THE DEVELOPMENT PLAN OF HIGH LEVEL APPLICATIONS FOR CSNS

Q.Gan, C.H.Wang, IHEP, P.O.Box 918, 100049, Beijing, China
P.Chu, SLAC, Menlo Park, CA 94025, USA

Abstract

This paper surveys the recent developments in high level applications. Especially, the XAL framework and its new progress in LCLS is studied. The design philosophy and requirement analysis of CSNS high level applications are also discussed.

INTRODUCTION

The accelerator application is of great importance in China Spallation Neutron Source (CSNS) project. It is not only the essential tool for the accelerator design and commissioning, but also a bridge of software between control system and database.

OVERVIEW

Currently, many high-energy physics laboratories have developed their own accelerator applications by different programming languages and integrated development environments (IDE). These applications are used to design accelerator lattice, measure devices parameters, commissioning, monitor the accelerator, and so on. The following table lists some popular accelerator applications and their maintenance labs.

Table 1: Some Accelerator Applications List

Application	LAB	Language
APPiX	Fermilab	Java
AT	Spear, ALS	MatLab, DLL
CDEV	JLab	C++
CSS	DESY, SNS	Java
ELEGANT	APS, World	C
MAD	World	C, F77, F90
PLACET	CERN, World	C
RACETRACK	Elettra	F77
SCP	SLAC	Matlab, DLL
SAD	KEK	C++, Java
SCP	SLAC	Matlab, DLL
SDDS	APS	
SEAL	LCLS	Java
UAL	RHIC	C++
XAL	SNS	Java

Compare with other programming languages, the Java language is more suitable for network and can be run on different platforms. Now more and more labs are preparing or doing the migration of their software to Java platform. But software migration costs lots of material resources, and some old programs are difficult to migrate to the new platform because of the outdated design modes. So, it is a good way for the labs' development to choose advanced languages and tools.

HIGH LEVEL APPLICATIONS FOR CSNS

The design principles for the CSNS accelerator application are:

- Advanced. As the rapid development of computer software, the CSNS accelerator physical software must keep pace with the increasing application demand of high-energy physics laboratory.
- Stable. As a facility of the large scientific instrument, it must have a high reliability to assure the formal operation. Meanwhile, it should be designed to control the beam lost so that the radioactive production will not affect the normal experiment and maintenance.
- Proper cost. Constructing and operating a large research facility is usually under tight budget. Therefore, the cost of software should be as low as possible. Hence it should be considered to immigrate some mature accelerator software into this platform.
- Potential to be improved. It is possible to improve the software by continuous updates.

Software Development Model

Software development model is the framework of all the process, function and mission of software developing, which includes different phases, such as requirements analysis, design, implementation, testing, integration, and maintenance.

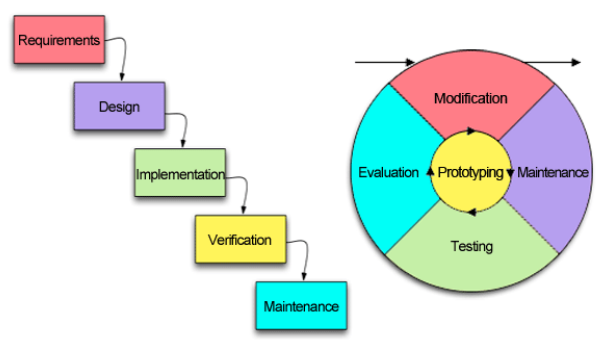

Figure 1. The waterfall model and the prototyping model

It's a clear way to express the total process of software developing. It transparently describes the target function and mission as foundation of software project working. Many accelerator applications adopted waterfall model, in which development is seen as flowing steadily

downwards through the phases of development process. But it's difficult to give precise requirements of the CSNS accelerator in the early phase. Therefore, the prototyping model will be used for CSNS accelerator applications.

Figure 1 shows the flowchart of the waterfall model and the prototyping model. Software prototyping is the process of creating an incomplete model of the future full-featured software program, which can be used to let the users have a first idea of the completed program or allow the clients to evaluate the program [1].

Software Prototyping and Requirement

For the similarities between the CSNS and SNS, the software framework used at SNS, XAL is a natural choice for the CSNS. According to the functions of the XAL tool set, the accelerator application can be divided into accelerator physics applications, commissioning tools, monitor tools, and some related software. However, some codes of XAL must be rewritten and improved since RCS (Rapid Cycling Synchrotron) of the CSNS is different from the accumulated ring of the SNS.

Since XAL is going forward, a new high-level application framework called SEAL (SLAC Eclipse Accelerator Lab) is under design for LCLS. It will implement XAL core into the Eclipse framework as a future version of XAL. So it can be an optional scheme for the CSNS.

Eclipse Environment

Eclipse is a very popular IDE (Integrated Development Environment) of Java. Meanwhile, it also can be used as a development environment for other languages (i.e. C++ and Basic). The greatest advantage of Eclipse is the RCP (Rich Client Platform)/Plug-in developing pattern. The applications which based on RCP/Plug-in show great extensibility. New functions can be added simply only by disposing the plug-ins to the application systems.

If the XAL core is implemented into the Eclipse framework, the XAL accelerator hierarchy will provide object oriented accelerator view while the Eclipse rich client platform will provide professional GUI (Graphical User Interface) experience and better extensions.

On the other hand, Eclipse framework is also used for possible future EPICS version which is called Control System Studio (CSS). With supports from DESY, SNS and many other accelerator laboratories, CSS has its first beta version available for test. Presently, CSS can only be used for general purpose display and simple analysis work. The best approach would be taking XAL physics work and Eclipse plug-in architecture to form a new framework for future applications.

Applications Infrastructure

The application infrastructure of XAL can be rebuilt by the Eclipse plug-in developing pattern. First, split functions that provided by various tools in XAL. Secondary, classify all these functions. Then integrate similar functions to facilitate the implement of plug-in with these functions. At last, according to software requirements, select corresponding plug-in and compose the final application. Doing this will not only avoid the repeated development of codes with same function, but can also customize functions and interfaces flexibly.

Based on the analysis of SEAL application group, the primary function of XAL can be divided as follows [2]:

- Basic function plug-in

1) XAL core as a plug-in
2) General purpose XML parser plug-in
3) Device/PV "tree" -like browse plug-in
4) Probe plug-in
5) "1-D" or "2-D" scan plug-in
6) Data correlation plug-in
7) "Strip-tool" -like plug-in
8) PV connectivity verification plug-in.
9) Online model plug-in
10) General purpose "knob" plug-in

- Application plug-in used in accelerator physics

1) Gaussian (and other profile shape) fit plug-in
2) Wire scanner plug-in
3) Emittance plug-in
4) Image plug-in
5) Trajectory optimization plug-in
6) Beam energy measurement
7) Bunch length measurement

Data Flow

There are many kinds of data required by the high level applications, including real-time data of control system and static data including equipments parameters, installation and survey information, design values of the accelerator physics, and so on.

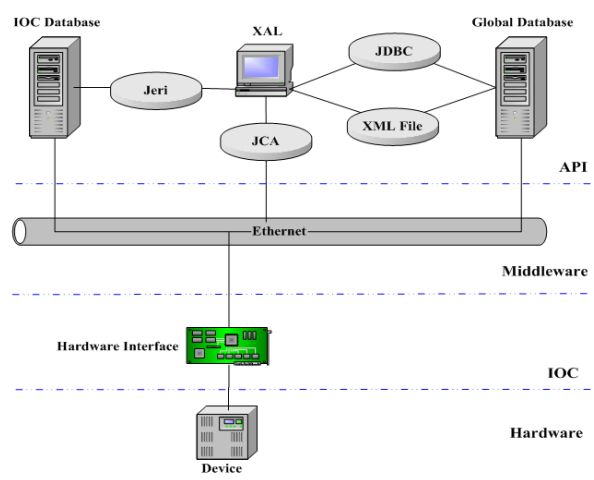

Figure 2. Data flow

The real-time data from EPICS CA (channel access) is accessed by JCA/CAJ [3]. While, the static data can flow via two alternative ways, JDBC and XML file as shown in figure2. Before the database establish, the XML files is recommendatory for applications initialization.

Concurrent Versions System (CVS)

CVS history record can be read with Eclipse. Code written by different programmers is marked with different colours. Furthermore, code which has been made modification will be highlighted in files.

The reconstructing script feature is introduced in Eclipse which can change the source code without changing its behaviour. This feature makes the code modification much easier. In addition, each reconstructing operation is saved to history record. These scripts can be saved to CVS or included in a JAR file, so that the JAR users can receive a new version with according change. This is different from the application patch. Patches can only be used in their specific source operation, while reconstruction scripts can be used in any source code operations to reconstruct API [4].

User Interface

Eclipse can be used to develop AWT, Swing, SWT (Standard Widget Toolkit), and other GUI. SWT is the unique and distinguishing feature of Eclipse which calls the graphic information of operation platform by JNI (Java Native Interface) [5]. Compared with other Java GUI, the user interface of SWT (including appearance and accelerator key) is consistent with the operation system. On the other hand, the SWT application is faster than other application developed by pure Java due to this feature. Therefore, CSNS will make the most of SWT to develop user interface.

However, SWT is not as flexible as Swing or AWT and its graphical functions are relatively simple. AWT and Swing will be used occasionally. The third-party graphics package will be used when necessary.

These GUI programme will be packaged in Eclipse RCP form so that the users who are not familiar with Eclipse can call them separately. Users who are familiar with Eclipse can use the corresponding plug-in or view in Eclipse directly as shown in figure 3.

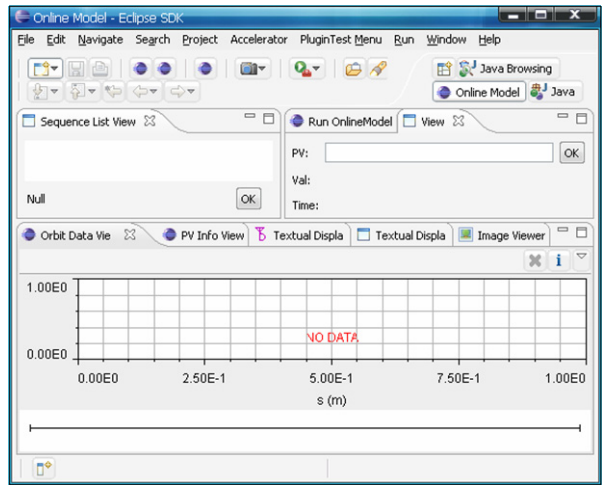

Figure 3. Eclipse/SWT user interface

CONCLUSION

The high level applications for the CSNS will use XAL core and Eclipse environment. The software development needs collaboration with SNS and LCLS. The CSNS will benefit from the collaboration for the high level applications.

REFERENCES

[1] http://en.wikipedia.org/wiki/Software_prototyping

[2] Greg White, Paul Chu, LCLS Online Physics Software Applications Summary, Stanford Linear Accelerator Center, 2007.

[3] J. Galambos, et al, SNS Global Database use in Application, SNS, 2003.

[4] http://www.eclipse.org/articles/

[5] Robert Harris and Rob Warner, The Definitive Guide to SWT and JFACE.