Evolution and Present Status of VEPP-5 Control System

D. Bolkhovityanov*, I. Pivovarov, O. Tokarev, BINP, Novosibirsk, Russia

Abstract

Initially the VEPP-5 control system consisted of a number of unconnected subsystems, each one having a different design. Lots of fieldbuses were used, including homemade ones. In the last two years the control system was almost completely re-engineered. Upper level consists of Linux PCs, which are connected via Ethernet. Low level consists of CAMAC hardware. The main fieldbus now is Ethernet, and CANBUS is begin being used. The control system software is network-transparent, and it complies the "standard model"[1]. In 2001 we began using Linux-based CAMAC controllers, which made the control system software much more unified and easy to maintain.

1 EVOLUTION OF CONTROL SYSTEM SOFTWARE

The first prototype of VEPP-5 centralized control system had appeared in 1994. That time all control logic was concentrated in crate controllers, and PC was used only as a terminal and file-server.

1.1 First generation: monolithic software

In 1995 we began using new crate controller based on Inmos T805 transputer[2]. Inmos transputer use special "links" for communication between them, and the same link was used to communicate with host computer. Due to the nature of transputer link, only one program could be the "owner" of the link on host side.

The first generation of Linux control software played two roles: it interacted with CAMAC hardware and presented graphical user interface. I.e., control programs were monolithic. This approach was satisfactory for first time, and several of those programs are used till now. But as the number of control programs increased, it soon became obvious that this approach wouldn't allow to build a single, unified control system.

1.2 Second generation: standard model

The three-level "standard model"[1] was widely accepted to be the most appropriate for large control systems, so we made an obvious decision: to develop a standard-model-compliant system.

(An obvious question: why didn't we use EPICS? First, EPICS wasn't free that time, and BINP simply couldn't afford buying EPICS and paying for training the personnel. Second, Linux support was added to EPICS much later.)

New control system was initially designed for VEPP-5 damping ring (which is a separate facility), and proved to be successful[3]. After thorough testing it began to be used for VEPP-5 preinjector linac control.

This system was named CX ("Control for Unix) and consists of four major components:

- Drivers for CAMAC blocks, which run in the controllers (lower layer).
- CX server — the only program which talks to controllers.
- Application programs, which know the "physics" (upper layer).
- "Management console" — a special program which controls operation of CX server.

While CX did the job it was designed for, its first version had several limits "by design":

- The control system operates with 32-bit channels, which allow to represent most of the CAMAC hardware. Any "large" channels should be served separately. This enabled to do some optimizations when several applications deal with same channels.
- The mode of operation is periodic, with 1Hz frequency.
- CX server supported only Inmos-based CAMAC controllers, but we got many other hardware: different CAMAC controllers, CANBUS devices etc.

1.3 Third generation: as little limits as possible

In 2001 CX server began to be redesigned.

On the first stage the built-in support of Inmos controllers was removed in favor of modular driver architecture. For each device which CX has to serve a driver is loaded. Drivers are built as separate modules (.so-files) and are loaded into server's address space dynamically. So, adding a totally new driver doesn't require recompilation of the system. In the near future we expect to add an ability to change hardware "on the fly" without server restart.

Two more trivial improvement were made: 1) instead of 1Hz the server can run with any clock period; 2) any number of servers can be run on a single computer, each with its own hardware and with different time cycle.

On the second stage support for "large channels" was added. CX server in this case works as a "pipe", simply transferring the data between drivers and applications. However, it implements simple caching. Such channels can have arbitrary size, and we plan to move Ethernet video cameras support(see Section 4) to CX.

* bolkhov@inp.nsk.su

On the third stage an incompatible change in server↔applications protocol is planned: 32-bit and large channels will get unified access interface.

2 CONTROL ROOM COMPUTERS

2.1 OS

From the beginning of VEPP-5 development it was obvious that Windows shouldn't be used in the control system, only Unix. Initially it was BSDI, which was later changed to Linux. Linux becomes more and more popular, and now we can use it in CAMAC controllers too[7]. Having unified environment throughout the whole system significantly eases sysadmin's, programmer's and user's life.

Motif was used for building user interface from the early days of the control system. Motif has proven to be a very good toolkit which still stands good against modern counterparts, but it becomes old and almost unsupported. Unfortunately modern toolkits (like Qt and Gtk) aren't stable and change too often. So we began creating an "abstract toolkit layer", which allows to build control system's applications with either Motif or Qt (Gtk support is planned in the future). First tests show this approach to be successful.

2.2 Computer's roles

On the early stages of VEPP-5 building two "typical" PCs were enough for control system: one for linac and one for damping ring. As the facility grew, the number of control room computers increased. Currently it includes 5 PIII-800 PCs. Computers are connected with 100Mbit Ethernet into a "loosely-tied cluster": they share filesystems via NFS, allow remote program execution via SSH and can use each other's displays via X11.

We tried to combine two roles in each PC: hardware control and operator's interface. But PCs have very limited number of PCI slots — max. 6. One slot is required for Ethernet card, one for SCSI controller, one or two are used by videocards. So, we have only *two* PCI slots for hardware control. And any significant change in hardware requires "jugglery" with available slots.

So we came to a conclusion that a better way would be to separate two roles: one PC does only hardware control (having 4 PCI slots for this), and another one is responsible for operator's interface. Video-related hardware (see Section 4) should also be installed into operator's PC in order to reduce network traffic.

2.3 Multihead Display

Control system requires much display space. Since 2000 it was possible to have multiple monitors on each Linux PC. We began using this feature almost immediately. So, currently VEPP-5 operator places include 4 4-headed PCs — two for linac and two for damping ring[4]. Each PC contains one AGP dualhead card (Matrox G450AGP) and one PCI (G450PCI).

Since we plan to separate hardware control PCs from operator ones, each operator PC would have to drive 8 monitors. Matrox had announced G450X4 PCI cards[5], which will allow to use only 2 PCI slots for display subsystem.

3 ETHERNET

Since the beginning and up to recent time VEPP-5 control system used BINP-designed "Odrenok" intellectual crate controllers ("Odrenok" has 24-bit ICL-1900 instruction set). In 1997 these controllers became Ethernet-enabled[6]. All these controllers are connected into "controller network", separate from the VEPP-5 local network, with only one dedicated PC having access to controllers.

Unfortunately, when designing the networking protocol for these controllers, a decision was made to use private, non-standard Ethernet protocol. This was justified that time, since it made transition of "Odrenok" OS to Ethernet much simpler. But it turned out to be a constant source of problems during exploitation of the system.

- Non-standard protocol requires a kernel support. So, a dedicated kernel module had to be written and maintained. But kernel implementation of a network protocol is a highly complicated task, prone to hard-to-find errors.
- Linux kernel interface changes very often, and every major OS upgrade required not only recompilation and reinstallation of the module, but also updating it.
- The kernel module didn't support all of the Unix I/O mechanisms (such as asynchronous I/O notification), so some of the standard Unix I/O techniques couldn't be used.

So, in 2000 a decision was made to switch from private protocol to UDP. Besides elimination of problems mentioned above, UDP gave us some bonus:

- In case of network problems standard diagnostic tools can be used.
- IP is a routable protocol, so with little system administrating efforts not only a dedicated PC, but any of the control system computers can have access to the controller network.

Currently "Odrenok"-related part of control system slowly migrates from private protocol to UDP.

"Odrenok" controllers themselves are rather old, and new controller has been designed by BINP electronics department[7]. It uses 100Mbit Ethernet for communication with host computers.

4 VIDEO

We use video cameras for beam diagnostics[8]. Three generations of these systems were built.

In the first generation the camera was connected to CAMAC block in a crate with dumb controller, which, in turn, was connected to the host PC via ISA card. This worked, but connection was too slow (about 0.2Hz) and reading 0.5M frame via ISA controller word-by-word significantly stressed the system.

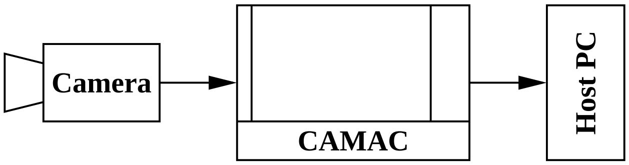

Figure 1: 1st generation video system

To get better performance in the second generation we used widely-available PCI frame-grabber cards. Such a card takes video from a camera and is able to either put it into PC's memory for processing or to blit it directly to the video card's framebuffer. We use Bt-878-based products, which have excellent support in Linux. This approach gives a very high performance (especially in the case of direct frame-grabber→videocard transfer), but has a serious disadvantage: each camera requires a PCI slot in the PC.

In addition, Bt-878 interface, while being very fast, has one more serious problem: it is synchronized with 50Hz frequency, instead of VEPP-5 operation frequency. So, we left Bt-878 for "human eye" diagnostics only.

During design of the third generation we took into account the experience of its predecessors.

Video transport media is 100Mbit Ethernet. Each camera contains: a CCD matrix, a simple Ethernet controller and an Altera chip which implements both CCD reading and communication with a host.

When designing the camera↔host protocol, there was a desire to use some custom protocol, which was very easy to implement in Altera. But we have learned the lesson of "Odrenok" controllers and decided to use UDP from the beginning. This eliminated the need to create a driver for Linux kernel and allows to use new camera with any other networking OS.

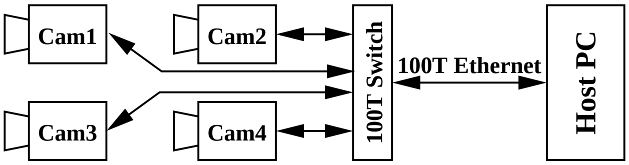

Figure 2: 3rd generation video system

Since cameras create large traffic and have no "IP stack", just a support for a subset of UDP, they work in a separate Ethernet segment and are placed behind the switch. This isolates cameras from each other and minimizes collisions, effectively making camera↔host connections point-to-point.

Currently both 2nd and 3rd generation systems are used.

5 CONCLUSION

VEPP-5 team had managed to support the control system without dramatic changes. The key features which made this possible are:

- Use of standard hardware — PCs on the high level, CAMAC and CANBUS at the lower level.
- Use of Ethernet for both connecting the control system's computers and as a fieldbus.
- Use of Linux.
- Application of standard-model-compliant control system.
- We understood in time that usage of private network protocols instead of standard ones gives us more problems than benefits.

We have learned a lesson: if some limits, looking reasonable that time, are imposed on the control system, sooner or later these limits will hamper the progress. Usually this happens much earlier than expected.

6 REFERENCES

[1] A. Goetz et al., "Experience with a standard model'91 based control system at the ESRF", Proc. ICALEPCS 93 (Berlin). NIM 352A (1994) 22-27.

[2] V.Shilo, "Intelligent CAMAC-controller Family Based on the 32-bits Inmos Transputers", Proc. 7th International School-Seminar on Automation and Computing in Science, Engineering and Industry, Yalta, Crimea, 1996.

[3] D.Yu. Bolkhovityanov, Yu.I. Eidelman, "Control System for VEPP-5 electron-positron complex", Proc. ICALEPCS'97, http://www.aps.anl.gov/icalepcs97/paper97/p066.pdf

[4] D.Yu.Bolkhovityanov et al., "Experience of Using Multi-monitor Workstations Under XFree86 4.x in VEPP-5 Control Room", Proc. ICALEPCS'2001, http://www.slac.stanford.edu/econf/C011127/WEAP030.pdf

[5] Matrox G450MMS http://www.matrox.com/mga/products/g450_mms/home.cfm

[6] A.N.Aleshaev et al., "Forinjector VEPP-5 Klystron Gallery Control System", Proc. PAC'99, New York, 1999 http://accelconf.web.cern.ch/AccelConf/p99/PAPERS/MOP41.PDF

[7] D. Bolkhovityanov, R. Gromov, "Experience of Using uClinux-based CAMAC Controllers in VEPP-5 Control System", PCaPAC'2002 poster TU-P14.

[8] A.A. Starostenko et al., "Non-destructive Singlepass Bunch Length Monitor: Experiments at VEPP-5 Preinjector Electron Linac", Proc. EPAC'2000 http://accelconf.web.cern.ch/AccelConf/e00/PAPERS/WEP6B08.pdf