THE SOFTWARE FOR RESEARCH FACILITIES AUTOMATING

A.V.Kuryakin, Yu.I.Vinogradov
Russian Federal Nuclear Centre – All-Russia Scientific Research Institute of Experimental Physics (RFNC-VNIIEF),
607188 Sarov, Nizhny Novgorod Reg., Russia

Peculiarities and requirements to the software of systems of research physical facilities automation are addressed. Principles are described of the software package CRW-DAQ arrangement, applied to a series of research facility automation at VNIIWF, JINR and CERN.

Introduction

Data acquisition and automation tasks are diverse and numerous. And different application areas impose different requirements on both the hardware and the software. The differences result not only from the technical characteristics of facilities, but also from their operating conditions, maintenance, character of the tasks, financial provision. For instance, requirements to an industrial automation system are quite different from those to a research facility.

The main task of the industry is providing troubleproof manufacture of some products. Therefore, industrial automation usually operates with an unchangeable set of standard equipment, in permanent operating conditions, solving one and the same circle of tasks. A uniform by the composition set of measuring devices, often from one firm, is usually employed. Question of the measuring equipment and the software cost is not very acute, since the expenditures in this case are nonrecurrent, long-term, and incomparably smaller than the production value. Hardware and software could be in operation for long time with minimal upgrades being done to it, what decreases their maintenance and support costs. For industrial automation systems many software packages were created, such as LabView, Genesis, etc. These packages have proved their practical usefulness, high reliability, convenience. It should be mentioned that for the industry the jural issues are of particular importance, certificate availability on the equipment and the software. That is why, commercial software packages are preferable for industrial automation, but neither homeware, nor systems with open code for which jural assurances are usually absent.

In research facility automation tasks the conditions are quite different. The main purpose of any research activity is novel knowledge obtaining. Therefore, an element of unpredictability lies in the nature of any research process. It inevitably affected the character of the research facility automation tasks, making them more dynamic.

1. Peculiarities of research facilities automation.

Last years (1997-2004) the authors were actively engaged in creating a series of systems of research facility automation and data acquisition for performing diverse experiments at VNIIEF (Sarov), JINR (Dubna), SPSU (St. Petersburg), CERN (Switzerland/France). The most part of these research facilities is related to the area of tritium technologies. Among the facilities that have been automated there should be specially mentioned:

- PROMETEUS – facility for studying into hydrogen isotopes permeability and diffusion, VNIIEF, Sarov [1];
- TRITON – tritium target for μ-catalysis investigation at the phasotron of NPL JINR, Dubna [2];

- ACCULINNA – tritium target for neutron-redundant systems studying at the cyclotron of NRL JINR, Dubna [3].

The accumulated experience allows us to reveal some characteristic peculiarities of computer-aided systems of research facilities. Thus, the values to be measured in investigations are extremely diverse, and their ranges are very wide. For example, at the facilities TRITON and ACCULINNA temperatures to be measured range from 10 K (liquid hydrogen) to 1500 °C, pressures – from deep vacuum up to thousands of atmospheres, volumetric activitiy from 10^{-7} to 10^{4} Ci/l. It is accompanied with spectrometry, chromatography, nuclear measurements, etc.

The variety of measured values compels the developers to use hardware of different firms, as well as that one designed specially for the given task, as no firm can provide the required set of measuring tools. Therefore, practically all the research facilities by their composition are hybrid, e.g. at TRITON and PROMETEUS facilities devices of ICP-DAS, Balzers, Advantech are used, as well as the original VNIIEF-developed ones.

As a rule, hardware and software is modified for each specific experiment or measurement series. It is connected with the character of research tasks – the investigator tries to embrace as large circle of phenomena as possible, and the circle itself is changing during the investigation process.

In the case of research tasks solving the control algorithm is often preliminarily unknown, and is worked out during the operation. The result is that the software creation for research facilities is much more effortful job than software of other types creating.

The peculiarity of the research facilities is that their hardware drivers are written not for a specific device, but for a certain problem. Standard drivers, even for standard devices, are capable of providing only most simple kinds of measurements, what is not always suitable. For example, standard drivers of ADC PCL-818L are able to provide high-rate buffered data input. But if according to the measurement results, for instance, prompt response needs to be transmitted to the executive device through DAC, drivers both for ADC and DAC have to be written specially for the given task.

Specialized equipment use and necessity of frequent algorithm correcting imposes heightened requirements on the software in terms of fast and quality software upgrading. Therefore, the authors deem that research tasks necessitate that the software source codes be open, and, if possible, built-in programming languages for control algorithms fast creating should be available.

The research facility equipment often operates under extreme and even hazardous conditions. That is why, high requirements are set to reliability and predictability of both hardware and software of the computer-aided systems.

Small research collectives cannot afford the programmers special training, what is quite necessary for creating predictable and safe systems on the base of closed commercial software. It is one more argument in favor of open software code, which could be read and understood, if needed. It is of particular avail even in the case, when its modification is not needed. For example, availability of complete Runtime source codes of Borland Pascal and Delphi libraries has enabled the software operation reliability to be essentially enhanced. And no corrections have been made to the “firmware”, it was used only as the most complete reference programming document.

The authors with scepticism regard a possibility of “black box” technologies application to the automation tasks. Such technologies are expedient from the commercial standpoint only, as they allow their producers to keep in secret the program implementation presenting the interface only. This is, likely, acceptable solution for office applications, but not for automation tasks, where details of the implementation, e.g. undesirable delays, can be of decisive importance. Information on the realization details being concealed (the “black box” basic idea) in no way contributes to the software operation predictability enhancement.

Legal questions related to certification do not play critical part in the research area. Scientific result reliability is usually corroborated by rational arguments and calibration measurement results, not by an appropriate certificate of the hardware and the software.

The use of commercial software does not automatically allow reducing the number of programmers and pertinent charges. In practice any research team necessarily includes a software specialist, because the character of research works implies that the facility couldn't be made forever unchangeable, and close cooperation of the investigator and the programmer is needed.

In view of the above facts, the authors consider application of standard industrial automation packages to the research facilities to be inexpedient. These packages are expensive, oriented to standard hardware, closed in terms of the source code. The authors regard as optimal for research facilities the open software use, or homeware creation.

2. Principles of research facility software constructing.

Having made the decision of homeware use for the tasks of facilities automating, the authors faced with the two ways – either not-large programs "from scratch" creating, individual for each facility, or writing a specialized package applicable to all facilities of the similar class and supplementing it with a set of applied programs individual for each facility. The both approaches have their pluses and minuses. Software for an individual facility could be made very effective and convenient, but it would take a lot of time and would be hard-to-maintain, and the maintenance problems tend to grow with the number of the facilities involved. Specialized package requires large initial expenditures, but further on it is easier to create and maintain programs for specific facilities. In practice the both approaches were used, but nevertheless the preference was given to solving tasks within the frames of a specialized package.

As a result, for several years, in process of practical measurement tasks solving the authors have developed a specialized package for research physical facilities automation, called CRW-DAQ**.** This package was applied to automate considerable number of research facilities. Only in some peculiar cases the software was developed "from scratch" for the specific device or experiment. At present two versions of CRW-DAQ exist - CRW16 for DPMI and CRW32 for Windows-95/98/NT/2K/XP. The package architecture enables it to be easily realized in any other operating system, though practical necessity has not yet arisen.

The software, the authors have developed for research facilities automation, is based on the following principles.

The software division into system and applied. The system software, common for all the facilities, provides graphic interface, language means for applied programming and other common services. The applied software, individual for each facility, makes use of the provided means for specific task solving. The advantages, such division provides, are as follows:

- The system software is the same for all the facilities. It is easy to maintain and upgrade, what enables efforts concentrating on its reliability increasing. CRW-DAQ package itself in fact plays part of the system software.
- The system software provides the applied programs with the built-in programming languages with well-protected software interface. It decreases possible errors in the applied programs influence on the reliability of the system as a whole.
- The applied software, which includes configuration files and applied programs written in the built-in languages, provides for the necessary flexibility of the software. Being written in the built-in language, it could be easily modified. Moreover, the system allows it to be modified just in process of measurements.

Priority free threading. Measurement programs are built as several software streams executed in parallel, priorities of which could be different. Being correctly applied, the priority system enables assured allocation of the appropriate time quantum to critically important streams, even with the processor high load.

High concurrency degree. In a standard measurement system CRW-DAQ several tens of software streams can be executed. Through the high concurrency degree each of the applied tasks executed in parallel can be relatively simple. First we had some anxiety, that the large number of streams would lead to the efficiency decrease, but the practice has disproved that. Taking into account tendencies of the modern computer technique and operating systems development (multiprocessor systems, fast switching of the stream context, presence of internal concurrency at the processor level), and the character of measurement tasks (in measurements many parameters need to be monitored in parallel) the authors consider that from the standpoint of strategy the chosen way is correct.

Real-time database. This is a set of public tags (scalar variables) and curves (dynamic arrays), which store the applied programs statuses and the measured data. As any applied program is executed in its own stream and has logically isolated address space, communication between the applied programs is implemented mainly through the database, except for the message exchange. In general, operation of the measuring systems controlled by CRW-DAQ can be presented as a set of running in parallel applied programs to populate and modify the database of tags and curves.

For example, the oven temperature regulation can be implemented in the following way. The ADC measures the sensor signal and puts it into the database. Operating in parallel, calibration program extracts the sensor signal from the database and according to the calibration translates it into temperature. Operating in parallel, regulation program extracts temperature from the database and forms the oven control signal. Finally, DAC takes from the database the control signal and transmits it to the executive device.

Such structure of the applied programs gives considerable flexibility to the system. In the above example it is easy to substitute one type of the ADC driver by another, if only its influence on the database is analogous. It enables the same applied programs to be used in many measuring systems, what saves essentially the development time.

Multiwindow real-time graphic interface. Allows the measuring process to be monitored and controlled in real time during the measurement process. The monitored data are displayed in windows in the form of mimic panels, graphs, texts, spectra or surfaces. The graphic interface meets the special requirements.

The main task for the graphic interface within a measuring system is not to impede measurements executed in parallel. For that the stream separation is provided for the user's interface stream and the control streams. All measurements are executed in separate program streams with high priority. The image refreshing and drawing goes on timer in the stream with normal priority. The drawing stream, having relatively low priority, always could be displaced by the measuring streams, i.e. it does not impede the measurement tasks.

The real-time representing system is based on the tracking monitor. It means that the most part of drawings the system implements by itself, with no applied programs involved, which in most cases do not care of drawing and even "don't know" about it. The applied programs work with tags only, and the monitor by itself keeps track of statuses of the represented tags and, when needed and possible, refreshes the image on the screen. For this reason the applied programs become simple and easy-to-understand.

Software interface of the applied programs contains no drawing functions, which could block the stream execution for long time. All the drawing functions are implemented through the messages sending measuring stream without blocking. The drawing procedures are also thoroughly elaborated to minimize the time of common resources blocking. Thus, the probability of measuring streams suspending due to the common resource blocking is decreased.

Asynchronous input-output system. Most part of the input-output functions is buffered with the use of FIFO and is executed asynchronously. It means that the input-output functions being called do not block the measuring stream. As a rule, individual input-output subsystems (console window, COM-ports, audio messages etc.) are implemented as server streams, which receive commands and yield results through FIFO.

Refusal of blocking calls use. Blocking calls are the calls suspending program stream execution till some new external event coming. For example, in Windows SendMessage function, WaitForSingleObject, etc. are the blocking calls. These methods are applied to lessen the processor load at diverse events waiting. The danger is that for some reasons the blocked stream might not wait until the expected event coming, what is equal to its hanging. Therefore, when creating the measuring package we always gave preference to periodic checks of event status without waiting and blocking. For example, to send a message the non-blocking call PostMessage was always used, and for temporal blocking the preference was given to Sleep call, which could not cause hanging. These measures, perhaps, are not optimal in terms of the processor load, but they increase the operation reliability, what is of greater importance for measuring systems. Blockings (critical sections mostly) were used mainly to protect public resources, and the blocking time was thoroughly controlled to be minimal.

Built-in programming languages availability. CRW-DAQ package has three integrated languages:

- Daq Script – expression interpreter, something interim between C and Basic interpreters. This simple interpreter serves in cases, when the program cannot be formalized at the compilation step. For instance, if temperature regulation according to the certain user's law is needed, the user can input the law as a formula in the input field, and the interpreter will fulfill it.
- Daq Pascal – compiler of the simplified Pascal language, generating a code of some virtual machine, plus interpreter of this virtual machine code. The language task is to create well-protected code for usual applied programs. High protection is provided, because the virtual machine code interpreter is well-protected, though it operates rather slowly as compared to the natural machine-language code. In practice the most part of the applied code of the measuring systems was written in Daq Pascal languge.
- Object Pascal – compiler of the object-oriented language Pascal into the natural for the given system machine-language code. For CRW16 this is Borland Pascal 7.0, for CRW32 – Delphi 5.0. The built-in compiler makes it possible to create, edit, compile and immediately execute subprograms, implemented in the form of DLL-libraries, which provide access to all the machine capacities. DLL-programs do not compete with Daq Pascal programs, they are created for special purposes only, when a lack of the Daq Pascal means or speed is felt. For instance, high-priority measuring device drivers or complex mathematical algorithms could be implemented as DLLs.

Each of the applied programs created on one of the built-in programming languages is executed in a separate stream and has logically isolated address space. To communicate with each other the applied programs can use a real-time database, as well as the message exchange. Directly during the measurement process each of the applied programs can be edited and compiled (in this case the applied program stream is temporarily suspended). The other applied programs executed in parallel continue their normal operation. This feature of CRW-DAQ package makes it possible to develop or modify the applied programs without measurements interrupting. Few measuring systems possess such capabilities.

As the practice has shown, availability of built-in multilevel language support is not something excessive. It allows reducing the task solving time, enhancing the applied programs flexibility and reliability. For each individual task a language is selected to provide its simple and quick implementing.

Combined environment for measuring systems developing and implementing. In most commercial systems the development environment is separated from the implementation system, that is why to modify the control program the measurements must be stopped, the project recompiled, and the measurements then restarted. It is not very convenient for research facilities, when control algorithms are often groped for in process of investigation, directly during the measurements, because after the measurements stopping the initial conditions are difficult or time-consuming to be restored. CRW-DAQ system by providing high concurrency and

possibility for development and implementation environment combining enables applied programs to be developed and debugged without measurement process interrupting. For example, one can run vacuum measurements and in parallel debug the mass-spectrometer driver. Since each Daq-program is executed in its own program stream, the mass-spectrometer driver recompilation does not directly influence operation of the vacuum measuring subsystem. Thus, development and implementation environment combining enables hastening the process of measurement systems development and modernization.

Another reason to create a combined environment of development and implementation is the desire of providing the package "all-sufficiency". It is connected with the fact that many works are carried out during business trips, at different institutions or remote sites, sometimes on "weak" computers. In such conditions the personnel often have no opportunity to deploy large development system, like Delphi, Visual C++ or LabView. CRW-DAQ package, taking about 20 MB, allows the whole development cycle of measuring systems to be implemented, including creating the interface, codes of applied programs and drivers, calibrations, help system etc. The entire CRW-DAQ system deployment takes several minutes, requiring no any additional packages or drivers installation.

Developed means for real-time tasks diagnosing and debugging. To such means there should be referred the resource monitor, "watchdog", debug files, console windows. The resource monitor is a console window, which enables watching the processor load for each of the applied programs, their polling frequency, physical and virtual memory use and other information related to the computation resources utilization. The watchdog is a monitor, which tracks the running program streams activity and gives warning if some processes are not showing "signs of life". In fact, it is the software diagnostics of applied programs hanging. The debug files and console windows are buffered means of text messages asynchronous output to a file or window. It should be mentioned that, the built-in languages of CRW-DAQ have neither debugger nor tracing. For real-time tasks tracing makes no sense practically, because for these tasks not only what is executed is of importance, but when it is happening, as well. But tracing suspends the program execution by that violating the process time characteristics. Therefore, the most useful debug mean for the real time mode is debug messages asynchronous output into a file or console window, as this does not practically violate the process time characteristics. For instance, the message exchange stream can be directed via COM-port to a file, to analyze afterwards operation of the devices connected to the port.

Support of standard communication interfaces. In most cases complex research facilities are implemented in the form of a distributed network, containing one or several control computers, and remote measuring devices connected through Ethernet, RS-232, RS-485. In all cases, when it was possible from the technical standpoint, preference was given to external devices connected via standard interface, e.g. RS-232. First, this enhances the system flexibility, allows the number of channels and devices to be incremented. Second, it is connected with the rapid development of computer technique. On average, each 2-3 years there takes place computer and operating system generation change. For peripherals connected through a standard interface this change is painless, but not for devices on the internal buses (ISA,PCI). The software interface of the package built-in languages includes functions necessary for work with the devices RS-232, RS-422/485, GPIB, ISA, PCI, LPT, Ethernet. This enables the considerable part of the device drivers to be implemented as applied programs in the built-in languages, not overloading the package kernel.

Library of built-in drivers. Drivers of the devices most often used in practice, were included into the package kernel. E.g. devices of the series ADAM-4000, ICP-DAS 7000, 87000, Balzers, some measurement cards of Advantech. The most part of the device drivers were not included into the package intentionally, because the package allows drivers to be implemented as applied programs in the built-in languages (Daq Pascal, Object Pascal). In the cases when these means enabled the set tasks executing, the drivers were not included into the package, but

remained the applied programs. As a result, the package kernel has become more compact and reliable, and it is rarely modified.

Built-in system of acquired data archiving. The data obtained during measurements can be stored for posterior offline processing. To save them several formats are provided. Most often used is the internal binary archiving format, allowing windows with all their content to be saved and loaded (with text, curve collection, etc.). Data exchange with other applications is usually implemented through text tables. At present, access to standard databases is not implemented, since in the area of scientific investigations they are rarely used. If the data need to be imported to or exported from the different formats a DLL library is to be written in the built-in language Object Pascal to implement the required actions.

Built-in means of data processing and preliminary analysis. Availability of these means enables data coming in process of measurements partial or full online analyzing. For instance, during the measurement process one can smooth a noisy curve, evaluate its derivative, subtract background, etc. Most of data processing functions are implemented as DLL programs written in the built-in language Object Pascal. On the whole, the processing is oriented towards graphic windows: input data in the form of a set of graphs should be placed into the window, playing part of the argument, and then an appropriate DLL should be called. The processing result appears in the new window, also in the form of graphs. This stile of operation strongly differs from that of table-oriented packages, such as Excel, Origin etc. The authors deem that for online analysis of the experimental data graphs are more natural and convenient form of data presentation than tables. Graphs make possible to assess by eye the character of dependence as a whole, what is impossible with table presentation. During experiment it is of particular importance, because the investigator often has to make expeditious decisions on the base of the coming data analysis.

Built-in help service. The package contains expanded help service including information useful for both programmers and users. The help service includes also the diverse information not related directly to the package, but often required in practice, e.g. certification of the most frequently used hardware, tables of physical constants, etc.

Conclusion

Created by the authors software package CRW-DAQ allowed a whole series of research facilities to be successfully automated. The package provides a good potential for further development, because it has integrated means for developing measuring systems and data processing software, a developed graphic user's interface.

In addition to the above tasks the software package CRW-DAQ was successfully used at creating cooling and thermostabilization systems of the photon spectrometer PHOS [4, 5], developed for ALICE experiment at CERN.

Refernces

1. Yu.I. Vinogradov, A.V.Kuryakin, A.A. Yukhimchuk et al. Automated monitoring, control and data acquisition system of "Prometheus" test bench// Material science, №1, 2002, P.46-50.
2. Yu.I. Vinogradov, V.S.Aryutkin, A.V.Kuryakin et al. Automated monitoring and control system of the facility for preparation of gas mixture in muon catalyzed fusion experiments. Preprint of RFNC-VNIIEF, 82-2002, Sarov, 2002, 26 p.
3. Yu.I. Vinogradov, V.S.Arutkin, A.V.Kuryakin et al. Monitoring and control system of tritium target for investigating exotic neutron-redundant nuclei. VAN&T, Series: Physics of nuclear reactors, issue ½, 2002, Materials of the 51-th Conference on Nuclear Spectroscopy and Nuclear Structure. pp.197-200.
4. Technical Design Report of the Photon Spectrometer (PHOS), CERN/LHCC 99-4, ALICE TDR 2, 5 March 1999.

5. V.F.Basmanov, A.M.Blick, M.Yu.Bogolyubovsky et al. Tetsts of the 64-channel prototype of the photon spectrometer for ALICE experiment. VAN&T, Series: Physics of nuclear reactors, issue ½, 2002, Materials of the 51-th Conference on Nuclear Spectroscopy and Nuclear Structure. pp.204-207.