PROGRAM MANAGER FOR DC-280 CYCLOTRON CONTROL SYSTEM

V. Zabanova, V. Aleinikov, S. Pashchenko, K. Sychev

FLNR JINR, Dubna, Russia

E-mail: badver@jinr.ru

On March 25, 2019 the experimental hall of the Superheavy Elements Factory (SHE) was opened at the FLNR JINR and its basic facility—the DC-280 cyclotron—was launched. The control system software of DC-280 is based on the NI LabVIEW platform with the Datalogging and Supervisory Control (DSC) module. It consists of many software programs that perform the corresponding tasks: device drivers, alarm monitoring, beam diagnostics, user interfacing, etc. The Program Manager was developed to supervise running processes and to inform operator in case of failures. The control of the new version and the update of the software modules are implemented. This paper describes the algorithm and the user interface of the Program Manager.

Keywords: DC-280, control system, Program Manager

Veronika Zabanova,Vitaly Aleinikov, Sergey Pashchenko, Kirill Sychev,

1.Introduction

To simplify design and ease of maintenance, the control system was divided into three subsystems: Injection (ECR source, axial injector), Accelerator (cyclotron, extraction, transport), and Low-Level RF. Each of them describes certain tasks. They are simply as follows: to produce, inject, accelerate, extract, and to transport the beam to the target. The total amount of the process variables is about 9000. Every subsystem requires specific drivers. Twenty drivers for field devices of various types were implemented [1].

Each subsystem project consists of many software modules that need automated control. For this purpose, a Program Manager was developed. It performs the following tasks:

- Automatically runs software modules package at startup;
- Controls each module individually (run, stop, restart);
- Updates new versions of programs from a centralized repository;
- Monitors the performance of the modules.

The program was developed in the LabView environment just as was the entire control system of the DC-280 cyclotron.

2.Algorithm

To implement the tasks, a version of the Producer/Consumer design pattern was used, where a user interface (producer) produces messages and tasks (consumers) consume them [2]. In this program, messages are also produced from a consumer loop (Fig. 1).

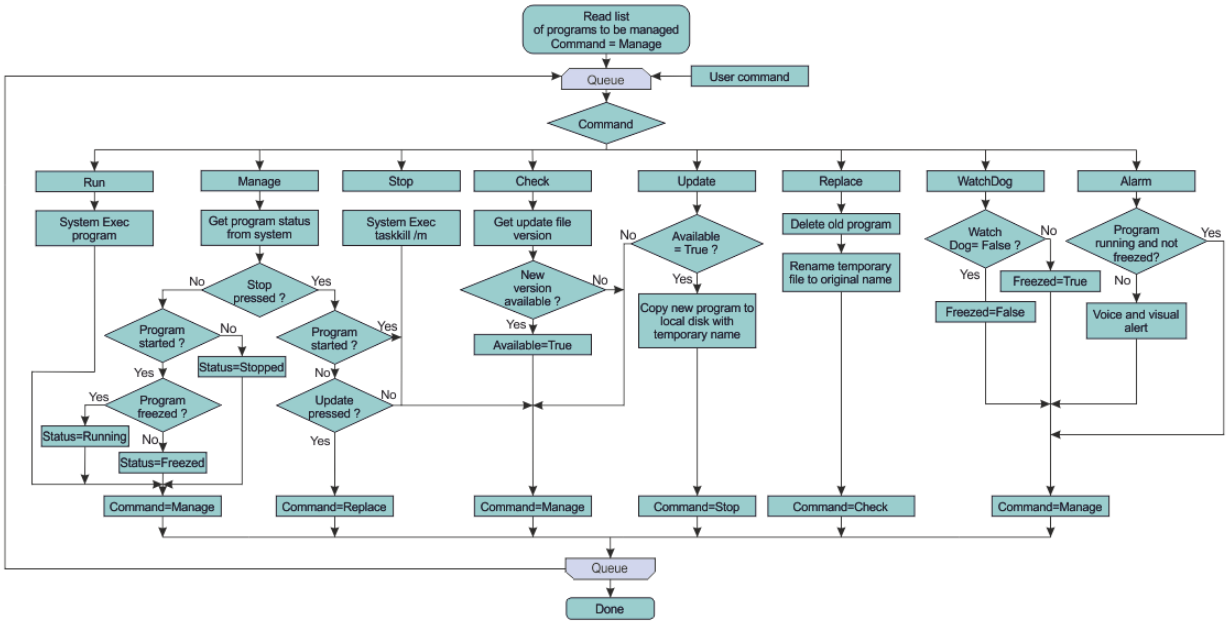

Figure 1. Program Manager Algorithm

The program repeatedly executes the following steps:

1. The user interacts with the front panel, adding a user command to the queue. The watchdog command is added in queue periodically.
2. The consumer reads the command, removes it from the queue, and executes it.
3. The consumer produces another command by the results of execution and adds it to the queue.

3.User Interface

The Program Manager starts automatically on each computer of the DC-280 network. Then it automatically starts program modules listed in the configuration file.

In the Program Manager, each program module is presented as an object with a set of parameters: a name, a path to the executable file, control buttons, a startup checkbox, control buttons and an operation status indicator, and version information (See Figure 2). The list of program modules can be edited in the operating mode and saved as a new list in the file if necessary. Different configuration files can be used for different subsystems. The user can also change the location of the new version repository, adjust the volume of the voice alarm, and choose the voice type.

The startup checkbox indicates that all program modules marked with a checkbox will automatically start when the manager starts.

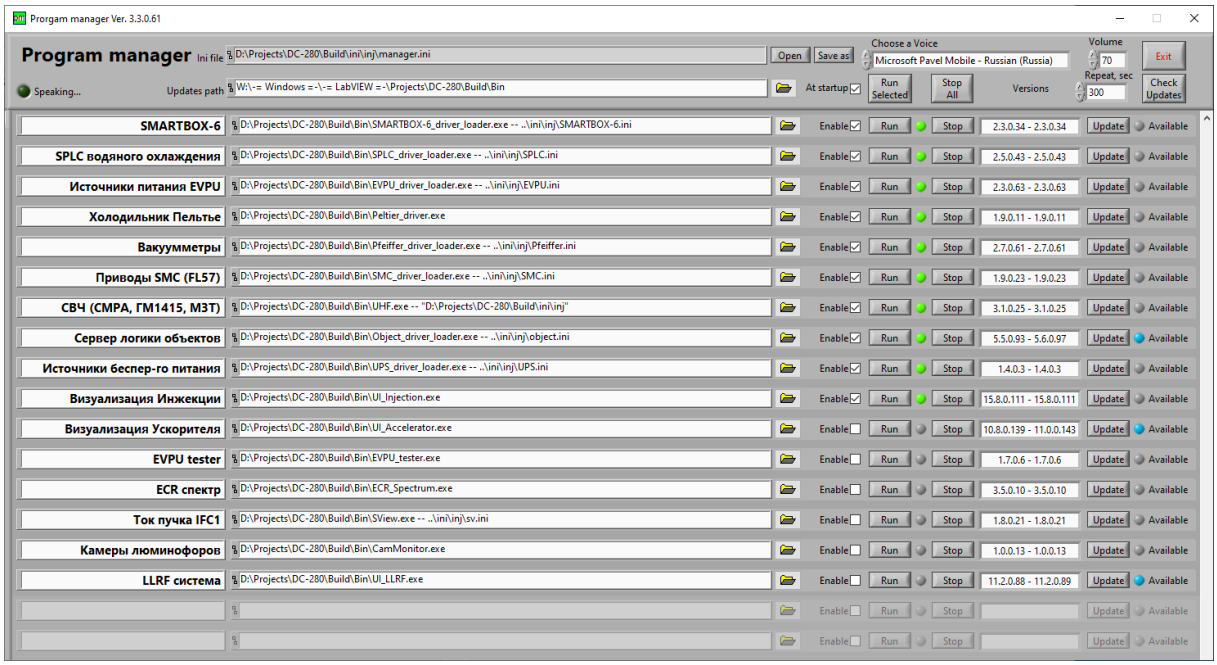

Figure 2. Program Manager Front Panel

To ensure the reliability of the system, it is very important to know that all the modules are running and not hanging. Two techniques were used for this purpose: the one that checks whether the module name is present in the list of the Windows Task Manager and the Watchdog Logic.

Based on these two techniques for checking the status of the modules, the manager generates an alarm if necessary.

Every driver executes at least one process that interacts with the dedicated device. Every process has its own time counter that is supervised by a driver. If any of these processes stop, the driver will increment the value of the "watchdog variable". The Program Manager reads this variable and if it is greater than zero, an alarm occurs. To exchange data on the network, HMI uses shared variables. All user interfaces are clients or subscribers to the Shared Variable Engine [3] (Fig. 3).

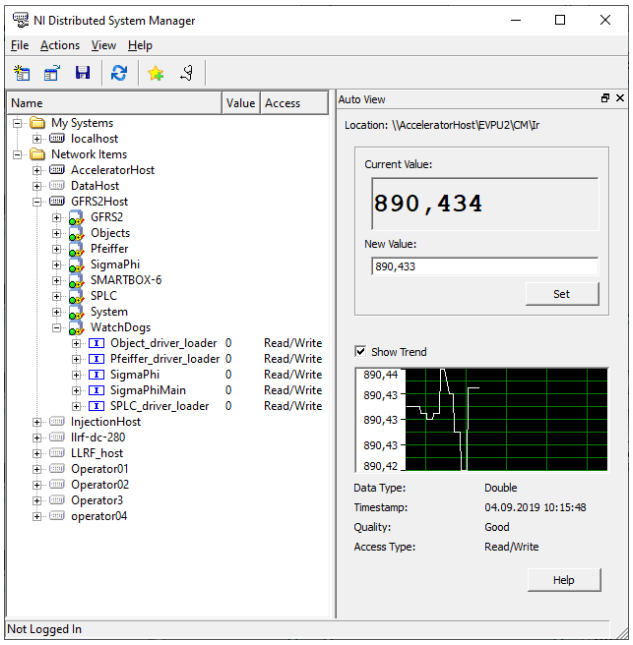

Figure 3. Shared variables

4.Conclusion

The Program Manager was used on all project computers for a year. During this time, it coped well with all the assigned tasks. It automatically launched all the necessary software modules after the server restarted. Voice notification allowed the operator to respond quickly and call service personnel to resolve the problem, which increased the reliability of the project.

Auto run of modules allows to simplify the startup of the software.

The available update service significantly saves system maintenance time.

The Program Manager is a standalone application and can be used on other systems as well.

References

[1] V. Aleinikov, I. Borina, A. Krylov, S. Pachtchenko, K. Sychev, "Using LABVIEW to build distributed control system of a particle accelerator." ICALEPCS 2017. Barcelona, Spain.

[2] National Instruments. Queued Message Handler Template documentation. Available at: http://www.ni.com/tutorial/53391/en/ (accessed24.09.2019)

[3] National Instruments. Using the LabVIEW Shared Variable. Available at: http://www.ni.com/product-documentation/4679/en/ (accessed 24.09.2019)