Data acquisition and monitoring for the KLOE detector

(draft version)

Revision history

A. Aloisio,[g] F. Ambrosino,[g] M. Antonelli,[c] C. Bini[j] V. Bocci,[j] F. Bossi,[c] P. Branchini,[l,‡] G. Cabibbo,[j] R. Caloi,[j] A. Cardini,[j] M. Casarsa,[n] G. Cataldi,[e,†] S. Cavaliere,[g] F. Cevenini,[g] P. Ciambrone,[c] E. De Lucia,[j] G. De Robertis,[a] P. De Simone,[c] S. Dell'Agnello,[c] A. Denig,[d] A. Di Domenico,[j] C. Di Donato,[g] S. Di Falco,[i] A. Doria,[g] A. Ferrari,[l] M. L. Ferrer,[c ⋆], G. Finocchiaro,[c] D. Fiore,[g] C. Forti,[c] C. Gatti,[c] P. Gauzzi,[j] S. Giovannella,[c] E. Gorini,[e] W. Grandegger,[c] E. Graziani,[l,‡] P. Guarnaccia,[a] M. Incagli,[i] C. Kuo,[d] G. Lanfranchi,[c] J. Lee-Franzini,[c,m] M. Martemianov,[c,f] A. Martini,[c] W. Mei,[c] L. Merola,[g] R. Messi,[k] S. Miscetti,[c] M. Moulson,[c] S. Müller,[d] F. Murtas,[c] L. Pacciani,[k] M. Palutan,[c] E. Pasqualucci,[j] L. Passalacqua,[c] M. Passaseo,[j] A. Passeri,[l,‡] F. Pelucchi,[c] D. Picca,[j] G. Pirozzi,[g] L. Pontecorvo,[j] M. Primavera,[e] P. Santangelo,[c] E. Santovetti,[k] G. Saracino,[g] B. Sciascia,[c] I. Sfiligoi,[c] J. Shan,[c] T. Spadaro,[c] E. Spiriti,[l,‡] C. Stanescu,[l,‡] L. Tortora,[l,‡] E. Valente,[j] P. Valente,[c] B. Valeriani,[d] G. Venanzoni,[i,†] S. Veneziano,[j] A. Ventura,[e] Y. Zhou,[c]

[a] Dipartimento di Fisica dell'Università e Sezione INFN, Bari, Italy.
[b] Permanent address: Institute of High Energy Physics of Academica Sinica, Beijing, China.
[c] Laboratori Nazionali di Frascati dell'INFN, Frascati, Italy.
[d] Institut für Experimentelle Kernphysik, Universität Karlsruhe, Germany.
[e] Dipartimento di Fisica dell'Università e Sezione INFN, Lecce, Italy.
[f] Permanent address: Institute for Theoretical and Experimental Physics, Moscow, Russia.
[g] Dipartimento di Scienze Fisiche dell'Università e Sezione INFN, Napoli, Italy.
[h] Physics Department, Columbia University, New York, USA.
[i] Dipartimento di Fisica dell'Università e Sezione INFN, Pisa, Italy.
[j] Dipartimento di Fisica dell'Università e Sezione INFN, Roma I, Italy.
[k] Dipartimento di Fisica dell'Università e Sezione INFN, Roma II, Italy.
[l] Dipartimento di Fisica dell'Università e Sezione INFN, Roma III, Italy.
[m] Physics Department, State University of New York at Stony Brook, USA.
[n] Dipartimento di Fisica dell'Università e Sezione INFN, Trieste, Italy.
[†] formerly Institut für Experimentelle Kernphysik, Universität Karlsruhe, Germany.
[‡] formerly Istituto Superiore di Sanità and Sezione INFN, ISS, Roma,Italy.

[⋆] corresponding author: Maria Lorenza Ferrer, INFN-Laboratori Nazionali di Frascati, Via Enrico Fermi 40, 00044 Frascati, Italia; tel. ++39-06-94032770, e-mail ferrer@lnf.infn.it

Preprint submitted to Elsevier Preprint *11 February 2003*

Abstract

The Data Acquisition system for the KLOE experiment, presently running at the Laboratori Nazionali di Frascati DAΦNE collider, has been designed to sustain an acquisition throughput of 50 MByte/s for an event rate of 10 kHz. Its two major components are the front end data readout, based on custom buses, and a complex network of computers and storage devices hosting a set of distributed processes. The end result is a seamless data transport from the readout system to the storage library, accompanied by concurrent on line calibrations and data quality control.

Keywords: Data acquisition; high throughput; online monitoring; data archiving; database

PACS: 29.40.Cs ????

1 Introduction

The KLOE experiment is installed at the Frascati DAΦNE Φ-factory [1], a high luminosity double-ring e^+e^- collider designed to attain a peak luminosity of 5×10^{32} cm^{-2} s^{-1} with 120 x 2 bunches and an interbunch crossing time of 2.7 ns. The collider is optimized for operating at a total center of mass energy of 1020 MeV, the mass of the ϕ meson, whose main decay modes are into a pair of charged or neutral kaons. The visible cross section for the process $e^+e^- \to \phi$ at the energies of the ϕ mass is approximately 3.3 μb which corresponds to a production rate of 1700 ϕ mesons per second at DAΦNE peak luminosity. The major motivation of the experiment is to study several aspects of kaon physics, with particular emphasis on CP-violation in the $K_{L,S}$ decays. Other decay modes of the ϕ are also of interest, therefore it is desirable to collect the largest possible fraction of the ϕ's being produced. Another important reaction from e^+e^- collisions is the well known Bhabha scattering, (Bhabha) $e^+e^- \to e^+e^-$ (and also $e^+e^- \to \gamma\gamma$). Due to their high production rate ($\approx$ 15 kHz within the detector's acceptance at the nominal luminosity) and their characteristic signature, Bhabha events are well suited for real time luminosity measurement and detector calibration and monitoring. Another class of events that can be exploited for online calibration are cosmic rays (cosmics), crossing the detector at a rate of approximately 3 kHz. Finally, trigger and data acquisition have to contend with unavoidable fluxes of machine background. The Data Acquisition system was designed to handle with high efficiency such event rates, and to combine the tasks of data logging with those of real time detector calibration and data quality control.

2 The KLOE detector

The KLOE detector [2] has a cylindrical structure surrounding the collider beam pipe, see Fig. 1, and consists of:

- a large, highly efficient drift chamber[5] for measuring trajectories and momenta of charged particles;
- an electromagnetic calorimeter (Barrel and End Cap)[3] with excellent timing capability, to measure the energy deposit and the impact point of photons and charged particles hitting it;
- a second electromagnetic calorimeter (QCAL)[7] fitting in the narrow space between the drift chamber and the beam focusing quadrupoles, to improve the acceptance and hermeticity of the detector.

The detector is immersed in the magnetic field of a superconducting coil of 2.5 m inner radius and 4.2 m length, capable of achieving field values as high as 0.6 Tesla (a value of 0.53 T is the experiment's normal running mode). The drift chamber provides tracking of charged particles in three dimensions,

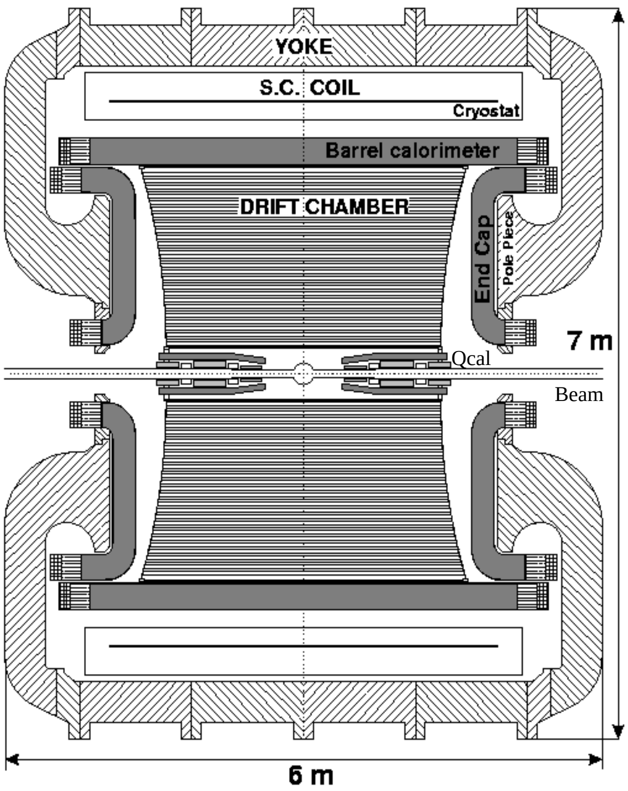

Fig. 1. Vertical cross section of the KLOE detector along the beam line

with a resolution in determining the $K_{L,S}$ decay vertices of 200 μm $\times$1mm over the whole sensitive volume. It also provides a good momentum resolution

($\Delta p_t/p_t \sim 0.4\%$) for low momentum tracks, and a fast trigger to complement the calorimeter based one. The drift chamber is a cylindrical structure of carbon fibers, containing a total of 52,140 between field and sense wires, organized to provide 12,582 drift cells. The digitization of drift times is performed by the chamber TDCs, based on a fully digital chip that was developed for KLOE. Pulse height information is also recorded by custom ADCs.

The basic structure of the electromagnetic calorimeter consists of 0.5 mm lead plates encapsulating 1 mm diameter scintillating fibers. Calorimeter modules are organized in rectangular (Barrel) or half disk (Endcap) modules, with fibers running parallel to the beam or in parallel half circles.

At both ends of each module, the light from the fibers belonging to adjacent elements is collected by photomultipliers, a total of 4880 for the whole calorimeter. Each photomultiplier signal is split three ways to provide the following functions:

- amplitude measurements via ADC;
- time measurement via TDC;
- trigger generation after analog sum of signals.

The calorimeter energy resolution for photons is $5.7\%/\sqrt{E(GeV)}$.

The QCAL is a tile calorimeter designed to offer a good photon detection efficiency (92 %) in the energy range (20 - 280) MeV, with a time resolution of 240 $ps/\sqrt{E(GeV)}$. QCAL is read out by 32 photomultipliers, each one feeding an ADC channel.

The Trigger system[8] is based on local energy deposits in the calorimeter and hit multiplicity information from the Drift Chamber. It provides a good signal acceptance and background rejection, Bhabha events downscale, cosmics veto and a fixed dead time following each trigger. The trigger hardware consists of 13 different types of custom-designed electronics modules, needed to generate the elementary trigger signals from the various subdetectors, or to produce the final trigger signals to be delivered to the Front End Electronics (FEE). A two level scheme has been adopted: a first early trigger signal (T1) starts calorimeter FEE digitization, while a second level signal (T2), delayed 1.8 μs with respect to T1, confirms the first level and starts both the digitization of the drift chamber FEE and the Data Acquisition read out.

The first level, originated within 350 ns from the bunch crossing and synchronized within 50 ps with the machine radiofrequency, requires 2 energy deposits above threshold in the calorimeter or 13 hits within 250 ns in the Drift Chamber. The Bhabha downscale is implemented at this level. The second level requires 120 hits within ~ 1 μs in the Drift Chamber. The cosmic veto is implemented at this level.

3 Data acquisition overview

The data acquisition, DAQ, of the KLOE detector was designed to sustain a trigger rate of 10^4 events per seconds, resulting from the combination of ϕ decays, down scaled Bhabha scattering, un-vetoed cosmic rays and residual machine background. This high trigger rate in turn imposes a strong requirement onto the readout system, which should be able to collect FEE data in average times much less than 100 μs. Taking into account the MonteCarlo prediction of an average event size of 3.5 KByte, the DAQ system was conservatively designed to sustain a total bandwidth of 50 Mbyte/s throughout the whole data path, from readout system to storage devices. Such a performance was achieved via the cooperation of processes running in a set of computers interconnected through high performance switched networks, a fast messaging system and a dynamically configurable data flow controller. Under the controller's management, events are distributed to different nodes of an online farm, responsible for event building, formatting and recording on disk. All these tasks need to be performed without adding dead time and assuring the completeness of every collected event.

In addition to the proper DAQ tasks, the on-line system must also perform a parallel activity of data monitoring and detector calibration. The online selection of Bhabha events is exploited to both calibrate periodically the calorimeter time and energy scales, and to determine the tracking chamber parameters. Moreover, a fast event reconstruction is required to monitor data quality in real time. To this goal, dedicated processes running in the online farm select Bhabha, cosmics and other interesting events from the main data path, copying them in temporary buffers, to be analyzed by suitable monitoring processes running in dedicated nodes.

It is expected that about 10^{11} events will be acquired, stored and analyzed during the KLOE lifetime. This requirement in data handling capability represents a strong challenge, since it surpasses the magnitudes typical of high energy collider experiments. The DAQ architecture uses a tape library to store data and system backups, a commercial database to log run conditions and file histories, and commercial software for data management, all of them well interfaced to the KLOE software environment. Enough disk space (approx 1.7 Tbyte) is connected to the online farm to allow for an immediate offline reconstruction, avoiding the need to stage files from tape.

In order to allow for periodic system upgrades according to real DAΦNE luminosity and new computer availability during the KLOE lifetime, DAQ software components were designed to assure portability to different computer platforms running UNIX operating system. Moreover a powerful monitoring system controlling performances of different components allows scaling of the system up to the maximum expected rate by simply increasing the number of network channels and/or the aggregated computing power in the online farm.

Since the major aim of KLOE is to perform studies at sensitivities of the order of 10^{-4}, DAQ error rates which might be quite acceptable in other situations, must be minimized, especially if dependent on event configuration. In particular, the possible build-up of instantaneous backlogs that might result in losses of parts of an event has to be avoided.

3.1 Data acquisition architecture

The DAQ readout system involves some 23,000 channels of Front End Electronics, FEE, from calorimeters, tracking chamber and trigger. For each trigger, relevant data coming from the whole FEE have to be concentrated in a single CPU, where a dedicated process is in charge of building the complete event. To this purpose, a three level data concentration scheme has been implemented.

The first two levels (L1 and L2 in the following, see Fig. 2) respectively gather single event data within a single FEE crate and then combine the information from several crates. Both these levels rely on custom buses and hardware controllers, in order to achieve high bandwidth without software penalties and CPU overheads. FEE and readout controllers use standard VME and custom boards but, while the VME protocol is used for FEE initialization and monitoring procedures, first level data readout within an FEE crate uses a custom bus, AUXbus, built around the free pins in the VME P2 connector. At the second level, intercrate data transmission is accomplished via a custom cable bus (Cbus). The third level, responsible for final event building and subsequent monitor and storage tasks, is implemented by means of commercial CPUs. Data transmission between CPUs relies on standard network media and protocols, and it works with packets of events in order to optimize the use of the network channels.

The DAQ components and the information flow are presented in more detail in the following. After receiving a trigger signal, each FEE channel performs analog signal conditioning and digitization during a fixed 2.2 μs dead time, after which the system is ready to receive a new trigger. Every FEE board contains a local trigger counter and a buffer of appropriate length, where each event number and the related data are memorized. A crate readout controller, ROCK[9], memorizes trigger counters in a local buffer and acquires FEE event data sequentially and asynchronously with respect to the trigger signal. The number of boards per L1 crate is optimized by taking into account the expected average occupancy, in order to ensure a readout time much lower than 100

μs. Variable numbers of readout controllers are chained using the Cbus[9]. The chain is terminated in a VME custom manager of readout controllers, ROCKM[9], which concentrates the data of each event in a local L2 buffer. This buffer is then read out by an VME CPU (a Compaq Alpha-based processor). Event data coming from a chain is referred to as a sub-event.

The interface between the trigger system and the acquisition chains is concentrated in a single VME crate, the Trigger Supervisor, connected to the readout controllers by means of fan-in/fan-out units. The busy and fault conditions are managed by the Trigger Supervisor, which inhibits the trigger system at their occurrence. Periodic synchronization cycles are started by the Trigger Supervisor in order to check the alignment between trigger counters in all the acquisition boards.

As mentioned before, the KLOE DAQ has been designed to sustain a throughput of 50 MByte/s all along the 'DAQ path', from the readout controllers to the storage system. A potential bottleneck in this path, where sub-events are routed to a single CPU unit for final event building, is managed by a 22 port FDDI switch. The function of the switch is to connect the VME CPUs in the Level 2 crates serving the 10 FEE chains to an online farm of seven SMP servers (IBM H50, 4 way 332 MHz PowerPC), where event building, formatting, monitoring and disk storage processes are implemented.

Packets of 100 sub-events are buffered in the VME CPUs and then transmitted through the FDDI using TCP/IP connections. The destination node in the farm is assigned cyclically by a central Data Flow Controller, DFC [19], that distributes to the VME CPUs trigger-destination tables via a commercial interconnect bus with mirrored memories, VICbus [4]. Each DFC table contains a flag for any node in the online farm, enabling or disabling each node participation into the data acquisition, and a word indicating the highest trigger number for which the table is still valid. This capability of redirecting the network traffic in case of sporadic high occupancy of a node strongly reduces DAQ dead times.

The Run Control process, residing in an SMP 4-ways server, centralizes the functions related to the control of the run activity and provides the graphic operator interfaces for DAQ monitoring and error logs. This node also acts as the NFS (Network File Systems) disk server of the acquisition network, where software, calibration data and history files containing run conditions and detector/accelerator parameters are recorded.

The Slow Control process, running in a VME CPU placed in a dedicated crate, is in charge of setting and monitoring detector conditions. Moreover it continuously collects in history files a set of relevant detector and accelerator parameters and provides graphic interfaces for their presentation.

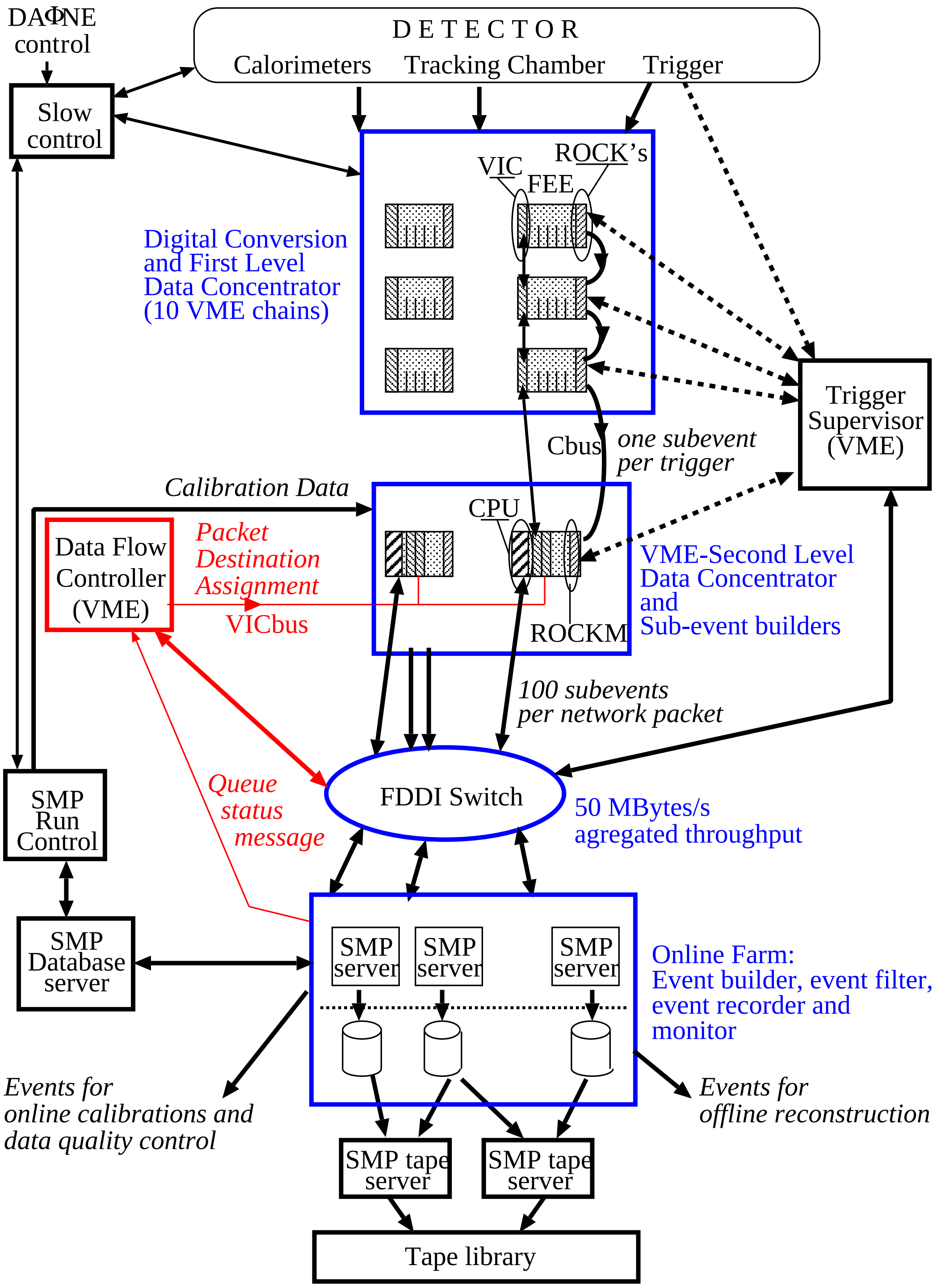

Fig. 2. DAQ architecture

A Database server keeps track of run conditions, together with accelerator and detector status for any run. Relevant parameters related to any acquired or analyzed event file are also entered in the database. A second Database server is used to store calibrations constants.

The online farm provides functionalities of event building and event recording in local disks. In parallel, specialized processes are in charge of selecting interesting events, to be used for monitoring the quality of the acquired data and the goodness of the detector calibration constants. Some crucial parameters resulting from this early data analysis (e.g. instantaneous luminosity, beam energy, etc) are fed back in real time to the operators in the control room of the DAΦNE collider.

Each node in the farm exports via NFS its own disk to an offline farm of servers, where the full off-line analysis is started as soon as the goodness of the calibration data is confirmed.

A Gigabit and Fast Ethernet switched network interconnects all the KLOE computers and is used in parallel with the 'DAQ path' for traffic not related to the data acquisition. Two SMP servers with Gigabit Ethernet interfaces are dedicated to archive files from the online disks onto a Tape Library.

4 Hardware components

A short presentation of the DAQ hardware components is given in the following, with particular emphasis on the KLOE custom devices.

4.1 Readout system

The Level 1 and 2 DAQ consists of a modular structure of VME crates tied by a vertical connection. Each Level 1 crate contains a VME Master ("VIC"), up to 16 DAQ boards and a local readout controller, the ROCK. The vertical connection (a "chain") links up to 8 ROCK boards to a chain controller - the ROCK manager, ROCKM - housed in the Level 2 VME crate.

The Level 1 process (Fig. 2) is an event-driven environment, where both the DAQ slave boards (ADCs and TDCs) and the controllers share the same trigger pulses. When a trigger is issued, slave boards digitize the data, label it with the respective event number and then push the frame in a data queue. Controllers store the event number into a trigger queue and then proceed to read out the slave boards on an event-by-event basis, retrieving the event number to be processed from the trigger queue.

In their simplest form, both L1 and L2 can be thought of as a distributed state machine (Fig. 3). Both ROCK and ROCK Manager controllers can be conveniently described using a five-state bubble diagram, where the next-state equations are functions of the presence of an event number to be processed. In Level 1, when a trigger is received, each ROCK board moves from the Idle to the Header state. A frame is opened in the data queue by writing in the

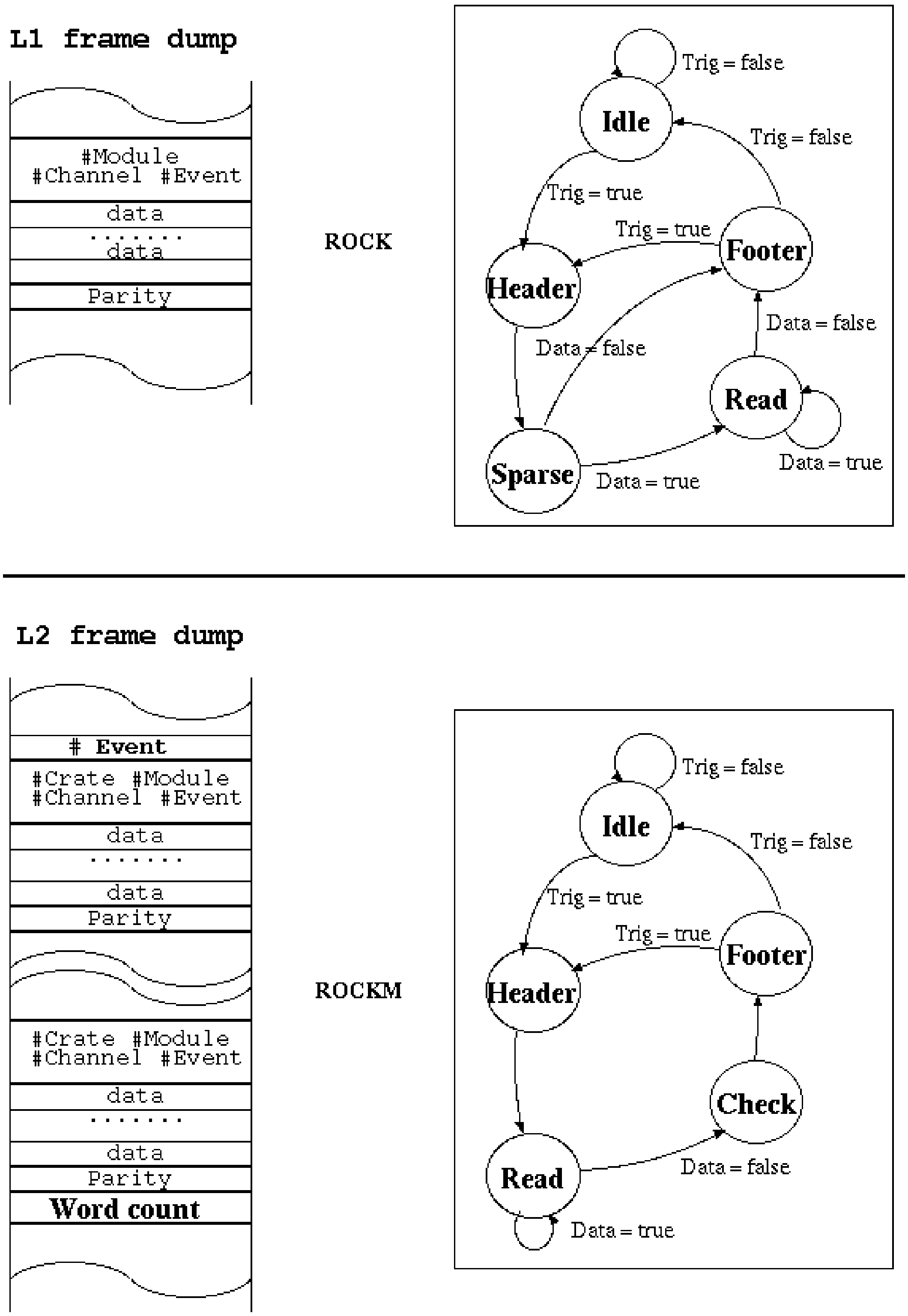

Fig. 3. ROCK and ROCKM state machines and frames dump

header field the event number and the assigned crate address; the controller then enters the Sparse state. A sparse-data-scan cycle starts in order to locate the boards in the crate with valid data pertaining to the event number in progress. The boards are read-out in the Read state and eventually a parity word is appended to the data frame in the Footer state. The Read state is skipped if no boards with data are found in the Sparse state and the frame is closed by writing the footer immediately after the header (signifying an empty frame). The next state can be Header or Idle depending upon the presence of an event number in the trigger queue.

In a similar fashion, the ROCKM moves to the Header state and opens a frame in its own data queue by writing as a header the event number, and then it enters the Read state. Each ROCK in the chain is requested to send the data frame built for the event number in progress. The Manager verifies the ROCK frames' integrity in the Check state using the parity word and finally appends the total word count in the Footer state. As before, the next state can be Header or Idle depending upon the status of the trigger queue.

Specific tools are needed to implement the L1/L2 architecture in an efficient way. The event-driven environment requires labelling the data transaction with the current event number both at crate and chain level. Sparse data scan cycles have to be performed to skip the empty boards. Each DAQ board should be read out in blocks, superseding the fixed-length transfer with a random-length block transfer. Event number alignment and data integrity are major issues in the chain design. A misalignment in the event number counters forces the controllers to an invalid state, crashing the system. Error detection needs to be implemented in order to check both the event number consistency and data corruption. Since software protocols and CPU-driven operations would slow-down the entire system even if state-of-the-art devices and communication channels were used, the L1/L2 readout architecture relies purely on custom buses and hardware controllers. CPU intervention is needed only for the initialization procedures at the beginning of a run, when all the information concerning the slave boards is downloaded from a database through the VICs.

When acting as the L1 readout controller, the ROCK performs data acquisition via the AUXbus, a custom, high speed parallel bus, which uses VME/J2 user defined TTL lines. The AUXbus' main features are data cycles labeled by event numbers, sparse data scan operations and an asynchronous protocol, optimized to achieve sustained data transfer rates in excess of 10 MHz. Data gathered on the AUXbus is framed in the ROCK's dual port buffer. Each frame starts with the header containing crate address and event number, followed by data words, and is completed by a parity word.

The ROCKM acts as the VME Level-2 chain controller, performing chain level readout on the Cbus, a custom cable bus with a daisy chain architecture and synchronous, random-length, block transfer capabilities. The Cbus is implemented using 50 twisted pair lines carrying TTL differential signals. The main features of the Cbus are data cycles labeled by event numbers and a token-like communication protocol to transfer data from the ROCK's FIFOs to the ROCKM dual port buffer.

Once framed in the ROCK Manager memory, data are read out by the VME processor, running in the UNIX environment, using block transfers, BLT. The BLT cycle length is dynamically optimized according to the effective throughput of the chain. The time to accomplish a BLT cycle is a function of many factors, such as:

- the time needed by the kernel to start-up the BLT engine;
- the duration of the atomic BLT;
- the size of free contiguous memory. When not enough contiguous memory exists to complete a BLT operation, the Unix kernel suspends it until new contiguous memory becomes available.

Some of these quantities, such as the maximum length for the atomic BLT (256 Byte), are defined by the IEEE standard, while other quantities have to be optimized taking into consideration both hardware constraints and system performances. In the ideal case we aim for the shortest possible initialization time and a very long BLT, delivering data to an infinitely long contiguous memory. An upper limit to the BLT magnitude is given by the size of the ROCKM FIFO (128 kByte); more precisely, for safety reason, FIFO readout is initiated when the "half full" bit is set, therefore due to the fact that the BLT magnitude has to be defined before the start of the operation, the maximum block transfer is limited to 64 kByte.

The strategy found to minimize the time needed by the kernel to initialize the DMA engine has been to allocate a huge amount of contiguous memory (64 MByte) at boot time and use it when needed by the DMA engine to write data on. In this way whenever a BLT takes place the kernel must only program the PCI register of the DMA engine and assert a start to perform the BLT operation.

This memory is also used to build frames containing streams of sub-events, corresponding to a set of 100 consecutive triggers that should be sent to a given node in the online farm. The strategy found in order to reduce the number of copies in memory before data transmission is described the following. Once a BLT operation is completed, data are scanned looking for the start of the successive set of 100 triggers. A footer and a new header containing the address of the destination node are inserted, moving only the remaining data and setting the start of the BLT operation to the first free position. When not enough space remains in memory to accommodate the maximum expected size of a 100 sub-events stream, the header is written at the first position in memory if the old frame was already transmitted. A temporary trigger-busy condition is generated when no available buffer exists.

Extensive tests were able to certify that the KLOE readout system offers very high performances both in terms of reliability and throughput. As an example, the system could withstand dependably trigger rates as high as 90 kHz, with an average sub-event size of 0.1 KByte.

4.2 ADCs and TDCs

Of the ten FEE chains comprising the readout system, two are dedicated to trigger modules, the remaining eight being assigned to Drift Chamber and Calorimeter ADCs and TDCs. The Calorimeter TDC and ADC modules, [3] [34], developed in collaboration with the CAEN [31] company, digitize energy deposition and arrival time, while the Chamber modules [5] [6], developed by the KLOE collaboration, digitize integrated charge and drift time. Their more relevant functionalities are:

- empty channel detection/suppression, via table look-up;
- hardware offset subtraction, via table look-up;
- channel masking functionalities;
- multi event buffering to allow asynchronous fast readout of each card;
- 12 bit local trigger counter;
- standard VME interface for initialization and monitoring;
- custom interface to the ROCK, using the free rows in the VME P2 connector.

4.2.1 Calorimeter ADC and TDC

Since DAΦNE is a relatively low energy (2 x 510 MeV) collider, particles produced in a typical KLOE event have a broad spectrum of velocities ($0.1 \geq \beta \leq 1.0$) and reach the calorimeter in a time spread as large as 200 ns from the time of the interaction. The TDCs operate in a common start mode, the start being provided by the main trigger signal T1, synchronized with the bunch crossing time and arriving at a rate of $\sim$ 20 kHz, some 2-300 ns after the collision. In order to compensate for the T1 delay, each calorimeter signal, properly shaped by the FEE discriminators [35], is delayed by $\sim$ 300 ns and then used as a stop. Every T1 signal starts the conversion process, but digitized data are discarded if no T2 signal arrives. Each TDC channel (30 channels per 9U VME card) is built around a monolithic time-to-amplitude converter, followed by a 12 bit monolithic ADC with a 1 μs conversion time. With a 220 ns full scale, the 12-bit TDC has a sensitivity of 55 ps/count. Measured performances on about 3000 channels yielded an integral non linearity of less than 0.2% of full scale and a temperature stability better than ± 0.3 count/°C.

The 12 bit ADC digitization, chosen for uniformity with the TDC, features a sensitivity of 80 fC/count which, accounting for the average photomultiplier gain, corresponds to ≈0.25 photoelectrons/count yielding a 1000 photoelectron full scale. The measured performances on about 3000 channels show a pedestal RMS of 1 count, an integral non linearity of less than 0.3% of full scale and a 3% spread of the pedestal values and gain coefficients.

4.2.2 Drift Chamber ADC and TDC

A fast non-gated charge ADC has been used for the dE/dx measurement in KLOE drift chamber. Due to He-based gas mixture used in the chamber the signal is characterized by a multi-peak structure and can last up to 1.5 μs. Moreover the signal can precede the L1 trigger as much as 200ns. To overcome these problems and to maintain the readout dead time lower than 2μs a new continuous sampling charge-integrating ADC has be chosen. The incoming signal (sum of 12 wires) is shaped by an antialiasing filter, sampled and digitized at 20 MHz by a flash ADC and digitally integrated using a left-Riemann sum algorithm. The integration starts on the T1 signal. If T2 signal arrives the data are sent to output FIFO otherwise they are discarded. The ADC resolution is 9 bits and the full scale can be programmed from ∼300 pC up to 1200 pC. In this way the sensitivity ranges from 0.75 pC/count to 3 pC/count. The maximum pedestal RMS is 2 counts and the integral non linearity is lower than 0.2%.

The ADC board is a 32 channel 9U VME card with AUXbus interface.

The Drift Chamber TDC board is a 96 channel 6U VME card with AUXbus interface. It is based on a 32 channel full custom device in 0.5 μm CMOS technology, working in common start/stop mode. In KLOE, common stop mode is used.

The circuit has 500 ps resolution over a programmable time window between 64 ns and 64 μs and is capable of recording up to 16 rising or falling edges per channel. It has a double edge resolution of 5 ns and can resolve a pulse of 3 ns. Hits are stored in an event buffer as 24 bit words, time information being stored in the 16 LSBs. After a time equal to the programmed time window they are discarded if not confirmed by a stop signal.

The chip has 4 event FIFOs, in order to be able to read out events while recording a new one. It is possible to record a new event 35 ns after a stop signal. If, after this signal, the buffer contains any hit, another one is used. Empty channel skipping is automatically performed at readout time.

Integral non-linearity has been measured using a pulse generator. In the 2 μs range, an upper limit of 0.2 LSB has been found, the measurement being dominated by the generator time-base jitter.

4.3 DAQ and Trigger interface

The interface between the DAQ and the Trigger is implemented in the Trigger Supervisor, TS, crate (Fig. 4).

The Trigger logic distributes the T1 signal to the TS and receives back from it the T1ACK which, when asserted, disables further T1 generation for a fixed time (2.6 μs). If a T2YES signal is generated by the Trigger logic (within 1.5 μs from T1) the TS asserts a T2 1.8 μs after T1. The signal is sent to the "Fan in-out (FIO)" modules, which distribute it to the corresponding readout chain with a jitter of 1 ns. Special trigger cycles have been implemented in the TS in order to check that all FEE and readout controllers received the same number of triggers. Whenever this cycle is issued by the TS, a Sync signal is asserted and, in response, all the FEE modules deliver to their ROCK the last trigger number received. The ROCK checks the alignment with its own trigger number and in a similar way the ROCKMs check the alignment between all ROCKs in their chains. Should a misalignment occur, a halt signal is asserted by the TS and the trigger generation is stopped. When this happens, Run Control starts a software procedure to identify the modules originating the failure, asking all L2 CPUs to check the status of all ROCKs in the chain. The restart of the run is then managed by the operator.

The sync signal is issued once every 4096 L2 triggers at a given "golden" trigger number defined in the run configuration file. Events between two consecutive golden triggers are built in the online farm but remain buffered until the arrival of new events proving the success of the last sync cycle. The failure of the sync cycle is very rare, 1-2 times per month.

4.4 Network

The readout, dataflow control and slow control systems are managed by VME CPUs installed around the detector. To provide cross-communication, different kind of commercial interconnecting buses are used for different purposes. In order to initialize and monitor DAQ VME boards, a vertical interconnect bus (VICbus) interconnects in any acquisition chain the VIC8250 module placed in the L1 crates, while a VICbus interconnects all the VIC8251 in the L2 crates with the dataflow controller, the unit responsible for assigning the destination of acquired trigger packets. Finally, the slow control uses a VME CPU and CAENET [31] buses connecting it to all VME acquisition crates.

All the VME CPUs have a 10 Mbps Ethernet interface, connected to a network switch CISCO 3500 installed in a dedicated rack placed near the detector.

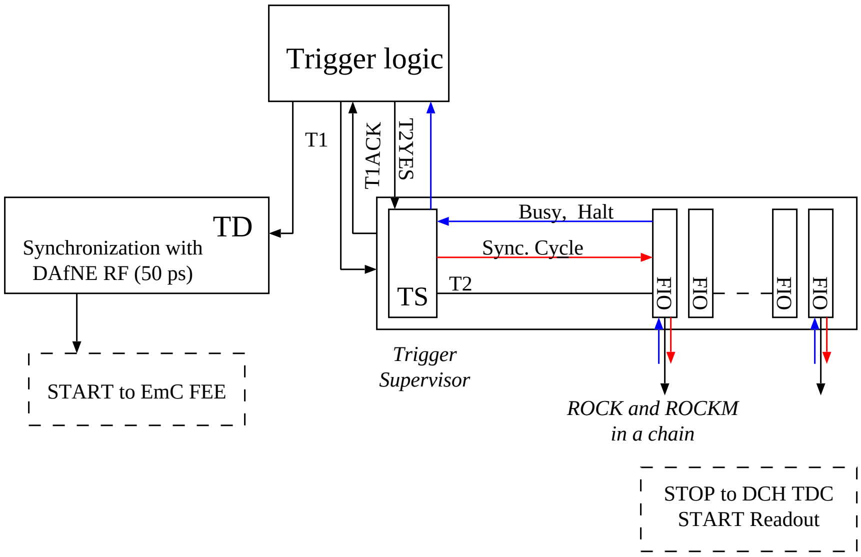

Fig. 4. DAQ and Trigger interface

Dedicated fibers connect the FDDI interfaces of the L2 CPUs to the FDDI Gigaswitch, a Digital Equipment high performance switch, located 200m away in the KLOE computing room. The above CISCO 3500 is connected via optical fiber and Gigabit interface to the central CISCO 6000 switch, located in the computing room, where the online and offline computer farms, disk servers and tape robotic are installed.

4.5 Online computing farm

The online farm consists of 7 IBM H50 servers. Each server has 4 Power PC 604 processors with a total CPU power of 48 SpecInt95, 1 GB memory, 1 SSA disk controller and 288 GB disk space (8 disks x 36 GB configured in stripping mode). Similarly to the offline farm, the disk containing the online operating system is mirrored to increase reliability. All systems are managed by a central node using the IBM NIM (Network Installation Management) package. Only 5 servers are presently used; 3 of them sustain the data acquisition throughput presented in this paper and 2 servers are used for online data quality monitoring and detector calibration,

Each online server is connected to the Gigaswitch via FDDI and to the CISCO 6500 via a FastEthernet channel built over two FastEthernet interfaces. This was motivated by the desire to

- achieve a throughput of 14 MByte/s, the present tape speed, when moving data from the server's own disk to the tape library, and
- maintain, through the same FastEthernet channel, enough residual bandwidth for data monitoring, calibration and full offline reconstruction processes.

4.6 Archiving servers and tape library

Two IBM H80 SMP systems are used as tape and disk servers. Each server (IBM H80 500 MHz RS64 III) is configured with 6 processors, 2 GB memory, 1 Gigabit Ethernet interface, 12 SSA loops disk and 3.5 TB total disk space, configured in striping mode (RAID 1) or RAID 5 when higher reliability is needed. Striping mode is used for file systems dedicated to temporary stage areas, while the archiving working area is configured in RAID 5 mode.

An IBM 3494 Tape library with 5500 cartridges of 40 GByte uncompressed capacity (to be upgraded to 60 GByte uncompressed), 12 (up to 26) tape drives Magstar E1A, dual active accessors, dual high-availability library control, and the ADSM software package are the components of the KLOE archive and backup system. In order to improve system response, tape drives are physically connected to both H80 servers, but each server manages only one half of the library capacity, maintaining its own database.

5 Software components

The acquisition software has been developed having in mind performance, scalability and independence from processor and network platforms. At the moment, the KLOE DAQ components run on HP, IBM and Compaq servers with operating systems HP-UX versions 9 and 10, AIX v4.3 and OSF/1 v4.0. Functionalities can be easily moved among different platforms with minor effort.

While the core of the data acquisition software was designed from scratch, commercial and public domain packages (see Tab. 1) are used for many applications.

Package	Application
DB2[10]	Online database
ADSM[11]	Archiving and Backup
HEPDB[12]	Calibration database
ROOT[13]	Histograms and Plots
ONX[14]	Event display
TCL/TK	Run control GUI, Monitoring tools
HTML, CGI	Slow Control interface

Table 1
Commercial and public domain software.

Processes taking part in the data acquisition are built according to a well defined template implementing the DAQ state machine. After the initialization phase, a DAQ process enters a main loop, where it executes sequentially its preset list of tasks, while remaining ready to react to external commands. A set of libraries written in "C" provides the interfaces to the Message System, hardware components, TCP/IP sockets, shared memories, etc., thus keeping in the process code only its specific functionalities[15].

The Run Control process, RC, manages the DAQ state machine, ensuring that all processes involved in the DAQ system coherently move between different run states. Fig. 5, shows the RC graphic interface that allows control of different types of run (normal, calibration, pedestal, pulsing), to set run parameters, to monitor the hardware status in case of failure and to start/stop all the procedures used to control data quality or to give feedback to the DAΦNE operators.

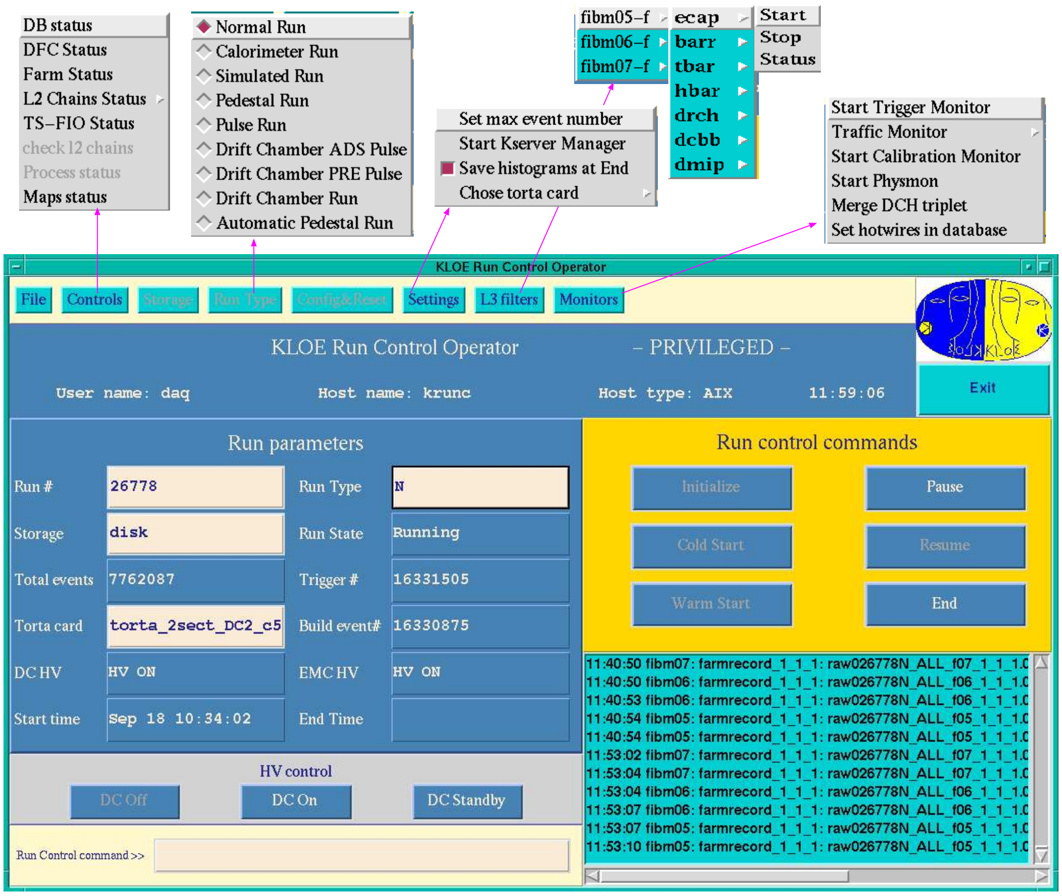

Fig. 5. Run Control graphic interface

5.2 Software libraries

In order to achieve reliable monitoring capabilities, KLOE libraries used by processes involved in the DAQ path are characterized by precise definitions of user interfaces, providing access to single registers of hardware components as well as to objects in the databases or single event data.

Moreover, high performance and low dead time in acquiring data from the readout controllers to the online farm have been achieved by defining the functionalities of the different processes in the way to optimize response when transporting data. As an example, this is achieved by reducing the number of buffer copies in the VME processes or tuning the TCP/IP network parameters as a function of the event size.

Particular care has been taken in defining the software library used to move events data between processes inside a DAQ node. In order to be efficient while maintaining a good monitoring capability the following characteristics have been emphasized:

- Circular buffers implemented as UNIX shared memories;
- Limited memory locking, compared to the time needed to put/extract packets of events;
- Periodic posting of the buffer occupancy, allowing the tuning of every buffer size according to the total available physical memory and the implicit process latencies.

5.3 The message system

Since the online processes are distributed over a large set of nodes in the acquisition network, they have to intercommunicate by means of an efficient messaging system. All processes have to change their state of activity coherently, following local or remote commands. Monitoring of the process activity should not interfere with its cooperation in moving data.

Having these principles in mind, an acquisition node was designed as a network device controllable by the standard Simple Network Management Protocol, SNMP[16].

The SNMP provides a general purpose inter-networking management protocol, allowing access to remote information about network configuration, traffic, faults, accounting and security. This information is made available as a tree of conceptual variables, defined in a Manager Information Base (MIB) [18] using elements of the ASN.1 notation[17]. Privileged accesses are also defined to force changes in remote devices or software applications. Adoption of such a system ensures the availability of well maintained, easy-to-use public domain software, implementing both dedicated daemons and utilities for remote access.

The structure of a typical data acquisition process, and its interaction with the message system, are described in the following. At the startup, the process specifies a set of its more significant variables, in order to subscribe to a 'process table', implemented as a UNIX shared memory. A process message queue is created and its identification is also saved in the table. During the process activity, the value of its variables is periodically updated. A special MIB tree has been defined that maps the structure of the process table in one-indexed and two-indexed lists. A KLOE command server running in any acquisition node is realized as a private SNMP daemon that implements this mapping and uses message queues and process signaling to distribute messages to its managed processes[16].

The command server is able to respond to "set", "get" and "getnext" queries. The "get" and "getnext" commands allow one to scan remotely the process table and their variables, thus producing a sort of generalized shared memory. Remote "set" operations allow the experimenters to modify the value of single variables, a feature extremely useful for debugging purposes; when the "set" function is directed to a particular "last command" variable, it is converted into local commands addressed to one or more processes in the node.

This message system allows the creation of centralized utilities which are able to monitor remote node activity by only invoking the command server cooperation in that node.

Within the SNMP protocol a two level acknowledgment has been implemented [19] to manage remote commands. Moreover, in order to speed up the execution of commands distributed to more processes, parallel command execution and delayed acknowledgment requests are defined. Such a feature is for instance very useful in implementing run control commands whose execution time by single nodes can be as large as a few seconds.

As a performance indicator, the average time to get a remote variable in the KLOE DAQ network is ~ 1.2 ms and the remote command completion time is ~ 4 ms, including the acknowledgment waiting time.The Data Flow Control protocol [19], described in the next section, is implemented by means of SNMP traps.

Examples of utilities running in a remote list of DAQ processes and their corresponding variables are given in Fig. 6.

Graphic tools have been built using the TCL/TK package in order to help full process control. In order to integrate the Message System library, it was only necessary to implement two new TCL commands, 'get variable' and 'send message'.

Getting the list of DAQ processes in a remote node	Getting the list of variables of a process in a remote node
Command: *procs@node*	Command:*varnames trgmon@node*
process: cmdsrv status;ALIVE runstate:INIT	*runnr = 18871*
process: lpcheck status:ALIVE runstate:INIT	*L1 = 4621*
process: receiver status:ALIVE runstate:INIT	*L2 = 3046*
process: build status:ALIVE runstate:INIT	*dead = 5.533*
process: recorder status:ALIVE runstate:INIT	*ratephi = 30.63*
process:trgmon status:ALIVE runstate:INIT	....
....	

Fig. 6. Getting process list and process variables

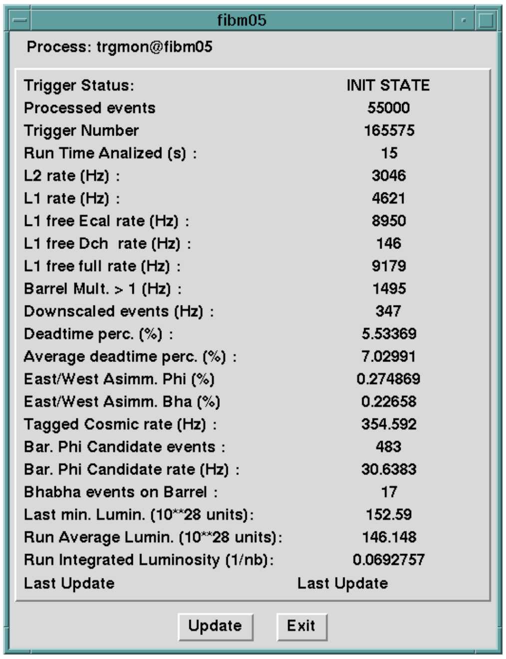

Fig. 7. Trigger monitor

As an example, Fig. 7 gives a graphic presentation of the “varnames” utility. At that time, the rate of the T1 signal was 4.6 kHz, T2 3.0 kHz, the dead time 5.5%, and so on.

5.4 Data moving processes

Processes involved in the ‘DAQ path’, from the ROCKM buffer to the online farm disks are shown in Fig. 8.

The *Collector* initializes the hardware modules in its VME chain and, in case of failure, executes debugging tests to identify the error source. In the main loop, it also reads the ROCKM buffer and builds packets containing 100 sub-events (subpackets). BLT cycles or single-word VME read operations are used,

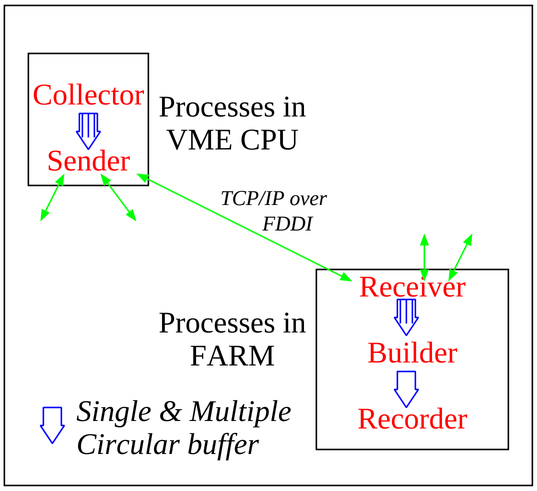

Fig. 8. Data moving processes

according to the buffer occupancy. Subpackets are stored directly in a shared multiple circular buffer (described in more detail below), according to the destination node in the online farm, as defined by the corresponding DFC table.

The *Sender*[20], which runs asynchronously with respect to the *Collector*[21], extracts packets from the circular buffer and sends them to the appropriate destination using TCP/IP connections that remain open throughout the run activity. TCP/IP parameters are tuned in order to reduce dead times due to network latencies. A hardware busy condition will be originated in a VME chain if the *Collector* detects a 'buffer full' status.

The *Receiver* reads data coming from the *Sender*s and loads them in a multiple circular buffer according to the data source. Moreover, it continuously monitors the buffer occupancy and keeps it under control using the following procedure:

- when the occupancy exceeds a given upper threshold, the *Receiver* sends an "almost full" SNMP trap to the DFC
- the DFC eliminates the almost full node from the acquisition chain by changing the validity word of the last flow table. Specifically, the 'actual' trigger number obtained from the Trigger Supervisor is increased by a delta that is dynamically calculated as a function of the trigger rate and the DFC internal latencies. Finally, a new flow table with infinite validity trigger is distributed to the VME CPUs
- when the *Receiver* detects a buffer occupancy below a given threshold, it sends to the DFC an "almost empty" trap signal; this will in turn originate a new DFC procedure to reinsert the node in the DAQ path.

Occupancy thresholds can be dynamically modified during the run according to the real trigger rate. The average time needed to receive and react to a trap is $\sim$1.2 ms, while the average total reaction time of the DFC system is $\sim$ 10 ms, equivalent to the arrival of 100 events at the maximum expected trigger rate of 10 kHz.

The *Builder* puts together subpackets coming from all the chains related to the same trigger number. It then checks their consistency and formats the event structure in YBOS[22] banks, inserting in dedicated banks variables related to run and DAΦNE conditions, as obtained from the Run Control and Slow Control processes.

Events are then loaded in a new circular buffer, from which they are subsequently extracted by the *Recorder* to be written on disk files (presently 1 GB in size) or to be simply discarded during debugging or monitoring runs. Files are finally archived by remote nodes that mount via NFS (Network File System) the online farm disks.

5.5 The multiple circular buffer

The circular buffer is a crucial element of the KLOE DAQ architecture. In order to maintain a high degree of global performance, processes cooperating in moving data have to minimize the required number of buffer copies. On the other hand, monitoring and calibrations tools need access to event data in order to check continuously data quality and calibration constants, and to re-calculate them if needed.

The circular buffer library has been designed and implemented having in mind such high performance requirements. Some relevant features are:

- Only one master puts data in the buffer
- Only one client removes data from the buffer permanently. Other clients can copy events (called 'spy events' in the following) from the buffer in their own local space, to be used as random samples for statistical analysis.
- The master reserves space every time a new packet of events has to be acquired, received or built, and puts the relative data directly into the buffer. Excess space is released when the buffer is validated and becomes available to the clients. A brief memory locking is requested, long enough to modify internal pointers

At present the KLOE circular buffer library is supported on AIX, HP-UX, Linux, LynxOS, OSF1 and Solaris platforms. It can be configured as a single or multi-thread environment using SYSV or POSIX compliant routines, to access shared memories and semaphores.

5.6 Processes in a node of the online farm

Fig. 9 presents the full software configuration of a node in the online farm.

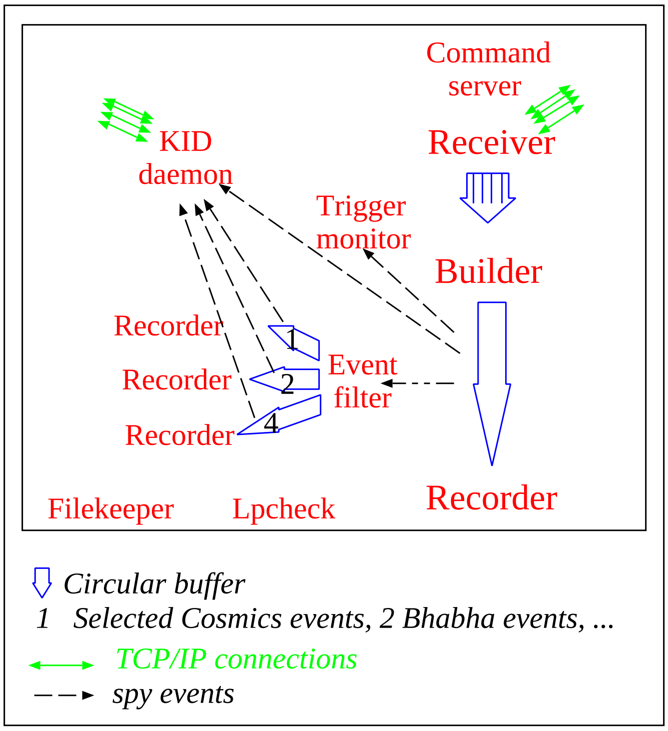

Fig. 9. Processes in a farm node

The *Trigger monitor* uses information collected by the trigger chains to measure trigger rates, acquisition dead times, cosmic and Bhabha event rates, DAΦNE instantaneous luminosity and other relevant parameters.

The *Event Filter* uses information from the whole detector, provided by the Builder, to select different event categories useful for monitoring and calibration tasks. A fast analysis of calorimeter hits and of trigger pattern creates four different online streams: L3BHA, L3COS, L3MIP, L3DCC. The L3BHA buffer collects with high efficiency Bhabha and $\gamma\gamma$ events produced with a polar angle θ between 35 and 145 degrees. The L3COS stream selects candidate cosmic ray events, as recognized from the trigger pattern, while L3MIP uses the energy reconstructed in different calorimeter planes to select golden minimum ionizing particles. The L3DCC stream selects cosmic rays with a minimum number of DCH hits.

The *KID daemon*, fully described in the next section, distributes 'spy events' to local and remote processes.

The *Filekeeper* maintains free space on disk, deleting files in a way as to ensure that data acquisition is never stopped due to lack of space, while at the same time keeping enough data on disk to minimize tape access on the part of the offline reconstruction processes.

The *Lpcheck* monitors the live-status of the processes from the data acquisition point of view, by requesting the change of an internal variable.

Fig. 10 shows the distribution of CPU usage in a node sustaining event rates of 0.8 and 1.6 kHz. Using an SMP-4way server, a single process is only allowed to use up to 25 % of the total CPU power, which is a natural limit to the resources used by processes with monitoring function.

In order to keep under control system dead times that could be originated by lack of resources in the farm, before reaching optimal performance the operating system and NFS parameters were subjected to a process of fine tuning. Methodologies used to obtain quasi real-time performances in the online farm were the use of Unix fixed priorities and pinning in real memory in processes involved in the main data path.

The occupancy of intermediate buffers in *Receiver* and *Builder* is checked whenever DAQ conditions are changed (e.g. new trigger setting, new accelerator parameters, etc.), in order to avoid trigger dead times due to lack of memory in the data moving processes. An example is presented in Fig. 11 showing, as a function of running time, the number of subpackets coming from the 10 DAQ chains that are waiting to be extracted by the *Builder* from the *Receiver* buffer (two top plots). Possible reasons for this waiting state are:

- Not all chains have sent the subpacket related to a given set of consecutive 100 triggers;
- The *Builder* is still completing the analysis of previous subpackets.

The *Builder* buffer is always emptied by the *Recorder*. In order to allow monitoring processes to access built events, the *Recorder* extracts the last 20 packets of events only at the end of each run. In this way, monitoring processes have always events to be analyzed. The bottom plot in Fig. 11 shows the content of the *Builder* buffer, that is always about 20 packets, together with the sum of contents of subpackets in the *Receiver* buffers.

5.7 Monitoring the traffic in the FDDI switch

The traffic in the FDDI switch coming from different chains is also monitored. An example is presented in Fig. 12, showing the traffic originated by the 10 acquisition chains over a 2 hour interval. The start and end of run, as well as the periodical topping up of the DAΦNE beams are clearly visible. The plot, showing a total throughput of 10-13 Mbyte/s, illustrates the big asymmetry in the traffic originated by the different acquisition chains. Still, such an asymmetry was shown not to introduce dead times in the overall DAQ

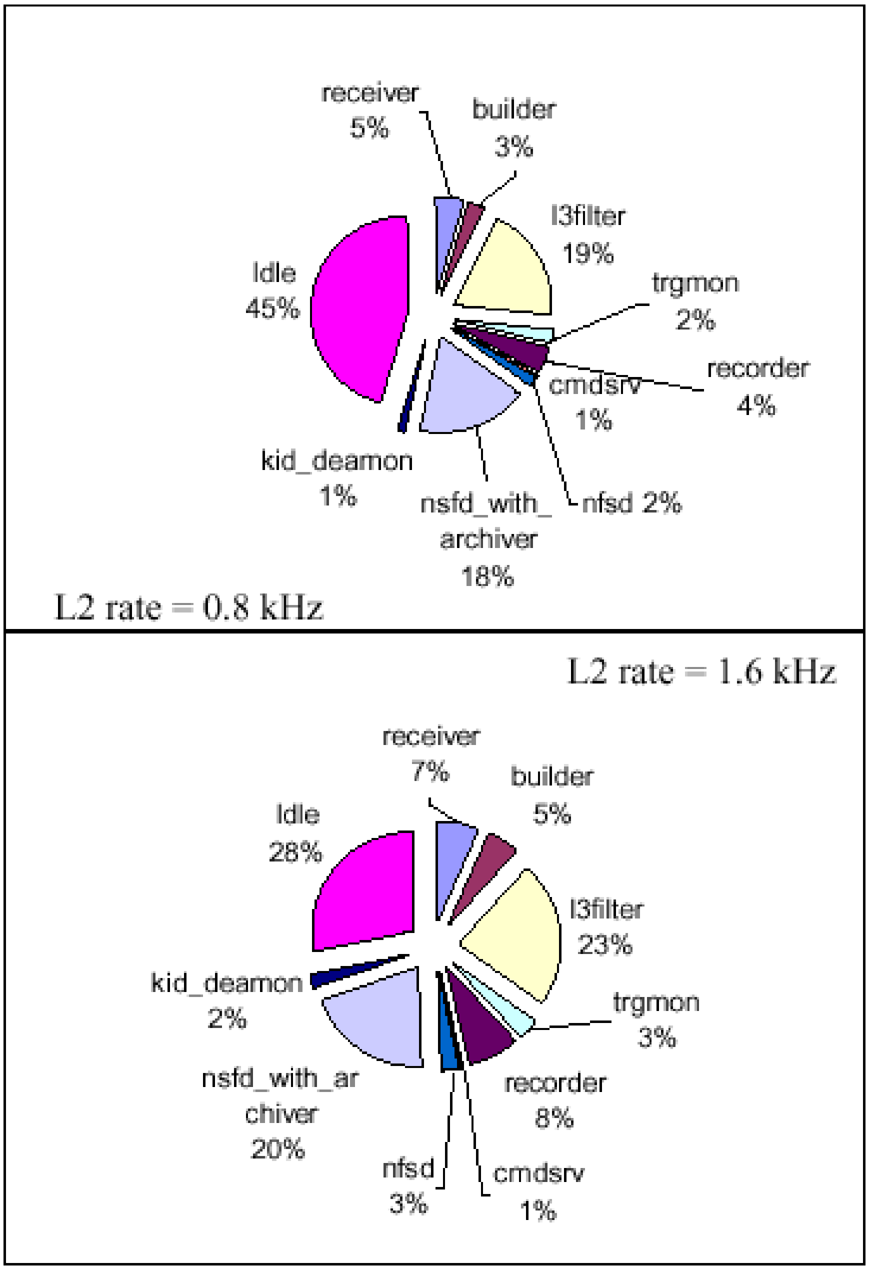

Fig. 10. CPU utilization in a node of the online farm

system.

5.8 Dataflow

The KLOE Integrated Dataflow (KID) [23], is a software package that provides a unique interface to any application needing to access the KLOE event data.

As mentioned earlier, 'spy events' can be peeked at directly from data acquisition buffers using specialized software interfaces. Moreover, KLOE data

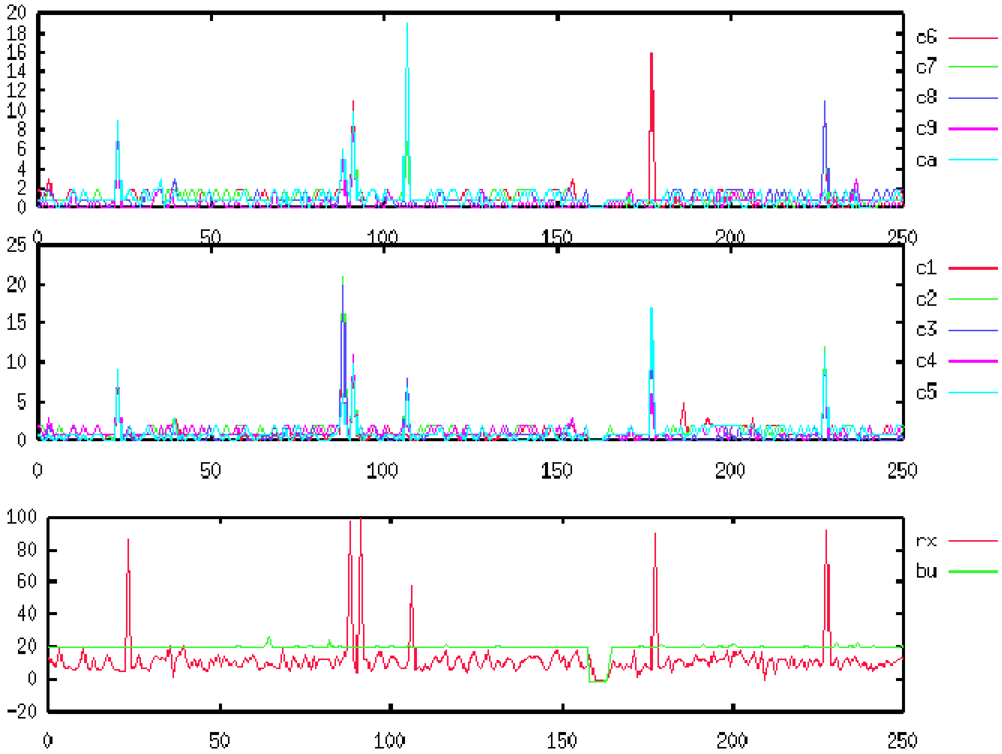

Fig. 11. Occupancy of circular buffers in the online farm versus time (in units of 15 sec)

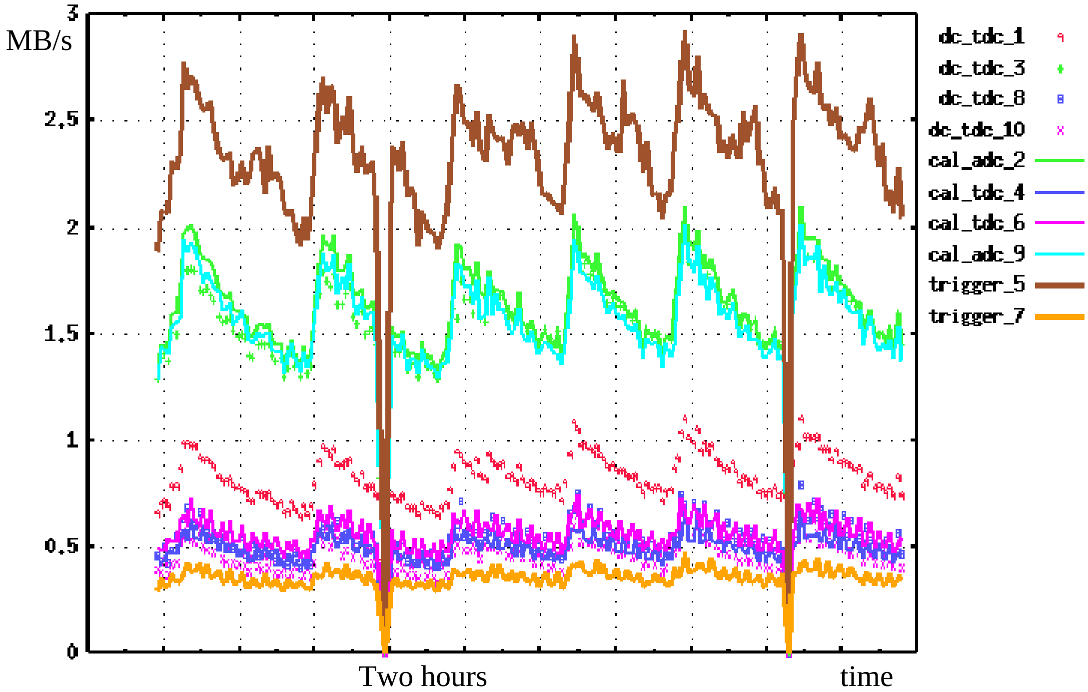

Fig. 12. Traffic in the FDDI switch originated by the DAQ acquisition chains

are stored in millions of large files containing raw, reconstructed or simulated events. All data files are indexed using a Relational Database that keeps track of file parameters and actual file location. Tapes are used for long-term storage and large disk areas are used for short-range data staging.

The KID package allows selection of the data source using simple and easy to remember text strings in the form of URIs, Universal Resource Identifiers. Several data sources can be combined together like circular buffers, or a set of files can be implicitly selected by SQL queries to the Database. This package provides a powerful environment, allowing the use of the same analysis tools (e.g., event display) both in online and offline processes, by simply defining the desired data source: 'spy events' from multiple circular buffers or files.

Examples of URI are:

- *spydaq:ALL*
  to receive raw unfiltered events from all the online farms
- *spydaq:L3BHA?multi=n?wait_run=no*
  to ask for raw events, identified by the online filters as Bhabha events. acquired in all the online farms
- *dbdatarec:run_nr between 19005 and 19011 and stream_code= 'rad'?report=true*
  analyze all files of type *rad* reconstructed from raw files in the given range of run numbers.

KID also optimizes data access by transparently moving files to the appropriate storage media (e.g., from tape to disk), eliminating the need to keep explicit track of the files location.

6 Database system

Event data are written in KLOE using YBOS banks. Two other database systems are used:

- the CERN HEPDB[12] package for calibration constants and detector geometry;
- the DB2[10], a relational commercial database called *DB-online* in the following, for run/detector/accelerator conditions related to any run and for history of event files containing raw, reconstructed and simulated data.

The HEPDB is a database management system tuned to the specific requirement of High Energy Physics experiments that has been heavily used in the HEP community. The KLOE configuration is made by a database server running in one node of the acquisition farm. Input to the database are ZEBRA files written in a default directory that is periodically scanned by the server. Database entries are then written by the server in a set of three files, containing respectively calibrations constants, geometry and run conditions data, that can be read out in parallel for processes needing access to the database. The maximum size of each HEPDB database file is 266 MB. The bigger one (162 MB now) contains all KLOE calibration data until the end of year 2002.

The DB2 server system runs in a dedicated SMP 4-ways server, IBM F50 166 MHz PowerPC, 256 MB memory. The client system is made by an user library and six non-privileged daemon processes running in the same node [25] [26] [27]:

- SQLr_selectd, answers to user process queries;
- SQLr_updated, implements update requests;
- SQLr_loggerd, updates log tables in request to DAQ, reconstruction and simulation processes;
- SQLr_mapsd, manages detector maps;
- recalld, copies files to requested disk areas;
- archived, requests ArchADSM processes, described in the next section, to archive files.

The client system provides a custom multi-thread environment able to open TCP/IP parallel connections between daemons and remote processes performing DAQ work or offline analysis. Data coming from remote processes are converted into local DB2 requests to the database server, via the Structured Query Language (SQL).

The database structure contains 170 tables. One group of tables defines all KLOE computing nodes and storage devices. Filesystems and tape areas (filespaces), used by the archiving processes, are also defined.

A second group of tables defines the bookkeeping database and contains entries related to configuration of the data acquisition runs, versions of the offline reconstruction programs, data cards used in the MonteCarlo simulation and data files produced at the different stages, just to mention the most important ones. This information is never deleted. Other kind of tables define information that changes continouosly during KLOE activity and is related to the position of files in the disk areas.

At the end of year 2002 the size of the DB-online is about 800 MB, almost three times the computer physical memory, and is well managed by the dedicated machine but uses all available system resources. In order to reduce the impact on the CPU, an upgrade of the physical memory at 1 GB, is going on.

A set of tools that can be activated by command line or by GUI interfaces (Fig. 13) allows the simple viewing of the database contents.

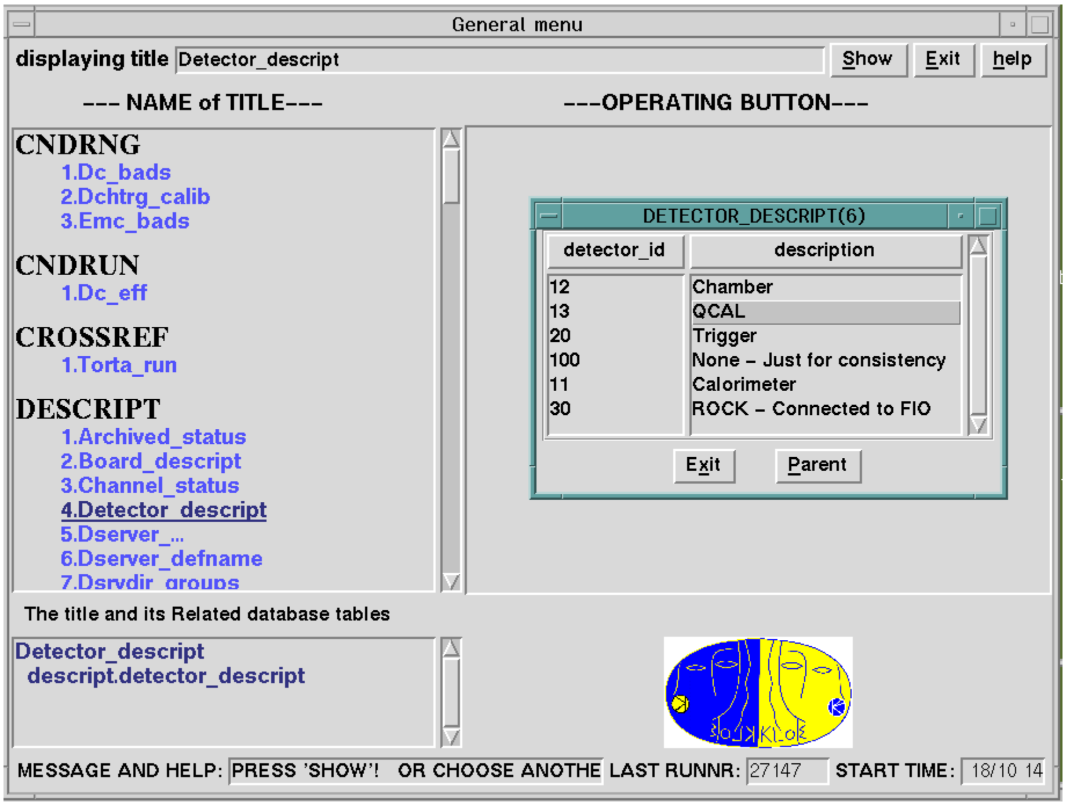

Fig. 13. Graphic interface to the database

7 Archiving system

As already described in section 4.6, the archiving system is built around an IBM 3494 Tape library, two powerful SMP servers managing one half of the available tape drives and the ADSM commercial software package.

The servers provide working disk areas to the offline computers, exporting them via NFS in read/write or readonly mode according with their use. All working areas, included the ADSM ones, are tuned from hardware and software point of view in order to optimize input/output performance and reliability. Working disk areas installed on the online computer farm are NFS mounted in readonly mode by the servers.

A simplified view of the archiving system is given in Fig. 14 where daemon processes that manage automatic archiving and disk space are presented.

The *ArchADSM* daemon executes file archiving requested by the process *archiver* running in the DB-online node. As described in the previous section, the DB-online contains information related to all files that require automatic archiving. The policy used by *archiver*, that looks for "files closed and not yet archived", tries to optimize performances by requiring the archive of groups of files when, for instance, the total size is greater than a default value. Because a different

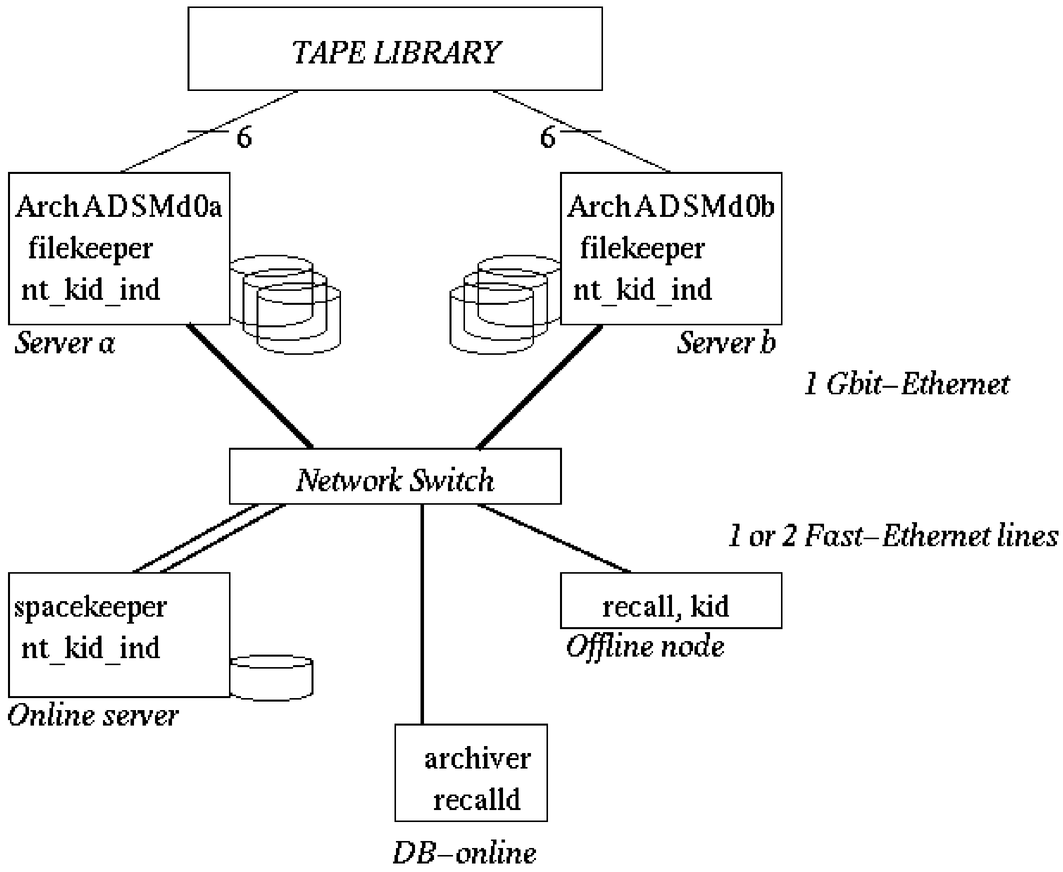

Fig. 14. Simplified view of the archiving system

filespace has been defined in the ADSM configuration for each type of file contents (raw data, reconstructed data, MonteCarlo data), the single *archiver* operation is related to a group of files of the same class.

The *filekeeper* daemon maintains free space in any working area by deleting already archived files when the space used is greater than a default value.

The *spacekeeper* daemon maintains free space in the online farm working areas.

Any application software running in the KLOE environment can use the *recall* command or the *KID* library to ask for a group of files selected according to their definitions in the DB-online. The *recalld* and *nt_kid_ind* processes submits their requests to *ArchADSMd* if files exist only on tape, or provides the actual position on disk.

The ADSM client option is also installed in any KLOE node requiring backup functionalities. Moreover, a limited number of tapes is assigned to users in order to archive their outputs of analysis processes. To do that the original ADSM client is used.

The Fig. 15 presents the performance of the single retrieving and archiving

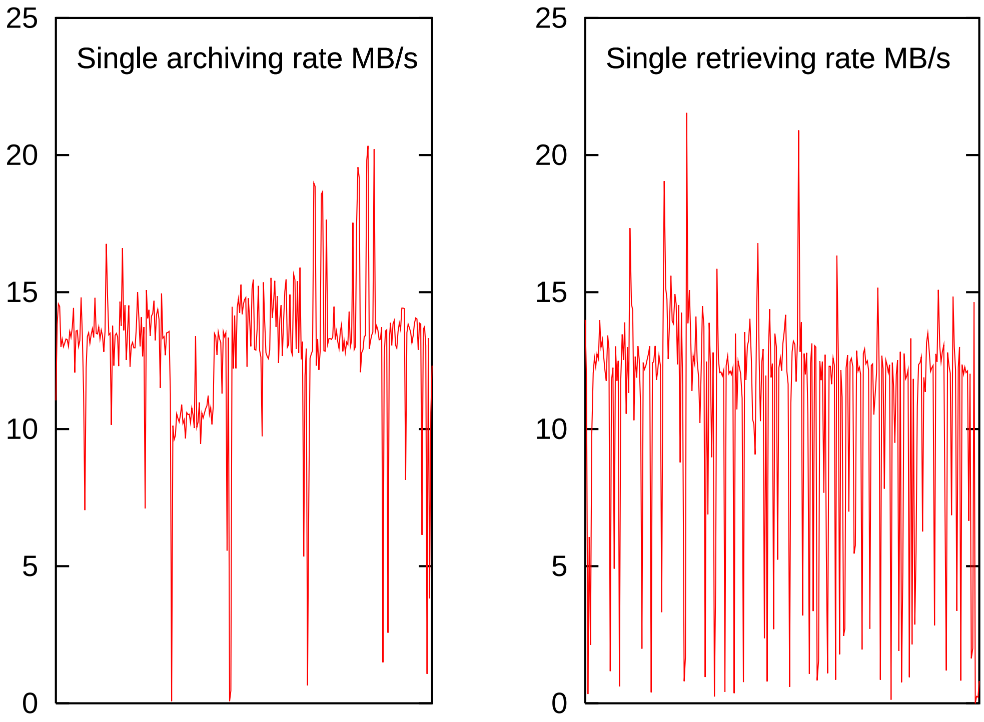

Fig. 15. Retrieving and archiving performance

operations during normal activities requiring tape mounting and dismounting operations.

The observed aggregate server I/O rate is about 40 MB/s per filesystem. Presently, up to 100 client processes use server data without constraints in their activity originating from lack of server resources

A 2 GByte ADSM database size has been successfully tested. KLOE is presently using 300 MByte per server for about 600.000 entries.

8 Online monitoring

Two different kinds of monitor tools have been implemented:

- debugging tools to identify readout modules causing DAQ failures;
- monitoring tools to control detector performances, DAQ rates, goodness of calibration constants and quality of acquired data.

The first set of tools allows the execution of a complete scan of bits and registers on all boards of the single VME readout chain. At any DAQ error, the Run control starts remote executions of the monitoring task in the VME-CPUs, looking for error conditions. Moreover a TCL/TK application is also provided for offline debugging. An example of this monitoring process (Fig. 16) shows the console pages of chain 8, the ROCKM, crate 1 and its ROCKs.

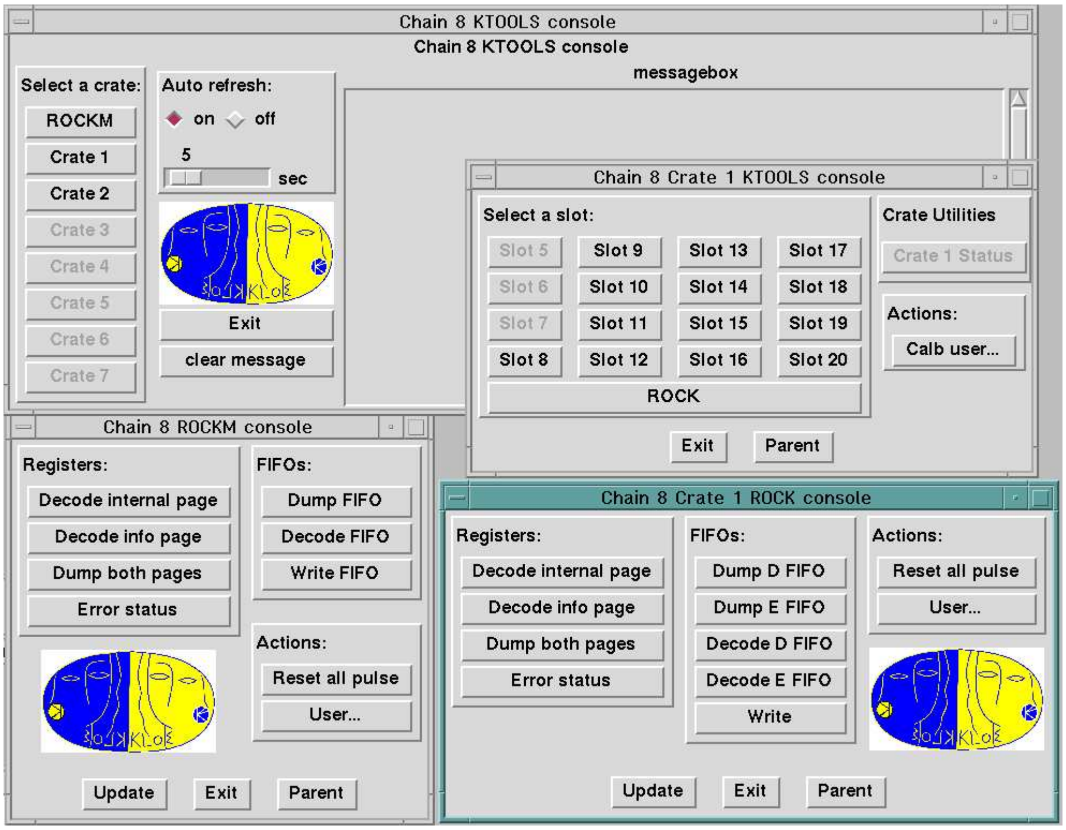

Fig. 16. VME chain debugging tool

Examples of the second kind of monitors are given in the following sections.

8.1 *Kserver and Kbrowser*

Two processes, *Kserver & Kbrowser*, implement a ROOT-based server-client system used for online monitoring[28][29].

The *Kserver* (KLOE server) creates thousands of histograms in ROOT format, organized in a tree structure according to physical or logical partitions (e.g. sub-detector, calorimeter sector, read-out chain), and updates their contents with the incoming raw data events from spy buffers. The *Kserver* receives commands from external applications such as "save histograms to ROOT formatted file" or "delete histograms contents". At the end of the run the histograms are automatically sent to the browser and then reset.

The client *Kbrowser* (KLOEbrowser) is in charge of browsing histogram files and displaying contained objects, specifically histograms, plots, profiles and one or two dimensional graphs.

8.2 The event display

The KLOE Event Display uses KID to access raw or reconstructed data and it is composed of two cooperating processes. One of them reads a single event from the requested data source, reconstructs its components (clusters in the calorimeters, tracks and vertices in the drift chamber) extending the event data with new banks. Alternatively, it uses already reconstructed values, depending on the user request. In both cases the event is written into a shared memory. The second process reads the event from this shared memory and implements the graphic interface.

Fig. 17 shows the basic KLOE Event Display interface. More information, not appearing in the figure, is available in terms of reconstructed track momentum and coordinates, energy and time of the calorimeter clusters and reconstructed decay vertices. This information can be displayed in a separated window just by clicking the appropriate object in the display.

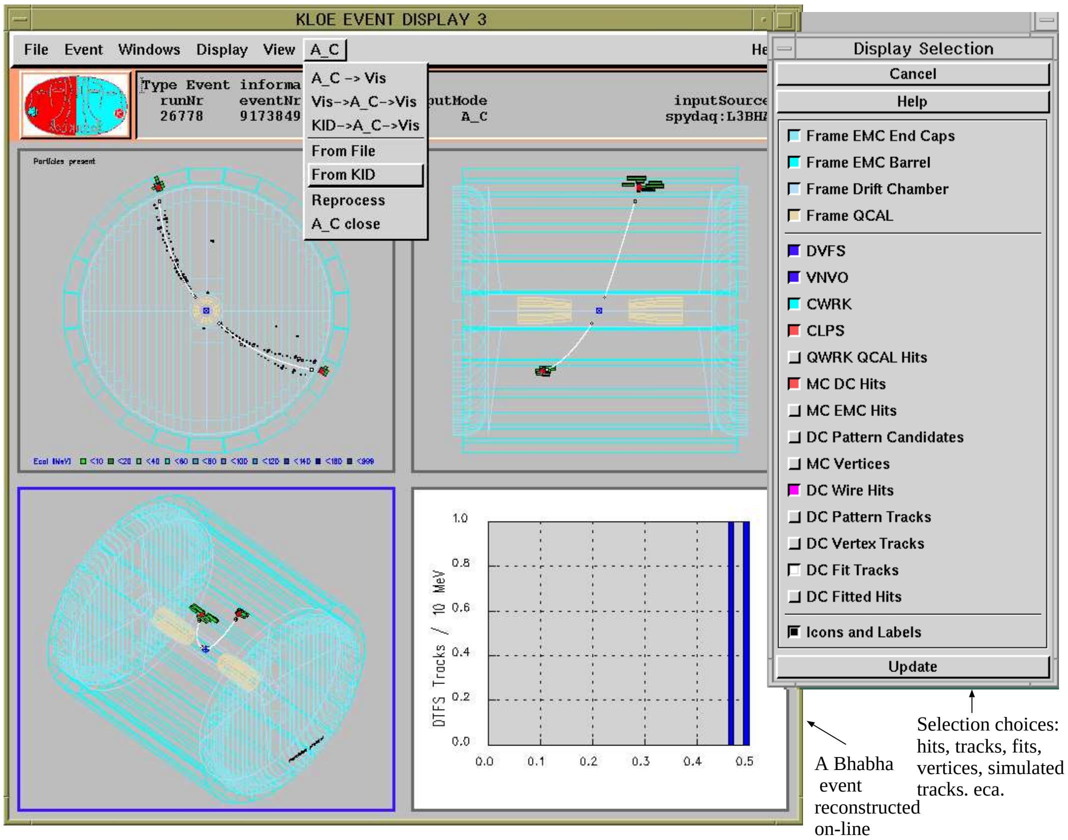

Fig. 17. Event display

8.3 Data quality control

A more sophisticated monitoring of data quality is performed by the *physmon* process, written in the Analysis Control [30] framework. This process allows a complete reconstruction of the event, including trigger segments, calorimeter clusters and tracks.

Events coming from the builder and event filter circular buffers are spied continuously during physics runs, allowing one to monitor the quality of collected data and of calibration stability. To this purpose, *physmon* contains four different subprograms, two for calorimeter monitoring (calmoncos, calmonbha), one for the tracking (trackmon) and one for the trigger (survey).

Each subprogram generates two set of histograms (one for the shifters and for the experts) in HBOOK format, after reaching a given statistics. The Kbrowser is able to transform these HBOOK files in ROOT format, therefore allowing the user to browse through the various files and subdirectories.

Together with the current histograms, a set of reference histograms is also available and they can be displayed in overlap (Fig. 18), to monitor the stability of the detector performance.

For each run the operator controls in this way a relevant set of high level information, such as occupancy on the whole detector, trigger pattern, EMC and DCH resolution, energy and timing scale, reconstruction efficiencies, background levels, average beam positions and beam energy, and so on.

Fig. 18 shows an example of DCH monitoring; blue(red) plots represent the current (reference/normalized) run.

Every few minutes, the averages of many constants (for instance the average beam position) are written on disk files, available to the slow control. These quantities are transmitted as fast feedback also to the DAΦNE machine operators.

8.4 Online Calibration

The online calibration tasks are subdivided into three groups:

(1) Periodic automatic calibrations. These consists of:
   (a) special runs to calculate the pedestals and suppression windows of all ADCs;
   (b) pulsing runs to fire all the preamplifiers to determine any malfunctioning or noisy channels.

   These procedures are executed whenever no beam is present, typically every one or two days.
(2) Specialized long term calibrations. After each extended shutdown, or whenever required, a data taking of cosmics with no beam is carried out for one or two days, until a few millions of "golden" minimum ioniz-

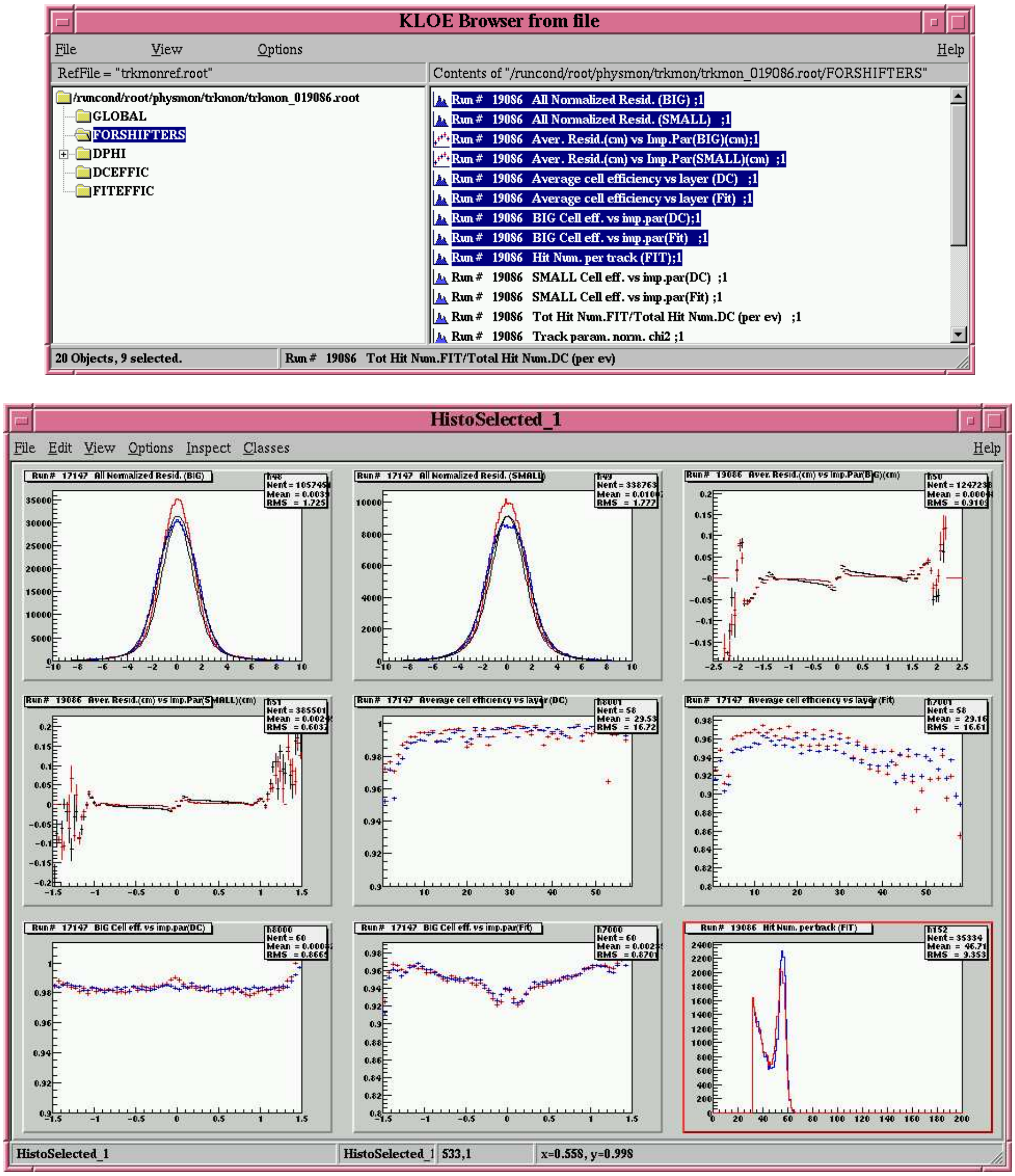

Fig. 18. Histogram browser

ing tracks are acquired. This sample allows the energy cross-calibration of all the calorimeter channels. The same data is also used to calibrate all the timing offsets, T_0's, for both the EMC and DCH detectors.

(3) Run by run determination or monitoring of the calibration constants. This task is the one more integrated with Run Control[32]. For the drift chamber the DCHCHECK procedure, driven run by run by RC, runs over cosmic ray events selected by L3DCC, to verify the stability of the time-to-space relations. Whenever a substantial deviation from the previous parameters is found, the offline calibration procedure is run to recalibrate all parameters and write them in the Calibration Database.

In Fig. 19, the atmospheric pressure is shown as a function of the run number, each black line represents the validity period of a single DCH calibration. The figure shows that changes in the external atmospheric pressure are well detected by the automatic calibrations procedure.

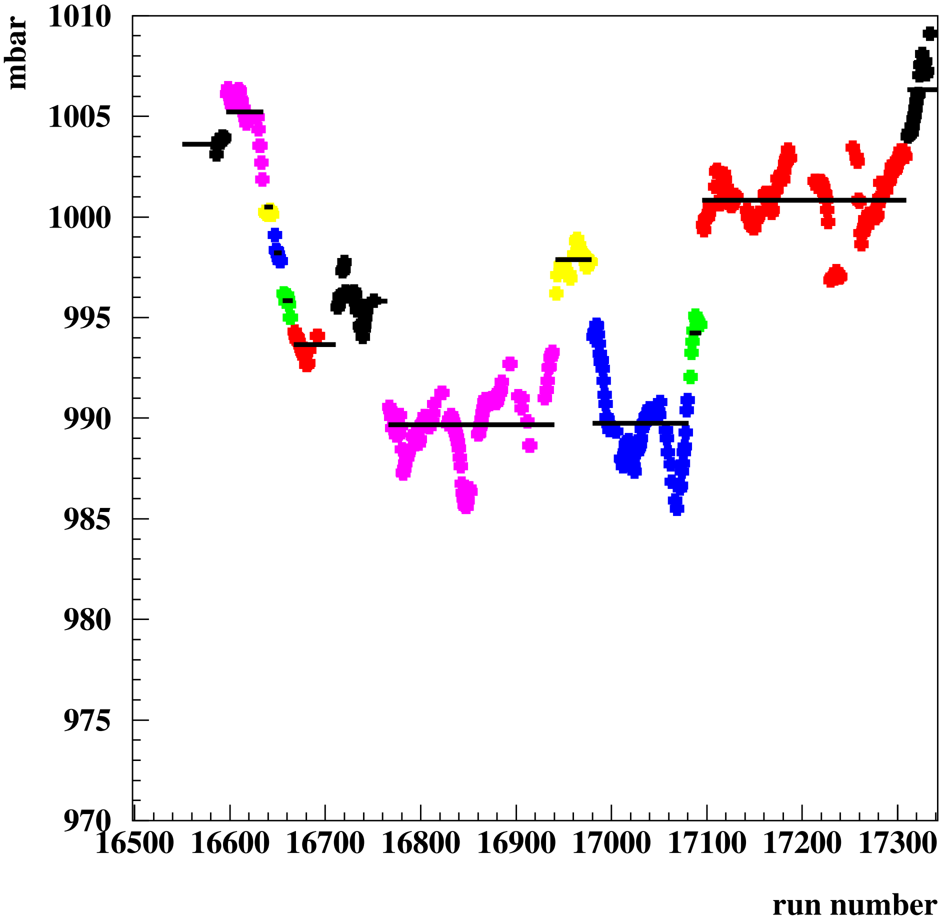

Fig. 19. Atmosferic pressure vs run number and DCH calibration

The EMC calibration is done in a more continuous way. For each run, the *t0gmaker* procedure (controlled by RC) monitors the calorimeter time scale by measuring the machine radiofrequency period, TRF, as determined from reconstructed timing peaks in $e^+e^- \rightarrow \gamma\gamma$ events. With the same procedure, the average T0 due to collisions is also controlled. In addition, every 100 nb^{-1} of integrated luminosity the calibration of energy and timing of all channels is performed using Bhabha and $e^+e^- \rightarrow \gamma\gamma$ events.

In Fig. 20 the dependency of the timing resolution as a function of the run number is shown for Barrel (top) and EndCap (bottom) detectors before and after the online calibration is performed.

The two procedures (Calib_ene, Calib_time) are queued by RC in dedicated batch queues.

At the end of each process, the relevant calibration constants are written in HEPDB and a "done" flag is set on DB-online, enabling in this way the start of the offline reconstruction process for the specific runs. Both the status of the calibration processes and the setting of calibration flags in the database are monitored by the operator via a graphic interface to check that each run is correctly calibrated or to notify that something went wrong.

This procedure was tested to be stable in more than 600 runs. The offline reconstruction procedure is now a slave-process of this calibration task.

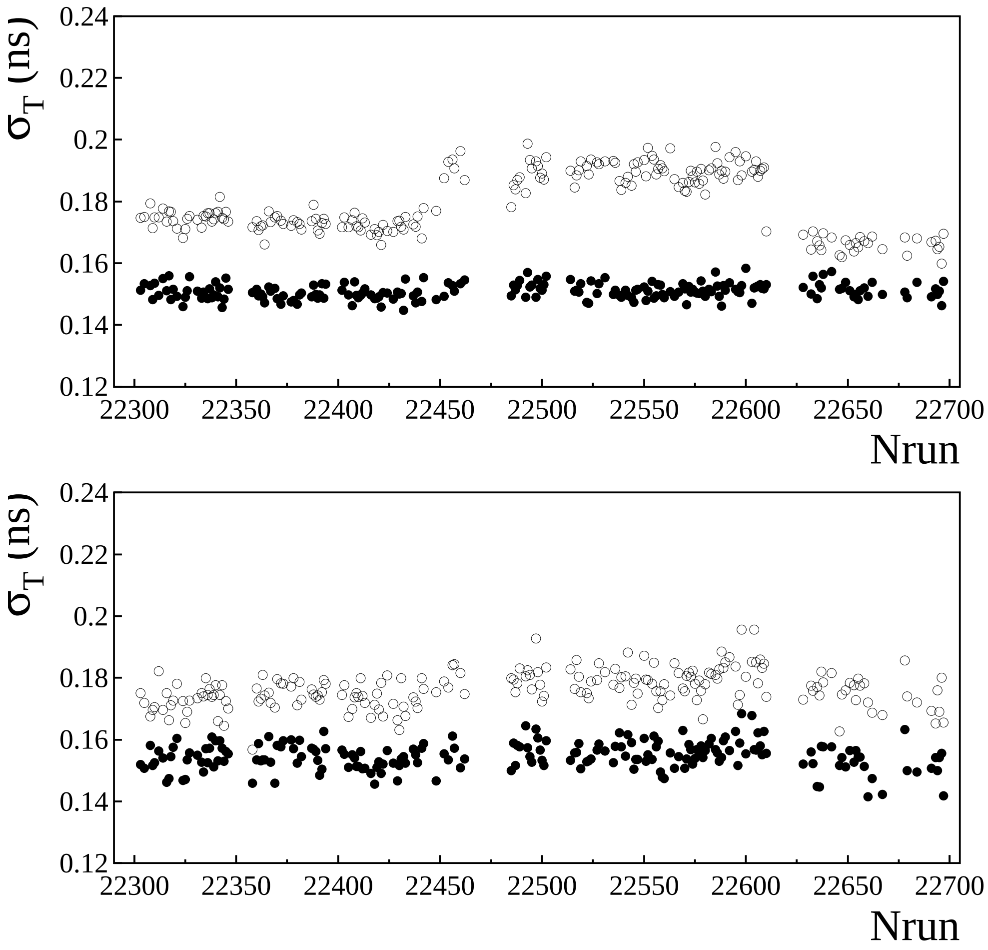

Fig. 20. σ_T stability

9 Slow Control

The KLOE slow control is a hardware-software system responsible for setting detector parameters such as high and low voltage power supplies, thresholds for zero suppression and signal discrimination, gas flow and composition, drift chamber noisy channels and currents, etc. It also monitors quantities such as trigger rates, background levels, gas system, magnet control system, accelerator constants, etc., and creates alarms and warning messages to alert about operation problems.

The core of the slow control system is a VME (Fig. 21) crate housing

- seven CAENet [31] serial interface boards, connected to the FEE crates;
- a CAENet module connected to a multichannel nanoammeter, in order to monitor currents in the inner part of the chamber;
- a scaler for the monitoring of trigger and background rates;
- an I/O register;
- a CPU with Ethernet interface.

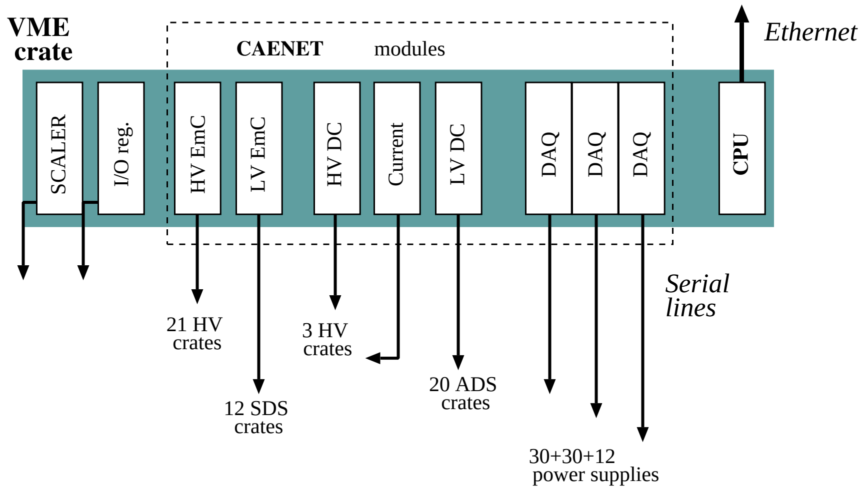

Fig. 21. A schematic view of the KLOE Slow Control VME crate

Processes running in the CPU are responsible for interfacing the slow control with the run control system, for setting hardware parameters at the run initialization time, and for periodic reading out of interesting variables. All data collected for this monitoring activity are handled and stored by a dedicated server. This server implements the user interface, generates and handles alarms and warnings, and communicates with the online software for the logging of the relevant detector parameters, both in the DB-online and, a selected set, in the event data.

The interface with the run control is realized via a dedicated DAQ process running in the slow control VME CPU. This process maps the run state machine and uses the same Message System to exchange data with the rest of the DAQ system. A short string received at the run initialization time provides the type of run, power supplies values, and noisy channels in the drift chamber that need to be disabled. This process distributes related initialization functions to all FEE modules and controls their completion.

The user interface is entirely written in HTML language, so that any Web browser can be used to access the system, both for the monitoring of the detector and the setting of all slow control parameters (see Fig. 22). The slow control server runs the Web server, and all the monitoring and control functions, as well as the alarm generation and handling, are implemented by Common Gateway Interface (CGI) programs. Special warnings, e.g. those related to the gas system, are managed by a watchdog process that automatically sends a GSM short message to a list of on-call expert mobile phones.

Systems not integrated in the FEE VME hardware, but with internal control/monitoring facilities such as the control of the DC gas, the magnet control and the accelerator control systems, are remotely monitored by the Slow Control server. Monitoring processes listen continuously for TCP/IP connections on a dedicated port and get data from remote clients. History files are periodically updated with the more relevant parameters, in a disk area accessible to all the KLOE network. Processes dedicated to the control of data quality or the monitoring of the trigger system also write data in history files.The *Presenter* utility, based in the ROOT framework, (see Fig. 23) is used to monitor sets of variables both during the run and all along KLOE's history.

10 Present DAQ performance

The KLOE DAQ has fulfilled the design requirements. The maximum event rate acquired during the last runs in year 2002 was about 5 kHz but an acquisition rate of 10 kHz has been tested by lowering the trigger thresholds. The number of VME chains is considered to be sufficient to manage event rates bigger than 10 kHz, equivalent to a total throughput of 35 MByte/s with an average event size of about 3.5 kByte. The present number of servers in the online farm is:

- One server for Run Control and histogram server;
- One server for the online database;
- Three servers for building, filtering, recording, distributing 'spy events' to monitor processes and serving NFS disks to the archiving system and to the offline reconstruction farm;
- Two servers for online calibrations and data quality control;

The three servers are sufficient to maintain an acquisition rate of 10 kHz. The tuning of performances allows one to assure major resources dedicated to the main path of data acquisition, while more monitoring processes are

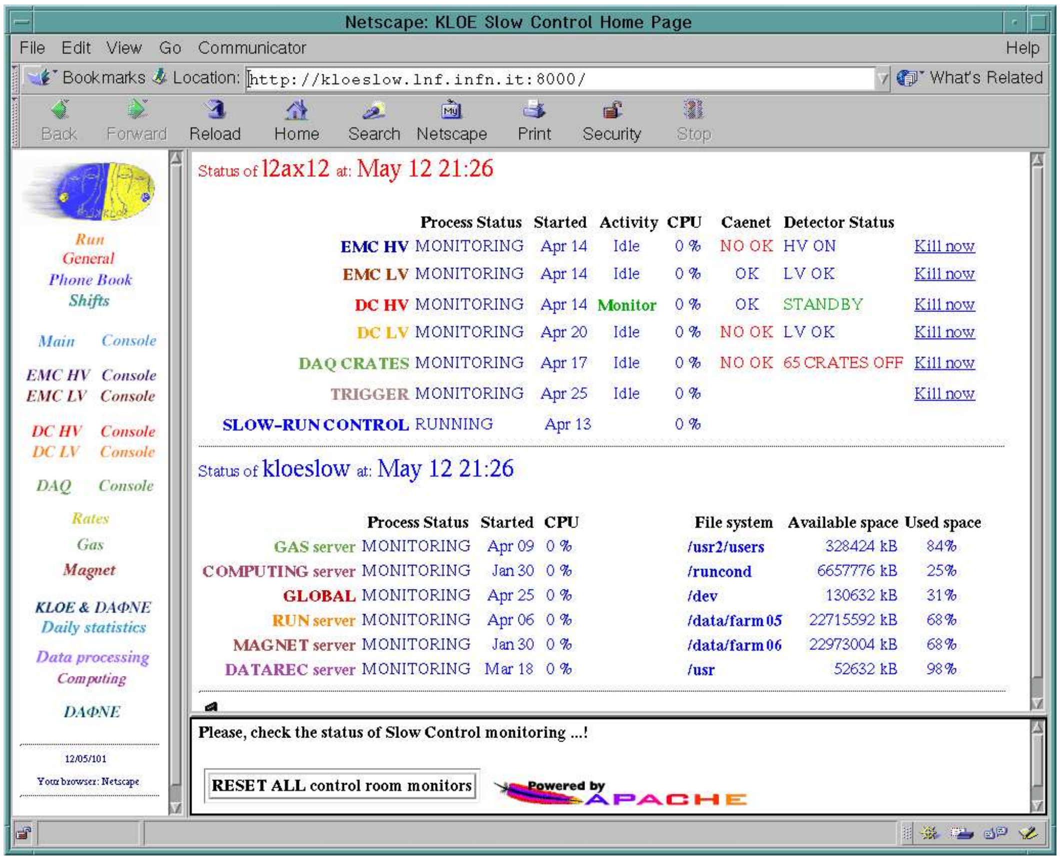

Fig. 22. Slow control HTML interface

periodically included. Up to 7 servers are presently available to increase the computing power of the online acquisition farm, allowing for more monitoring or filtering processes to be running in the online farm.

As a reference, Fig. 22 shows the DAΦNE luminosity achieved during the KLOE run period in September 2002. The collider was working with 48 bunches, and the delivered peak luminosity was $\sim 7 \times 10^{31}\ cm^{-2}\ s^{-1}$. The average data acquisition rate, L2 rate, was about 3.5 kHz allowing the collection of 4.8 pb^{-1}/day.

During the last months of activity a new process was included in the online farm, the t3filter. The main reason for that was to increase the trigger efficiency on $e^+e^- \to \mu\mu$ and $e^+e^- \to \gamma\gamma$ events. This was achieved by disabling the cosmics trigger and by implementing an efficient software selection of cosmics events. A very efficient tracking algoritm was implemented in C code and after fine retuning the DAQ system was able to maintain the previous performances and low dead-time.

With t3filter in the online farm main data path, the event rate of acquired event was about 1.6 KHz (including 600 Hz of downscaled cosmics and 250

Hz of downscaled bhabha events).

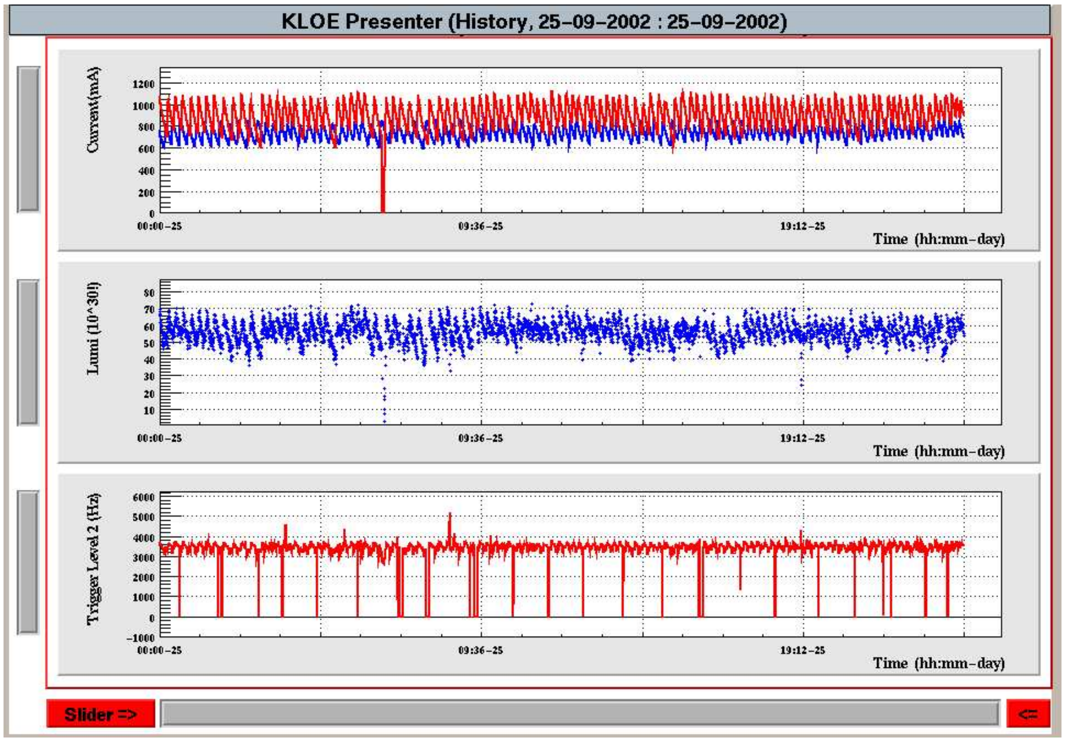

Fig. 23. Example of Presenter monitoring tool

At the end of year 2002 KLOE collected an integrated 488 pb^{-1}, more than 90% of the luminosity delivered by DAΦNE, about 10^{10} events are archived in 105 TB (raw data) and 78 TB (reconstructed data). The rest of the initial library capacity is occupied by simulated events, DSTs, user backups and archives of analysis. To cope with the limitation on archiving space, the library capacity is being upgraded to 300 TB by using new tape controllers able to write 60 GB uncompressed in the installed cartridges.

11 Conclusions

The core of the KLOE data acquisition has been designed anew, having in mind high performance, computer platform independence and distributed monitoring capability.

The use of custom buses and a fine tuning of network and operating system parameters guarantees low acquisition dead time, even when the available computing resources are partially dedicated to activities like event selection, data quality control and detector calibration.

The KLOE message system, based on standard UNIX mechanisms and the SNMP protocol, has proved to be a robust and reliable system for data acquisition and process control.

Commercial software for archiving and database has been successfully integrated with the custom KLOE dataflow software, which allows a transparent access to event files independently of their physical location.

The slow control system is fully integrated in the DAQ framework; this helps to keep track of some relevant information in the event data and more generally in the online database.

The online feedback given by the KLOE DAQ to the DAΦNE team, related in particular to luminosity evaluation, position of the interaction region and backgrounds, has been a very useful tool for the DAΦNE tuning.

Acknowledgements

We want to warmly thank G.F. Fortugno for the continuous support in configuring, improving and maintaining the KLOE computing and network services.

We also thank the computing and electronic services of the LNF:

- M. Pistoni for the support in integrating the KLOE network in the LNF site
- A. Balla, G. Corradi and M. Carletti for the design and maintenance of control systems
- M. Santoni for the support in the installation of the DAQ infrastructure
- M. Beretta for the development of the drift chamber ADC board.

We also acknowledge the staff of the electronic laboratory of INFN section of Naples for their support and in particular P. Parascandolo and A. Anastasio for their great contribution to the development of the AUXbus project.

We have to thank the electronic service of the Sezione INFN and Physics Departments of Roma "La Sapienza":

- F. Cidronelli, S. Di Marco, E. Gennari and A. Rossi for the design, assembling and maintenance of the read out electronics

We thank also the electronic services of the Sezione INFN of Istituto Superiore di Sanitá and Roma III:

- R. Lomoro for the support in testing read out VME boards;
- V. Dante and A. Migliore (Helwelt Packard) for the contribution at early stage of DAQ design using a custom single board computer.

Work partially supported by German Ministry of Education and Research (BMBF) under contracts (06 KA 654 TP3), (06 KA 564 TP2), (06 KA 860),

(06 KA 957);
Work partially supported by Graduiertenkolleg 'Elementarteilchenphysik an Beschleunigern';
Deutsche Forschungsgemeinschaft;
Work partially supported by EURODAPHNE network, contract No. FMRX-CT98-0169;
Work partially supported by INTAS contracts No. (96-624) and (99-37);
Work partially supported by TARI contract HPRI-CT-1999-00088.

References

KLOE notes are available at **http://www.lnf.infn.it/kloe/pub/**

[1] The DAΦNEProject Team, *Proposal for a ϕ-factory*, LNF-90/031 1990;
C. Milardi *et al.*, *Status of DAΦNE*, Proceedings of the Workshop on Physics and Detectors for DAFNE, Frascati, Italy (1999)

[2] KLOE Collaboration, *The KLOE Detector Technical Proposal*, LNF-93/002, (1993);
KLOE Collaboration, *The KLOE Data Acquisition System, addendum to the Technical Proposal*, LNF-95/014, 1995;

[3] KLOE Collaboration, *The KLOE electromagnetic calorimeter*, Nucl. Inst. Meth. A 482, 363-385 (2002).

[4] http://www.ces.ch/

[5] KLOE Collaboration, *The Tracking Detector of the KLOE Experiment*, Nucl. Inst. Meth. A 488, 1-23 (2002).

[6] KLOE Collaboration, *The QCAL tile calorimeter of KLOE*, Nucl. Inst. Meth. A 483, 649-659 (2002).

[7] KLOE Collaboration, *The KLOE trigger system od the KLOE experiment*, Nucl. Inst. Meth. A 492, 134-146 (2002).

[8] A. Aloisio *et al.*, *Level-1 DAQ system for the KLOE experiment*, CHEP'95, Rio de Janeiro, 1995

[9] DB2 Product Family, http://www-4.ibm.com/software/data/db2

[10] Storage Solutions http://www.storage.ibm.com/

[11] HepDB, a High Energy Physics Database Management package, http://wwwinfo.cern.ch/asdoc/hepdb/

[12] ROOT, a Object-Oriented interactive data analysis tool, CERN Main Page http://root.cern.ch/

[13] OnX, a package to handle GUI with XML, Main WEB page, http://www.lal.in2p3.fr/SI/OnX

[14] M.L. Ferrer *et al.*, *Interprocess and interprocessor data flow in the KLOE data acquis ition system*, CHEP 95, Rio de Janeiro, Brasil (1995)

[15] E. Pasqualucci, A. Doria, M.L. Ferrer, W. Grandegger, *Data Flow Control and Management in the KLOE Data Acquisition*, DAQ96, Osaka (1996)

[16] ASN.1 information site, http://asn1.elibel.tm.fr/en/

[17] AA. VV., *The Simple Web - The source for SNMP and Internet management info* http://www.simpleweb.org

[18] E. Pasqualucci *et al.*, *Process and Data Flow Control in KLOE*, CHEP'2000, Padova Italy (2000)

[19] E. Pasqualucci *et al.*, *The Online Farm of the KLOE Experiment*, CHEP 98, Chicago (1998)

[20] P. Branchini *et al.*, *Front-end DAQ for the KLOE Experiment*, CHEP 98, Chicago (1998)

[21] http://www-cdf.fnal.gov/offline/ybos/ybos.html, Fermilab CDF library

[22] I. Sfiligoi *et al.*, *The KLOE Integrated Dataflow (KID)*, CHEP'2001, Beijing, September 2001

[23] E. Pasqualucci *et al.*, *Process and Data Flow Control in KLOE*, Comput.Phys.Commun.140, 139-145, 2001

[24] I. Sfiligoi, *KLOE Database - The Bookeeping and Computer System Catalog* KLOE note 184 (2003)

[25] I. Sfiligoi *et al.*, *Data Handling in KLOE*, CHEP'2000, Padova Italy (2000)

[26] A. Doria *et al.*, *Online Monitoring System at KLOE*, CHEP 2000, Padova Italy (2000)

[27] A. Doria *et al.*, *A ROOT Based Monitoring System for the KLOE Experiment*, AIHENP 99, Crete (1999)

[28] http://www-cdf.fnal.gov/offline/a_c/a_c.html, Fermilab CDF library

[29] http://www.caen.it/

[30] E. Pasqualucci *et al.*, *The KLOE Online Calibration System*, CHEP 2001, Beijing (2001)

[31] G. Mazzitelli, F. Murtas, P. Valente, *The KLOE/DAPHNE Status Logging, Analysis and Database System*, ICALEPCS 2001

[32] AIHENP 99, Crete (1999)

[33] http://www-cdf.fnal.gov/offline/a_c/a_c.html, Fermilab CDF library

[34] http://www.caen.it/

[35] E. Pasqualucci *et al.*, *The KLOE Online Calibration System*, CHEP 2001, Beijing (2001)

[36] G. Mazzitelli, F. Murtas, P. Valente, *The KLOE/DAPHNE Status Logging, Analysis and Database System*, ICALEPCS 2001

[37] M. Antonelli *et al.*, *The ADCs and TDCs for the KLOE electromagnetic calorimeter* Nucl. Inst. and Meth. in Physics Research A 409 (1998) 675-678