QNX BASED SOFTWARE FOR CONTROL SYSTEM OF FLNR

V. Aleynikov, A. Nikiforov
Joint Institute for Nuclear Research, Dubna, Russia

ABSTRACT

The control system of FLNR cyclotrons uses commercial SCADA FlexControl running under RTOS QNX 4. Good interaction possibility allows integrating self-made programs and commercial software. Basic ideas for inter-process communicating, data flow and overview of FLNR control system are described.

INTRODUCTION

Since 1999 we have been using Supervisory Control and Data Acquisition (SCADA) software named FlexCtrl 4 (BitCtrl Systems Ltd., Germany). It runs under UNIX-style commercial real-time operating system QNX (QNX Software Systems Ltd., Canada). After 6 years of using this commercial software there were written custom visualization server and a lot of device drivers, that successfully replace and expand the regular software.

We have designed and integrated in SCADA device drivers for the following hardware:

- SMARTBOX data acquisition module (FLNR, Russia);
- Rotating encoder input device (FLNR, Russia);
- RADIS GM2524-100 RF generator (RADIS Ltd., Russia);
- EVPU PS24-500, PS15-30, PS25-120 power supplies (EVPU, Slovakia);
- DANFYSIK power supplies models 883, 853T, 855 (DANFYSIK A/S, Denmark);
- FESTO DGE positioning system (Festo AG, Germany);
- PFEIFFER TPG-256, 261 vacuum measurement and control units (Pfeiffer Vacuum GmbH, Germany);
- KEITHLEY digital multimeter model 2000 (Keithley Instruments Inc., USA);
- AGILENT 33220A function/arbitrary waveform generator (Agilent Technologies, USA);
- AMI Model 420 Power Supply Programmer (American Magnetics, Inc., USA).

It was designed and put into operation control systems for 6 charged particles accelerators and radiating installations. Two more cyclotrons (DC-72 and DC-60) are at the final stage: assembled, tested in FLNR and ready for shipping (see table 1).

Table 1.

Year	Project	Descriptions	Country	Process variables
2001	EA-10/10	Electron accelerator	Germany	1800
2002	DRIBs	Dubna Radioactive Ion Beams	Russia	3500
2003	CyLab	ECR ion source	Slovakia	2800
2003	U-400	Isochronous Cyclotron	Russia	4600
2004	U-400M	Isochronous Cyclotron	Russia	5200
2005	IC-100	Isochronous Cyclotron	Russia	3700
…2007	DC-60	Isochronous Cyclotron	Kazakhstan	5700
…2008	DC-72	Isochronous Cyclotron	Slovakia	7600

OPERATING SYSTEM

In the late 90th we decided to renew Control System software based on MS DOS and Turbo Pascal. We decided to launch a market survey on two components:

- Operating system;
- Development tools and run-time systems for the automation of technological process.

In order to find the best OS for our needs we formulated these requirements:

1. **PC (x86) platform support** because all control system nodes were x86 based.
2. **Multi-tasking** and suitable inter-process communication technique to run at the same time PLC's protocol driver, HMI, RTDB server and printer manager.

3. Good **network integration** to link remote nodes allocated over large area.
4. **Openness** and ease to write and integrate **device driver for custom hardware**.
5. Provide an **embedded windowing system** with full-featured GUI to afford convenient Human-Machine Interface.
6. Have **powerful development tools** to create application software.
7. Real-time and **fault tolerant**.
8. Be **commercial, well known** and have good customer support.
9. **Have choice of SCADA** software.

After a general pre-selection, we stopped on three systems – Microsoft Windows NT, QNX and Linux. Other known operation systems (UNIX, OS-9, pSOS, VxWorks) did not meet more then 2 points of our requirements; we did not take them for consideration.

Doing serious real-time coding in MS Windows is not simple work. Moreover writing device driver for custom hardware requires a hugely talented individual (real expensive) who can work in the bowels of the OS. We have plenty of hardware that comes without any communication driver. MS Windows is designed as a server environment for hosting/workgroup/office applications. It is huge compared to QNX and Linux and did not meet real-time, openness and reliability demands.

As a monolithic OS, Linux binds most drivers, file systems, and protocol stacks to the OS kernel. Hence, a single programming error in any of these components can cause a fatal kernel fault. In QNX these components can all run in separate, memory-protected address spaces, making it very difficult for them to corrupt the kernel, or each other. QNX therefore provides an environment for real-time applications that is more robust than Linux and much reliable than the unprotected real-time kernels. Moreover in the late 90th Linux (FreeBSD as well) was in developing phase and did not have final commercial version. Its graphical system was poor and there was no well known SCADA software for Linux (OS-9, pSOS, VxWorks as well).

Finally we have chosen commercial real-time operating system QNX. QNX Software Systems Limited (QSSL) has over 20 years of real-time OS experience on x86 platforms. That time QNX outsells every other real-time OS for PCs and supported in almost 100 countries worldwide. It is modular and designed specifically to allow the competent engineer to control precisely the priorities of the process running on the hardware. It is UNIX-style scalable, multi-user, multi-tasking, real-time, network and POSIX-compliant operating system.

QNX 4 supports Photon microGUI as graphical user interface. It has Windows look and feel. The tool kit also offers a GUI development tool called the Photon Application Builder (PhAB), which allows an application programmer to create prototype GUIs, without writing a single line of code; build an entire GUI, by pointing and clicking; create applications with a consistent look and feel. The QNX package includes the Watcom C/C++ highly optimizing compiler and tool set. The OS is designed to allow users to develop and extend it. The hooks are here and examples are ready to be followed. QNX as an OS is designed for real-time control and open for custom software/hardware.

SCADA SOFTWARE

As soon as we decided to use QNX it was only a few SCADA for QNX available: Sitex, Realflex, FlexCtrl, PCP Virgo, Fiord microSCADA. Upon market analysis and test drives we have selected SCADA FlexCtrl [1].

FlexCtrl is a process control system for the automation of technological processes. It is modular and extremely scalable. The interface to the system is open and the user has the possibility adding custom device driver to the system.

All parts of the FlexCtrl application can be managed with the project engineering system, which configures process model (process variables with all characteristics). The system includes Process Model Editor, Graphics Editor, Network Configurator, User Administrator, Driver, Visualization and Run-time Compilers. Project and Run-time Installation tools complete the project development with generating files for the start of run-time components.

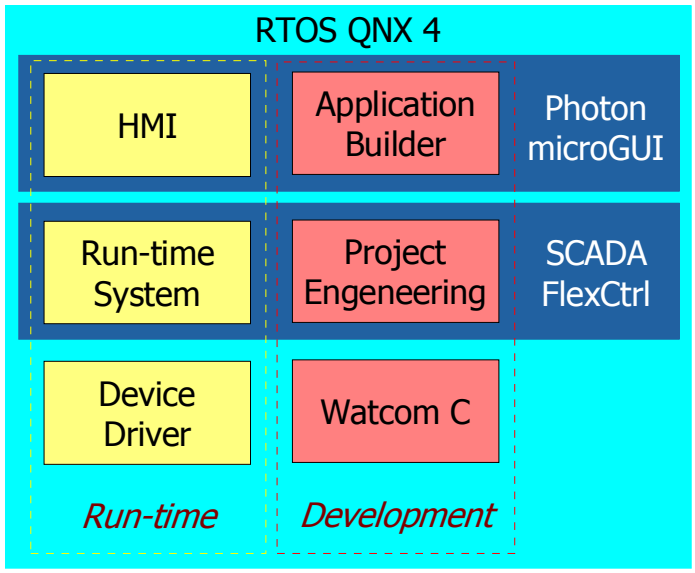

Fig. 1. Basic structure of the software development

After a week of training we were able to develop protocol driver for custom PLC and connect it to the working SCADA project with automation algorithm, graphical visualization and control. Basic structure of the software development is shown on Fig. 1.

APPLICATION SOFTWARE

Since FlexCtrl Graphics Editor and graphical library does not meet our requirements we have decided to develop custom image library. For creating Human to Machine Interface (HMI) we use Photon Application Builder instead of the FlexCtrl Graphics Editor. We use Photon Application Builder as graphical editor, object configurator and application compiler. We have developed a library of functions, which we build into application to control the screen and the keyboard, as well as any input from Visualization Server. Visualization Server intends for managing message queues for every HMI client in the network.

Developer starts GUI design by arranging active images (widgets) on the screen. Every widget represents process variable in text or graphical form. Every active image has property that contains the name of the coupled variable in RTDB. At final step Application Builder compiles GUI and custom library into the HMI application.

At start up HMI connects to the RTDB and reads coupling information from widgets to link them to the RTDB in its memory. Next step it requests Visualization Server to open message queue for it. This queue is a FIFO buffer of events of data changes managed by Visualization Server. Widget picture can be imported from a file of BMP or GIF format. For better look and understanding of control process we have used 3D images created in Solid Edge tools.

HMI allows analyze process data in real-time trend, store and retrieve a set of variables to repeat important system modes. Reports can be configured, printed and exported in text form.

IPC AND DATA FLOW

QNX depends on the exchange of discrete packets of information – *messages* – to handle virtually all inter-process communication. Message passing lies at the heart of the operating system's microkernel architecture, giving the OS its modularity [2]. This paradigm applies to all levels of programming, from device drivers to file system and LAN. From the standpoint of a user's process there is no difference between a local call and a call from the network and hence all resources on all network nodes are transparently available everywhere on the network.

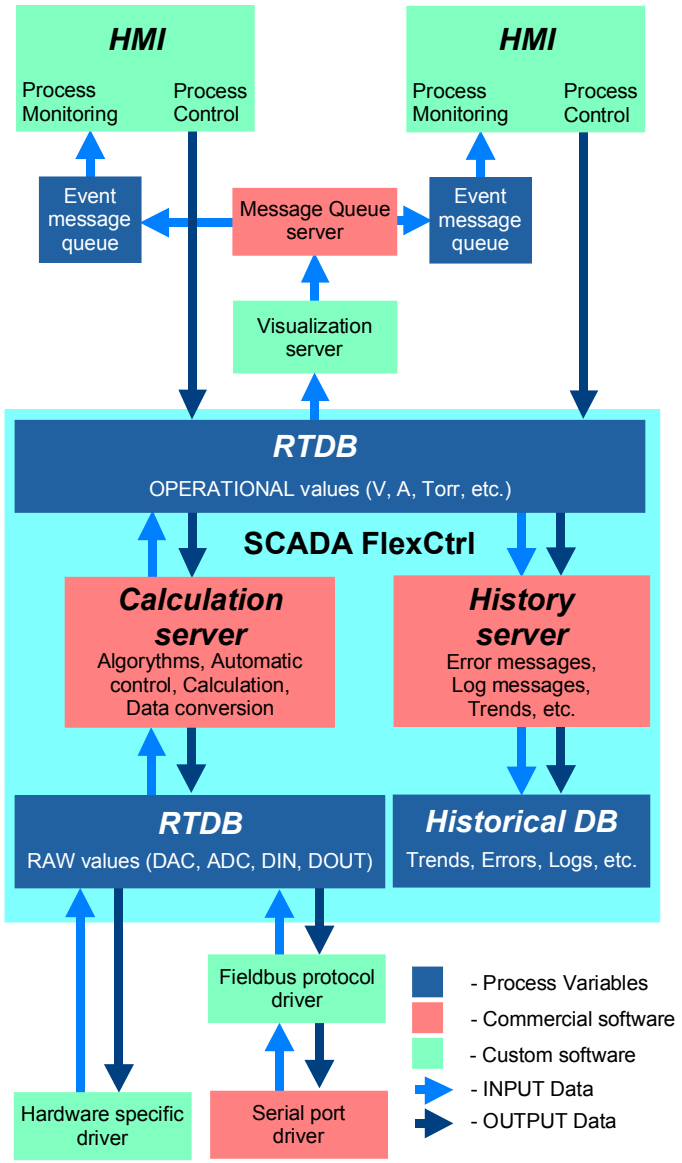

Fig. 2. Data flow

FlexCtrl strongly uses this message passing technique as for internal run-time communication as for interfacing to device drivers. The components of FlexControl can be partitioned, installed and run on several computers of standardized PC network [3]. Every job of some significance in FC is handled through a single computational process (server). FlexControl comprises a long list of these tasks:

- Process administration and process supervision
- Real-time database (RTDB)
- Calculation server (Soft SPC tasks)
- Visualization server and HMI
- Alarm server
- Message server
- Protocol drivers

The tasks are able to communicate and synchronize with one another. The data from the transmitting process are exported via message passing over the network and imported directly into the address area of the receiver process. Deadlocks and blocks do not occur in any form in FlexCtrl. The configured coupling parameters of process variables (PV) and drivers are retrieved by respective driver process when started up. The data flow diagram for process control and monitoring is shown below, see Fig. 2.

Monitoring

For process monitoring device driver cyclically reads device status and compares it with the old one saved in its memory. In case of any changes the driver writes new values to the real-time database. Control Server manages real-time database access and triggers off coupled server for every instance the PV is changed in the real-time database. Calculation Server processes these events, converts ADC codes (raw values) to real physical (operational) values and writes them to RTDB. Visualization Server gets information about changes of operational values as well. It appends these events of data change to the message queue (FIFO) driven by Message Queue Server. The Human to Machine Interface (HMI) cyclically checks for the messages in queue, reads them out and represent by graphical image or text on the screen.

Control

Operator controls the process via HMI. He presses buttons, enters new values, etc. HMI writes new values directly to RTDB. The fact the operational value (Ampere, Volt, Torr, etc.) was changed causes Calculation Server to convert it to DAC code and write it to RTDB (raw values part). Next step Control Server sends message with the new DAC code to the coupled driver. Device driver receives new data from Control Server and issues write command to the device.

CONCLUSION

The disadvantages of using FlexCtrl and QNX are:

- Lack of QNX 4 device drivers for the most recent hardware (video, network);
- In a few years QSSL and BitCtrl will not provide support for outdated software versions (QNX 6 and FC 6 are available now);
- Poor support of office application software;

The benefits are:

- Stability since the core of the OS and SCADA system is well optimized and tested;
- Development tools decrease total project engineering time and allows concentrating more on visualization and automation algorithms;
- Flexibility. Openness and good interaction possibility allows easily add new hardware;

At present time we do not have serious problems with the selected solution (QNX and FC) and we will continue using it for the further projects.

REFERENCES

[1] V. Aleynikov, S. Paschenko, “Using commercial SCADA in control system for ECR CyLab.” PCaPAC 2000. Hamburg.

[2] Rob Krten, “Getting started with QNX 4. A Guide for Realtime Programmers”. PARSE Software Devices, 1998.

[3] FlexControl - System Architecture Manual. BitCtrl GmbH. October 1998.

[4] V. Aleinikov, A. Nikiforov, “Integtrating custom software and commercial SCADA”, NEC'2003.