THE IMPLEMENTATION OF THE SOFTWARE FRAMEWORK IN J-PARC/MLF

T. Nakatani, Y. Inamura, T. Ito, S. Harjo, R. Kajimoto, M. Arai, JAEA/J-PARC, Tokai, Ibaraki, Japan
T. Ohhara, H. Nakagawa, JAEA, Tokai, Ibaraki, Japan
T. Aoyagi, JAEA, Taito-ku, Tokyo, Japan
T. Otomo, J. Suzuki, T. Morishima, S. Muto, R. Kadono, S. Torii, Y. Yasu, KEK, Tsukuba, Ibaraki, Japan
T. Hosoya, M. Yonemura, Ibaraki university, Hitachi, Ibaraki, Japan

Abstract

To perform neutron scattering experiments efficiently, it is necessary to use many kinds of software components such as data acquisition (DAQ), equipment control, analysis and visualization with ease-of-use and high throughput environments. At accelerator-driven neutron facilities with MW-class proton power, such as J-PARC (Japan Proton Accelerator Research Complex) and SNS (Spallation Neutron Source), data sizes are varied from several hundred MB to several ten GB per hour, depending on beamlines. Therefore, the software is necessarily scalable and flexible for the various experiments and the enormous data. In J-PARC/MLF (Materials and Life science experimental Facility), a common software framework has been constructed and then many software components are running on this software framework. Our software framework is based on Python and processes network distributions with XML (eXtensible Markup Language) messages over HTTP (Hyper Text Transport Protocol). This software framework enables experimental users to seamlessly perform neutron experiments and analyze the acquired data, as well as instrument scientists to coordinate their instruments and manage the configurations.

INTRODUCTION

In J-PARC/MLF, the neutron experimental instruments which have groups of detectors with thousands of pixels are measuring the neutron signals while changing scanning conditions of the several equipments such as sample environment devices and beamline components. The data size on each scanning condition is quite possible to be more than GB. Because J-PARC/MLF is a high intensity pulsed neutron source, the measurement at one scanning condition can be performed in very short time (less than 1min.). Users need to optimize the condition in a short time by taking account of the result of previous or current data analysis. Thus, the data processing must be scalable.

There are 23 neutron beamlines in J-PARC/MLF. Various kinds of instruments are in use, being commissioned, constructed and proposed. Each instrument is used in several disciplines with various experimental setups and various types of software. However, the human resources to construct and maintain the computing environment of the instrument are limited. Therefore, we have decided to construct the software on a common and flexible software framework.

The whole information of the instruments should be recorded for maintenance. The parameters of the measurement under various scanning conditions and the analysis results associated with the measured raw data are also recorded as experimental meta-data in a database. The results of the simulation such as a first-principle calculation are also recorded in it. The experimental users can make use of this information before their experiments.

The number of experimental users who visit J-PARC/MLF is expected to be about ten thousands per year. Majority of the users are not professionals in neutron experiments. Hence, the system of the instrument should be user-friendly. In addition, since the raw data size is enormous (several tens GB), it is very difficult to bring the data back to their home laboratories. They often access the MLF computing environment and analyze the acquired raw data from their laboratories.

On the basis of these, we have been constructing the software framework that meets the requirements shown below.

- Scalability: high throughput for the large scale data of gigabyte order.
- Flexibility: adaptability of various experimental purposes and reduction of resources for development and maintenance.
- Database: logging the whole experimental information and practical use of previous experimental results.
- Usability: user-friendly system for a lot of users considering remote access.

MLF COMPUTING ENVIRONMENT

Figure 1 shows the diagram of the MLF computing environment. The computing environment has many software components distributed in the network. The software framework is responsible for the communication between components with XML and input/output to data storage including database.

One of the components, a user interface (UI) component called “Working Desktop” (WD), has been developed based on an object oriented script language, Python. Users can perform their experiment and data

analysis by sending Python commands through the component. In another words, users can seamlessly operate experimental controlling, data analysis and visualization with one Python script. So far, basic commands were implemented. Graphical user interface (GUI) of WD component is under development. We have been constructing several components in WD.

The experimental controlling component consists of controlling DAQ hardware and other equipments such as sample environment devices and beamline components. We introduced "DAQ-Middleware" [1] as the standard DAQ software. Since DAQ-Middleware recognizes and absorbs differences between detectors, such as ^{3}He gas detector, scintillator detector and so on, commands on WD are same for every kind of detectors.

The analysis component called "Manyo-lib" [2] is a C++ library wrapped in Python by SWIG (Simplified Wrapper and Interface Generator). Manyo-lib provides the standard data structure called "Element Container" based on STL (Standard Template Library). Users can process the data by using built-in functions of Manyo-lib (for mainly data reduction) on Python command line and also add new functions written in Python or C++ for their own scientific analysis. Manyo-lib is mainly responsible for heavy task like the conversion function of histograms form event data. Common and established functions written in Python and other legacy language will be implemented in Manyo-lib.

The developed visualization component written by wxPython and matplotlib visualizes the 2D/3D data processed via Many-lib.

The database component is now being developed. It will be an XML-typed database system. It logs and monitors the whole data in the MLF computing environment linking to the J-PARC proposal submission system managed by the J-PARC Users Office. The components of simulation and remote access are under discussion.

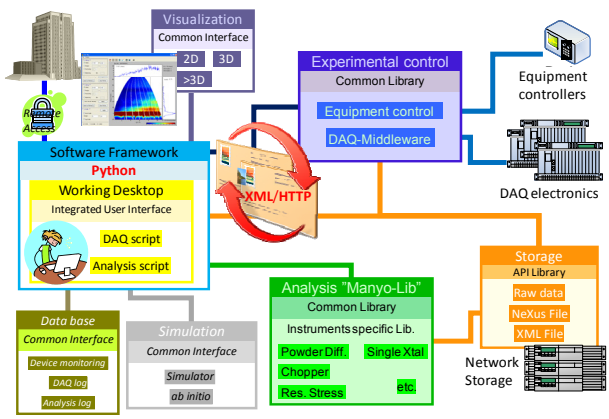

Figure 1: The diagram of the MLF computing environment.

DETAILS OF THE SOFTWARE FRAMEWORK

Server-Client model was adopted to realize a network-based framework: WD is the client and it sends requests to servers. The software framework consists of following types of software written in Python:

- Working Desktop: Integrated user interface and the client in the network
- Instrument management server: Measurement state machine of hardware control
- Analysis control server: Analysis and visualization state machine.

The communication protocol between WD and the other components is the exchange of XML messages over HTTP. Table 1 shows examples of the relation between the Python commands and XML messages over HTTP.

Table 1: Examples of Relation Between Python Commands and XML Messages over HTTP

Command	URI	Request	BODY
SE.params(Cond1)	/sample/params	POST	<params> Cond1 </params>
DAQ.begin()	/daq/begin	POST	
DAQ.status()	/daq/status	GET	

Working Desktop

Users can execute these Python-base functions from a character user interface (CUI) and a GUI of WD. The CUI is an execution environment using the command line of Python, and the GUI is an execution environment constructed by wxPython which is one of the user-friendly GUI tool kits for Python. The common GUIs such as the status display, the launcher, the sequencer and the authentication are developed. The status display always displays the operation status of the instrument, the status of the DAQ system, each status of the equipments and the neutron source status of J-PARC/MLF. The launcher controls the executions of software for measurement, analysis, visualization and the maintenance, and also operates the DAQ system directly. The sequencer helps to make a Python sequence with Python functions and the parameter input. The sequencer also supports execution of the sequence and monitoring its progress on the GUI. The authentication specifies who uses the instrument and achieves the exclusive control with the ticket.

Instrument Management Server (IMS)

The instrument management server (IMS) executes automatic measurements combining DAQ hardware and several equipments according to measurement conditions and operation control scripts. The measurement conditions consisting of the initial and scanning conditions are written in a form of XML, and the operation control scripts are written in Python according

to the guideline of the software framework. The operation control script is executed as a background task. The scanning parameters of each condition are recorded in a form of XML.

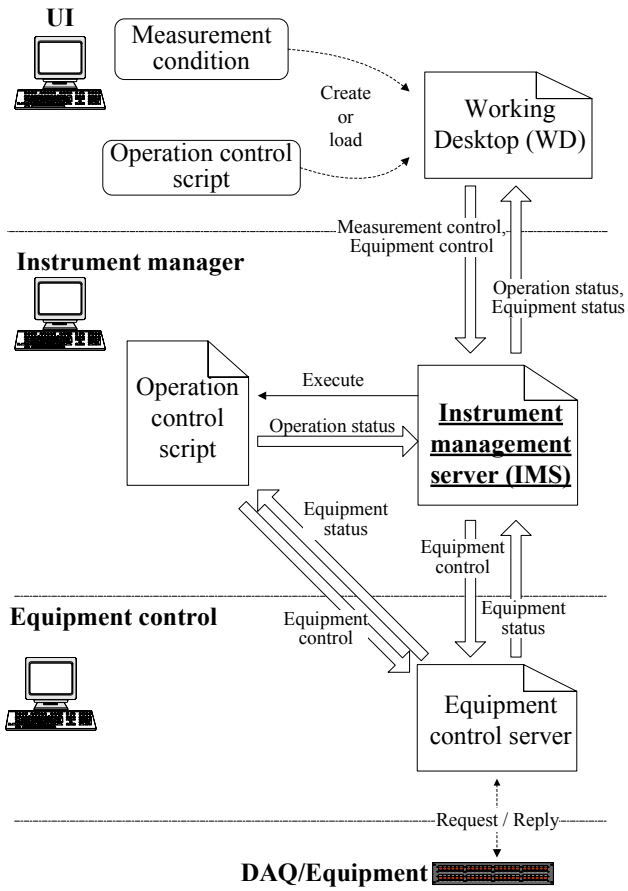

Figure 2: Measurement procedure.

The measurement is performed according to the following procedure: (figure 2)

1. WD: Create or load the measurement condition and the operation control script.
2. WD: Send the measurement condition and the operation control script to IMS to start measurement.
3. IMS: According to the operation control script, threads are invoked and requests are sent to DAQ and each equipment control.
4. IMS: Monitor operation status and each equipment status.

Analysis Control Server (ACS)

The analysis control server (ACS) executes analysis and visualization according to analysis conditions given from WD. The analysis conditions consisting of the information such as required analysis modules and the execution sequence are written in a form of XML. We can describe the procedure of the calculation of parallel / serial or synchronous / asynchronous as an analysis condition. Analysis parameters given to analysis modules are recorded in a form of XML. The ACS has not only the network processing mode but also the stand alone processing mode. The users can execute the same analysis in their home laboratories by using the stand alone mode.

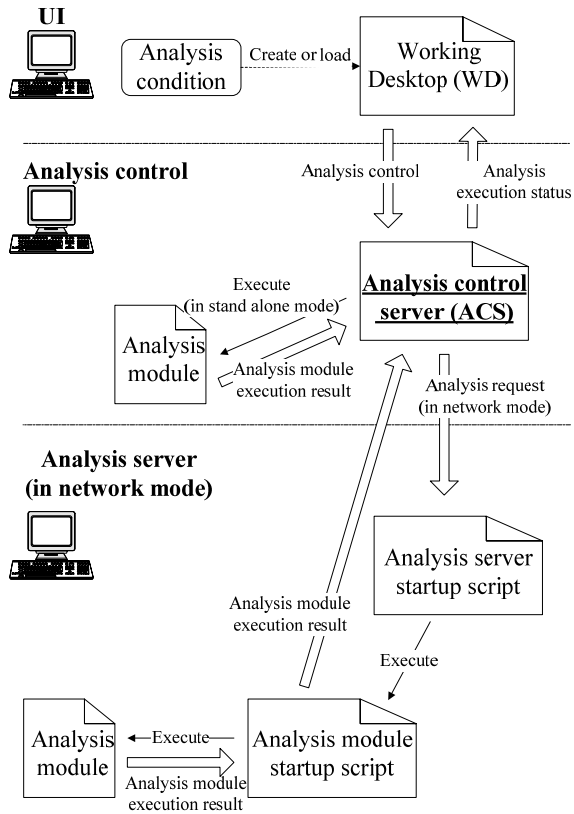

Figure 3: Analysis procedure.

The analysis is performed according to the following procedure: (figure 3)

1. WD: Create or load the analysis condition.
2. WD: Send the analysis condition to ACS to start analysis.
3. ACS: Execute analysis modules inside in the stand alone mode, or send an analysis module startup script to analysis server and execute analysis modules by analysis module startup script at the analysis server side in the network mode.
4. ACS: Monitor analysis execution status.

SUMMARY

We have developed the software framework for DAQ, equipment control, analysis and visualization of the neutron experimental instruments in J-PARC/MLF. The software framework is scalable and flexible by Python and the distributed network processing with XML/HTTP.

REFERENCES

[1] K. Nakayoshi, Y. Yasu, E. Inoue, H. Sendai, M. Tanaka, S. Satoh, S. Muto, N. Kaneko, T. Otomo, T. Nakatani, and T. Uchida, Nucl. Instrum. Methods Phys. Res., Sect. A 600 (2009) 173.

[2] J. Suzuki, T. Nakatani, T. Ohhara, Y. Inamura, M. Yonemura, T. Morishima, T. Aoyagi, A. Manabe and T. Otomo, Nucl. Instrum. Methods Phys. Res., Sect. A 600 (2009) 123.