DOI: 10.17199/NOBUGS2016.4

The unified software package Sonix+. Analysis of development experience

A.S.Kirilov

Frank Laboratory of Neutron Physics, Joint Institute for Nuclear Research, 141980 Dubna, Moscow Region, Russia.

E-mail: akirilov@nf.jinr.ru

Abstract. The software package Sonix+ (Sonix) [1] developed as a unified control software for neutron instruments. It has been in operation since 1995. Currently it is used at instruments of the IBR-2 (FLNP, JINR), as well at some instruments at other centers of the Russian Federation (totally about 20 installations). Though the main ideas of the Sonix + are largely similar to the decisions taken at other centers of the NOBUGS community, the consideration of specific needs and traditions prevailing at the FLNP, had substantial influence at the structure of the complex, and some implementation decisions. This applies, for example, the choice for the operating system of instrument control computer, implementation details of so-called "database" - parameter storage with fast access for inter module communication, a graphical user interface, choice of the spectra recording format, etc. During the long period of exploitation the complex was developed, it has been added new components and has been changed to the new requirements. This led, in particular, to the adjustment of a number of initial decisions, and to a certain eclecticism. Some implemented features have unclaimed, and others, on the contrary, have required further development.

1. Few words about history

The prototype of the Sonix was maintained at the Neutron Spectrometer of High Resolution NSHR) at beam 7a of the pulsed reactor IBR-2 of the FLNP JINR in March 1995. A VME based computer running Os-9 was used for the instrument control. The GUI was based on the X11. The Varman database [2] has been used for the inter client communication. Further this software has been adapted for several other instruments and was named the Sonix (SOftware for Neutron Instruments on the X11 base). The adaptation was not very easy because the initial software did not intended as universal one. For example, there was no possibility to organize hierarchical structures of execution modules, the custom made script was rather poor, there was no visualization of spectra for data from PSD, etc.

In the early 2000s it was decided to replace the expensive control VME computers with cheaper PCs, and the Unix-like Os9 operating system with the Microsoft Windows. This stimulate to re-create our control software due to obtained exploitation experience and recent trends. The new version is called Sonix+. Currently the Sonix + is the set of modules, united by the common philosophy, which allows to organize control system for arbitrary instrument rather simply.

2. Main features of the Sonix+

The Sonix+ is a unified, universal modular set of modules. It has many features both in structural principles and implementation ideas more or less similar to most of modern control software [3]. There are some important differences.

Sonix + is mostly local system. Computation power of modern computers usually is sufficient to do the whole job, so this decision simplifies the system maintenance. For those applications that require synchronization with external computers so called "Cchannel" (command channel) module is provided. In our practice this need is very seldom. For remote user access several possibilities are available - the Windows Remote Desktop, the VNC, the WebSonix service [4], etc.

The Microsoft Windows is used as an operating system on control computers. This choice was made on the urgent wishes of our users, many of whom are perceived the Unix-like Os9 operating system very negatively. Another reason for this choice was the advantage in the software development systems, compared with the Linux at that time.

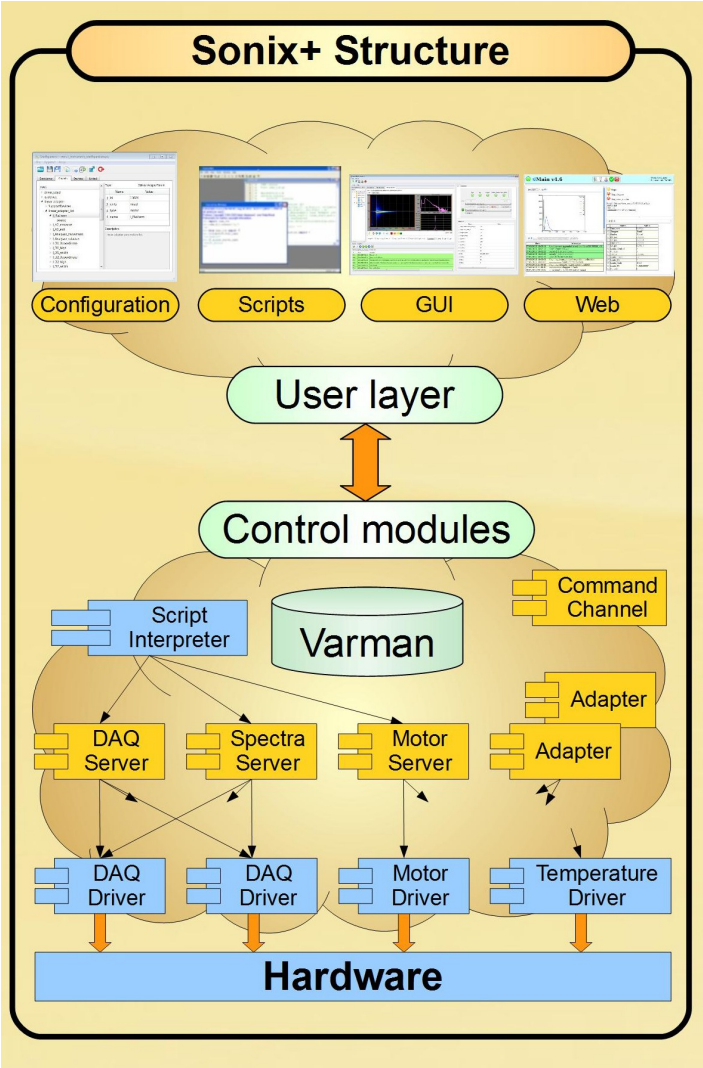

Figure 1. The Sonix+ structure

Unfortunately, there is separate data format for each IBR-2 instrument up to now. So, the Nexus standard [5] for storing measurement results were not chosen because our users do not interested in it.

In this situation we decided to our own format as common internal format for all instruments. It comprises two files, one of which contains the actual spectra and the second - the Varman database snapshot. This image contains all measurement parameters registered in the Sonix+. Finally data are transferred to format specific to each spectrometer.

We have proposed and implemented a universal approach to GUI [6]. This means that the same set of clients are sufficient to satisfy requirements of an arbitrary instrument without change.

3. Sonix+ structure

The structure is presented in the figure 1. In a modular system every component (module) is responsible for some device or function. In practice, a set of modules for each instrument has a non-linear structure. Some modules serve hardware devices – they are at a lower level of the hierarchy. The other modules are used to maintain work of the others. We have chosen these components and tried to create them as universally as we can expect. We called these modules servers. If one has to port Sonix+ to another instrument, servers can be used without redesign.

Of course, for communication of servers with other modules specialized protocols are needed. Actually the following protocols have been established:

- universal protocol for device control and inquiring;
- DAQ protocol;
- motor protocol;
- remote inquiring and control protocol via sockets.

All of these were proved by the long practice with the exception of motor server. It will be eliminated in the nearest future.

4. Variable manager (Varman)

During the experiment, all modules communicate through the special storage. It is called Varman database. This storage provides very fast access to parameters. At every moment the contents of the Varman database completely represents the current state of the measurement process at the instrument with the exceptions of spectra.

The idea and initial soft were taken from IRI TU Delft [2]. Redesign for the Windows was made by V.Yudin [7]. Initially all communications were made through simple variables like "motor position", etc. Now operations mostly deal with "structures" of variables like "device status" which consist of significantly complex structures. So some initially implemented Varman features like "interests" - reaction on appropriate events - are now out of use. Another feature which seemed very important for Varman authors - differentiation access to variables for various user categories - did not even implemented in PC version. It lost practical significance at this moment. Unfortunately, another problem - protection of the control computer against external attacks - is quite more urgent.

5. Script utilization

The Python language is used as a script language in the Sonix+ and all nice features and packages of Python are available to control experiments and for preliminary data processing.

This allows one to reasonably separate the specificity of instrument measurements into the most flexible and easy-to-change component. This fact is very useful from the viewpoint of unification and for facilitating the transfer of Sonix+ to new instruments.

It is also important that using the Python as a script language opens a wide door for the user to program experiments. Unfortunately, during these years there were only two user's attempts to take this opportunity.

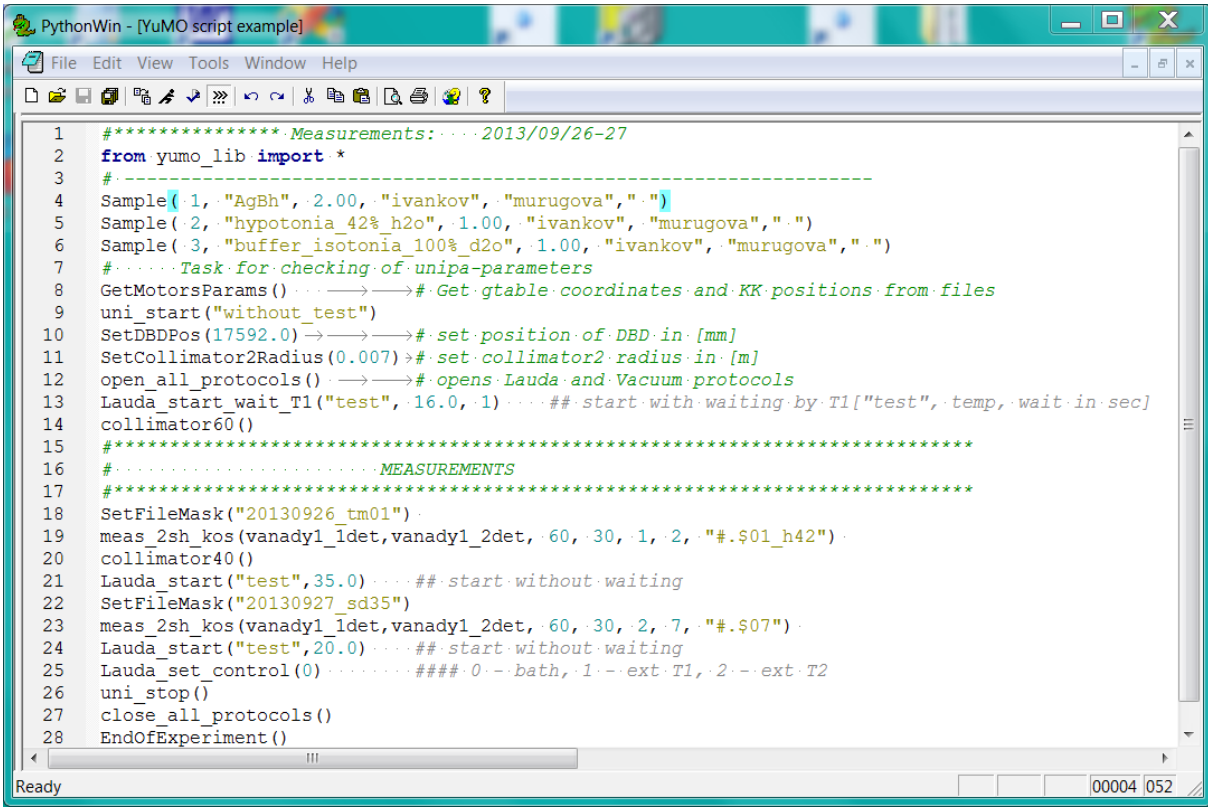

Figure 2: The YuMO instrument script example

The Sonix+ script is the pure Python code. To control the script interpretation a special module is developed. As in other cases, some possibilities inherent in it were not in demand in practice by users. With instruments evolution and to satisfy new users requests actual script is becoming more and more difficult. To simplify user's life each spectrometer is equipped with a library of standard operations. It provides the ability to validate the script before it execution also.

6. User interfaces

There are generally three kinds of Sonix+ interfaces: command line, GUI and web interface.
Command line propose to enter commands as a Python string using any of standard shell (mostly PyhonWin client).

Sonix+ GUI also has undergone a long evolution from separate window for the each controllers to a single common window. There are universal GUI and some specialized. The universal GUI is available at all instruments. It is organized according to principle – *each sub window is dedicated to one of the main user's needs*. There are three main needs to watch: current state of the instrument, the measurement history (log file), the picture of spectra (spectrum). The fourth need is to control the measurement process. Thus, four programs (windows, widgets) are generally sufficient to conduct an experiment. There are additional programs as well (the Load control panel, the Configuration editor and some else).

Specialized GUI are created by special request, for instance, for instrument tuning. Some specialized interfaces were created for instruments at other centers, if their users are known to be so far from modern computers.

7. GUI components set

The reasons for the development of a specialized set of widgets to control the experiment, as well as the rationale for its specific implementation considered in the work [8]. The set of visual components

(widgets), which facilitates the development of graphical user interface (GUI) to control the measurements on neutron instruments were implemented in PyQt. The set components correspond to the basic functions of the user interface for managing the measurement and visualization of spectra, as well as provide program upload / download for the package components and a number of other functions. Most of these widgets could be used as independent client or as a part of more complex window.

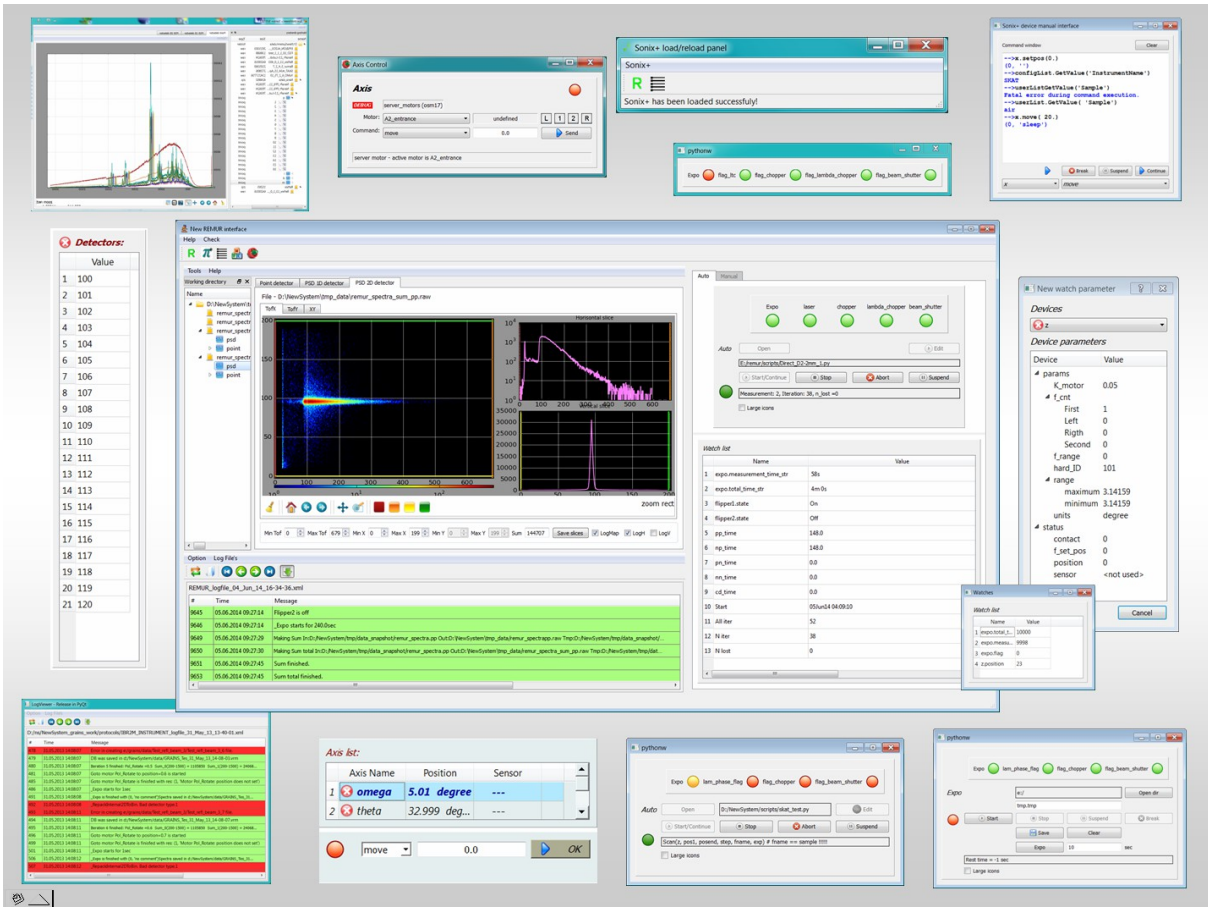

Figure 3: The universal GUI and the widget set.

8. Spectra visualization

Spectra visualization includes a set of widgets for visualization of mono detectors, 1D PSD and 2D PSD data. Spectra can be read from files (including zipped files) or from the DAQ controllers directly. The matplotlib library panel is used to display the graph. Besides curve drawing the panel implements typical operations like scaling, shifting, etc. Additional operations implemented in the widget are:

- linear/log scaling;
- automatic/manual definition of limits in the display window;
- some others concerning dimensionality of the data.

9. Instrument tuning

The tuning program (ICE) is developed for reflectometers of the IBR-2. The ICE program is implemented as an add-on the control complex Sonix + and is designed to adjust the instrument before the main measurement. Program have been written using the PyQt and the graphics library matplotlib.

10. Future plans

The main direction of our future efforts is organization of centralized repository for measurement data with corresponding services.

11. Conclusion

Overall the Sonics/Sonix+ project success is proved by its practice. At the same time, as would be expected, some of the previous decisions have to be revised, in accordance with emerging needs and opportunities.

12. Acknowledgements

For a long period of development and use of Sonix many people in the FLNP and abroad contributed to the project with their work, ideas and criticism. The author is deeply grateful to all of them.

References

[1] http://sonix.jinr.ru/wiki/doku.php?id=en:index
[2] Sckipper M.N. 1997 *Proc. of DANEF'97* E10-97-272 (Dubna) pp 288-294
[3] http://www.nobugsconference.org/Topics:ESS
[4] I.A.Morkovnikov and A.S.Kirilov 2013 Upgrading WebSonix - Remote Instrument Control System Experiment on the IBR-2 Reactor *Proc. of NEC'2013* (Dubna) pp 181-185
[5] http://www.nexusformat.org
[6] A. S. Kirilov, S. Veleshki, S. M. Murashkevich and T. B. Petukhova The unified GUI for neutron instrument control based on PyQt 2013 *Proc. of NEC'2013* (Dubna) pp 158–160
[7] A. S. Kirilov and V. E. Yudin 2003 The Implementation of realtime database for control of the experiment in the MS Windows environment *Prep. of JINR* (Dubna) R13-2003-11
[8] Kirilov A. S. and Petukhova T. B. The Set of Components to Create GUI for Neutron Instrument Control Systems on the Base of PyQt *Comm. of JINR* (Dubna) P10-2015-38