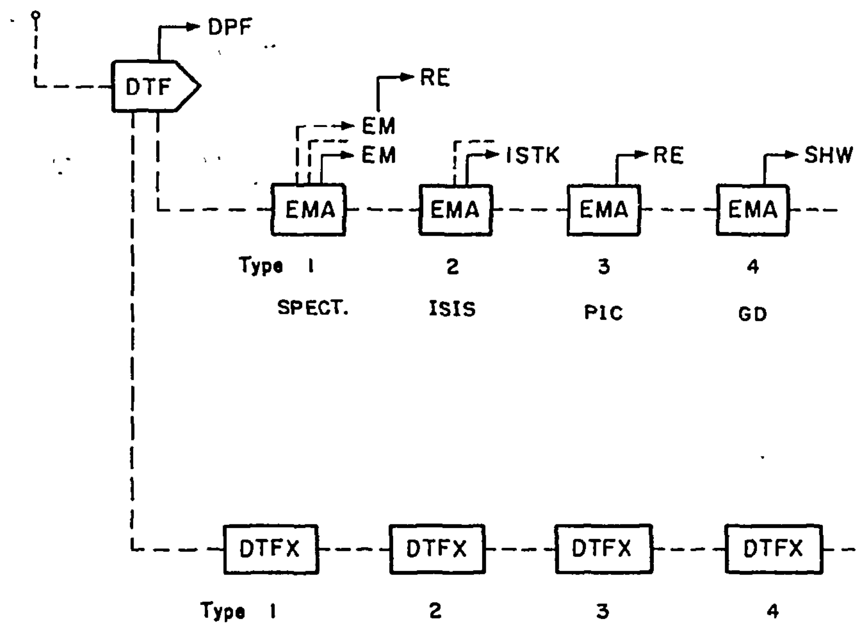


Fig. 5