Computing environment design at J-PARC/MLF

Jiro SUZUKI[a], Toshiya OTOMO[a], Michihiro FURUSAKA[a], Masatoshi ARAI[a], Shun SUZUKI[a], Atushi MANABE[a], Setsuya KAWABATA[a], Narutoshi KAWAMURA[a], Masao YONEMURA[a], Kenji NAKAJIMA[b], Takeshi NAKATANI[b], Shuki TORII, Ryoichi KAJIMOTO[b], Takashi OHARA[b], Tetsuo AOYAGI[b], Yoshitomo UNO[b]

[a]High Energy Accelerator Research Organization

[b]Japan Atomic Energy Research Institute

Materials and Life science Facility (MLF) of Japan Proton Accelerator Research Complex (J-PARC) will provide one of the highest intensity pulsed neutron and muon beams. The construction will finish at the end of 2007. Because it is expected that gigabyte order time-of-flight data will be produced per one experiment, system developments and engineerings of computers are crucial as well as instrument hardware designs to make really good use of all the data. At first, we defined very simple components that should work correctively to realize the computing environment for MLF; experiment, analysis, simulation, database, user interface (we call this as working desktop), graphic, network (collaboratory), and security components. The working desktop will interconnect between each component. This environment will provide common functionalities for the neutron and muon experiments. Instrument specific functions are expected to be developed and connected through the working desktop component. The development of the analysis component, a framework, is progressing with C++ and it was named as Manyo-lib. A systematic approach based on the object-oriented methodology was taken to develop a software framework, in which users could easily develop and execute their analysis software in the neutron scattering experiments. The framework has common and generic analysis functionalities for the neutron scattering experiments. The first version of the framework was applied to the analysis of a real experimental data and was found to be a convenient environment for setting up and running data analysis software for the neutron scattering instruments.

1. Introduction

Japan Proton Accelerator Research Complex (J-PARC), is a new proton accelerator complex, and frontier science in particle physics, materials- and life-science, and nuclear technology will be performed. J-PARC, the joint project between High Energy Accelerator Organization (KEK) and Japan Atomic Energy Research Institute (JAERI), is under construction at Tokai campus of JAERI and its construction will be finished in 2007. The Material and Life Science Facility (MLF) is a user facility providing neutron and muon sources for experiments in J-PARC. Twenty-three instruments for neutron scattering experiments will be installed in MLF. Averaged user activity will be 100 users a day, and the time period of each experiment will be 0.5-2 days a group. The instruments and its data analysis environments should be fully provided by the facility side.

Our basic concept is to provide a framework that has common and generic analysis functionalities for the neutron scattering experiments, and the framework becomes a standard analysis environment for the data analysis in MLF. The merit of object-oriented approach helps us maintain and improve the analysis software for a long term. Since the framework will work on many instruments, the framework should be available on various operating systems; Linux, MacOS-X, FreeBSD, AIX and Solaris. The framework was named as "Manyo-Lib".

The analysis framework comprises a C++ framework, network distributed data processing modules and user interface modules. The C++ framework was designed so as to enable to handle large scale data in high-performance, and primarily consists of C++ class library. The base-classes are designed to construct analysis operators in the class library. The generic and common analysis operators for the neutron scattering experiments were developed as derived-classes of the base-classes. User interfaces and the network distributed data processing module were designed on Python. The interface between C++ and Python is created by SWIG[1]. The SWIG, an interface compiler between C++ and script languages, does not require to modify the C++ source code. The interface combines the high-performance applications written in C++ with the Python environment. A number of analysis tools have been developed on Python, and some TCP/IP connection tools have been provided as Python standard library. The Python environment helps users apply the framework to their applications without compiling the framework.

2. Design of the Analysis Framework

2.1. Data Container

Design of data structure is one of key issues. Simple and efficient data containers are required, because their designs determine the system performance. Two types of class templates are provided in the framework using the Standard Template Library (STL), in terms of which various types of experimental data objects can be stored hierarchically.

The first type is a basic container for simple and generic purpose and consists of a set of sub-containers and a header object container. The sub-container can store any type of data-object. The number of sub-containers can be changed at any time, and the information about the set of sub-containers can be installed into the header object. Each information has the name-tag to manage its meaning and property. The header object works like a note-pad and dictionary. The second type of container consists of a header object container, a set of sub-containers, and each sub-container's name-tags. Each sub-container has the name-tag, and can be extracted by using its name-tag. This design is very useful to store many number of data objects into this container. The capacity of data container and its hierarchically structure is scalable.

2.2. Analysis Functions

The analysis package is divided into six class categories. System and Calculation categories provide basic tools of analysis package. Correction, Projection, Merge and Analyzer categories provide common and generic analysis functionalities for the neutron scattering experiments, each of which uses System and Calculation categories.

Fig.1 shows the design concept of an analysis operator. The calculation module cor-

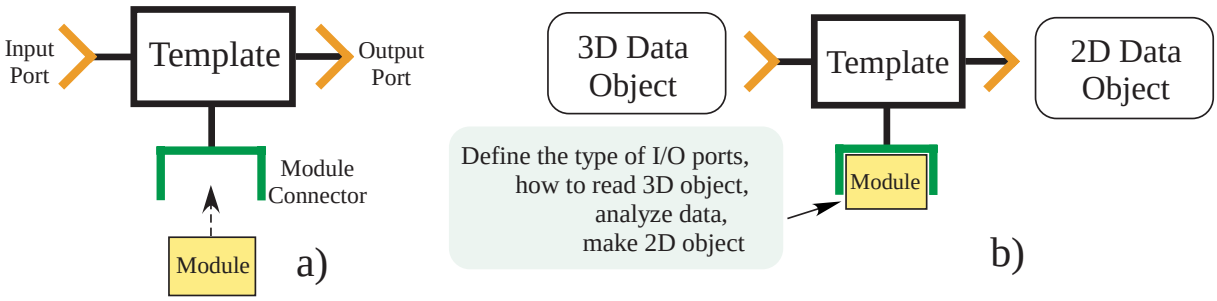

Figure 1. a) A schematic structure of analysis operator. The template is a base class of module, and consists of input and output ports and a module connector. The module describes the analysis procedures for each use case. b) An example of the basic and simple use case of an analysis operator. An analysis functionality is organized by using relationship of inheritances.

responds to each use case. Generic and common functionalities, including data flow, for the neutron scattering experiments are provided in the framework, while the calculation modules specific to user instruments should be prepared by the user side. The tools for converting the data objects from the common formats of the framework to the NeXus[2] format will be prepared. NeXus is one of the common data formats for the exchanging

of neutron muon and synchrotron scattering data among facilities and user institutions. The NeXus format uses the Hierarchical Data Format (HDF) which is portable, binary, extensible and self-describing.

2.3. Network distributed data processing environment

Fig.2 shows a general design of the data analysis environment for the neutron scattering. It can be applied to on-line and off-line data analyses.

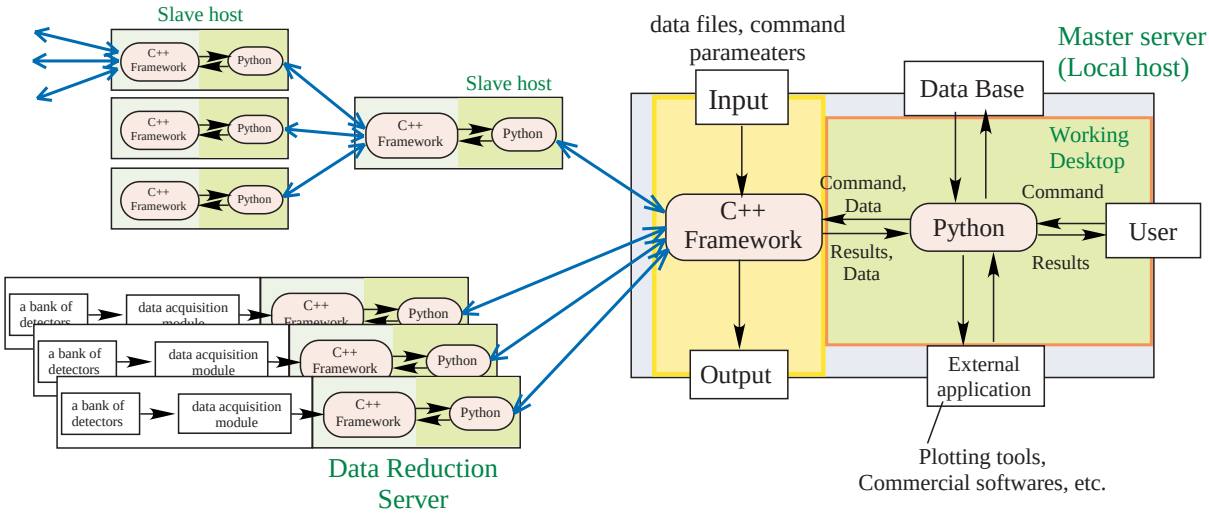

Figure 2. The diagram of the framework. The C++ framework on the data reduction servers, the slave hosts and the master server are the same source code, and their Python wrappers apply the C++ framework to each use case. The wrapper is the interface exchanging command and data through the network between the hosts.

2.3.1. Data Reduction Server

Hundreds of detectors are arranged into a detector bank. Each detector bank is connected to a data reduction server (DRS) which consists of data acquisition, reduction and analysis modules. The data, just collected by the data acquisition module, should be corrected to real physical values. These corrections relevant to detectors are done in the analysis modules. The corrected datum are gathered and merged into a small size data in the data reduction module. The C++ data objects produced on the DRS are in the common format of the framework. The objects are sent to the master server through the network.

2.3.2. Master Server

As shown in Fig.2, the master server consists of the C++ framework with the Python wrapper and the user interface. The character-base user interface have been developed on Python. Python scripts, to be executed on the DRS and the slave hosts, are generated on the master server, are sent through the network to the DRS and the slave hosts, and then the master server receives its results. The scripts are executed as parallel processing on the DRSs and the slave hosts, while their results are collected and analyzed on the master server.

2.3.3. Slave Host

The slave host has only the C++ framework with the Python wrapper. It can be worked as a calculation server. This server receives the Python scripts from the other hosts, executes them in high-performance, and returns the results. Since the C++ framework and the wrapper on the slave hosts are identical to those on the master server, a slave host can govern the other slave hosts. The capacity of data processing on this framework is scalable.

3. An Application to the Experiment

A application software was developed based on the framework, so as to check the satisfaction of the requirements. The application software is a data analysis software on the framework for the Small/Wide Angle Neutron Scattering Instrument (SWAN) at Neutron Scattering Facility of KEK (KENS). SWAN uses the wide wavelength range of neutrons (0.05 - 1.1 nm), and three types of neutron detectors with different efficiencies and geometries were installed[3,4]. Neutron scattering from a sample is measured by the time-of-flight (TOF) method. Fig.3 is the schematic analysis flow diagram for SWAN. The scattering intensities observed by each detector bank were formatted into 3d (three dimensional) histogram objects in the data-formatter. The intensities at the same geometrical condition were averaged in the projection with the projection module which should be prepared for each use case. Neutron wavelength dependent factors like detector efficiencies, incident neutron flux, sample transmission were corrected for each averaged intensity in the correction. Histogram objects of the neutron cross section of sample versus momentum transfer were obtained in the analyzer. The formatter, projection and correction run on the DRSs assigned to each detector bank. The analyzer and the user-interface run on the master server.

The application software was developed and executed on the Python environment. The application is executed on a master and slave servers whose operating system are Linux, FreeBSD and MacOS-X. The applications are working very well.

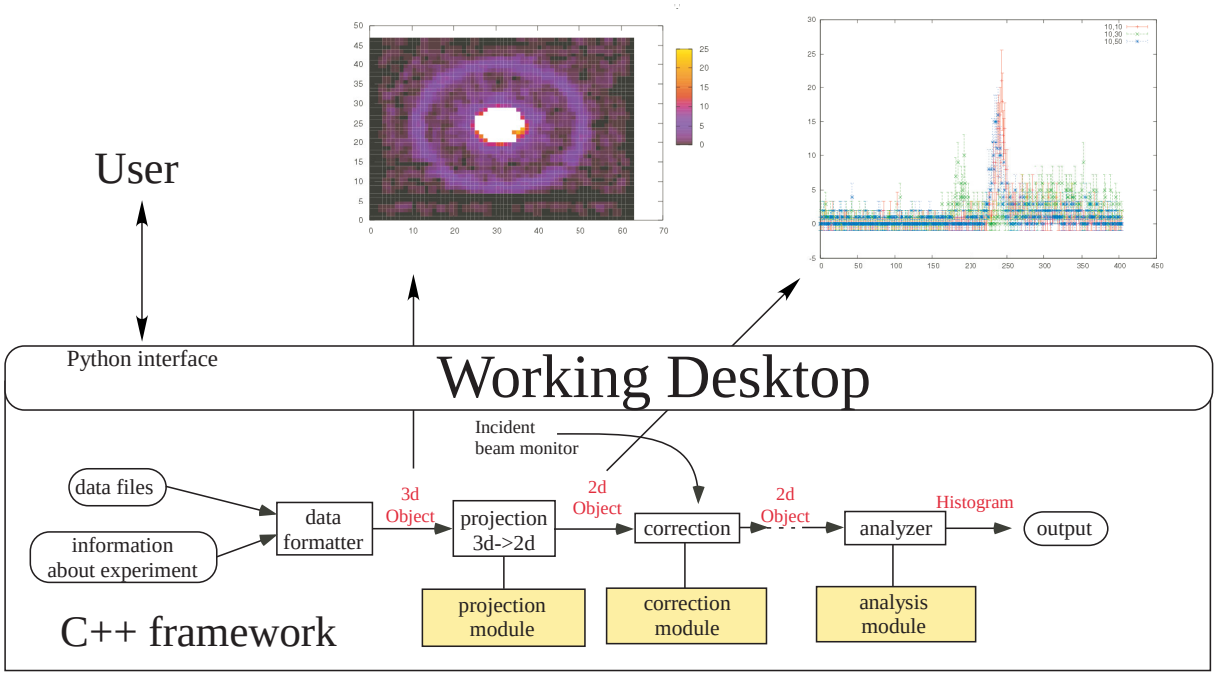

Figure 3. Flow diagram of application software. This system is executed on a master server and two slave hosts. 3d Object, 2d Object and Histogram are three dimensional histogram object, two dimensional histogram object and one dimensional histogram object, respectively.

4. Conclusion

An analysis framework for the neutron scattering experiments based on Object-Oriented approach has been developed. The framework, "Manyo-Lib", provides common and generic data analysis functionalities for all the neutron scattering instruments. We applied the framework to the analysis-software and confirmed its functionalities and stability. The framework works in high-performance with ease, since the analysis functions are called from the Python environment but run in the C++ environment. Because the structure of the network distributed data processing environment is hierarchically, the capacity of data processing in the framework is scalable. The Object-Oriented methodology helps us develop and share the data analysis environment for the neutron scattering and its data for a long time. The current version of the framework and its manuals are available from the web site[5].

REFERENCES

1. http://www.swig.org/
2. http://www.neutron.anl.gov/nexus/
3. M. Furusaka, Proceedings of ICANS-XI (International Collaboration on Advanced Neutron Source), Tsukuba, (1990) 667.
4. T. Otomo, M. Furusaka, S. Satoh, S. Itoh, T. Adachi, S. Shimizu and M. Takeda, J. Phys. Chem. Solids, 60 (1999) 1579.
5. http://research.kek.jp/people/jisuzuki/.