Class Analysis and the Object Model in the KEK PF Linac console

Isamu Abe and Kazuo Nakahara

KEK, National laboratory for High Energy Physics
Oho1-1, Tsukuba-shi, Ibaraki 305, Japan

Abstract

The Linac device controllers (front-end computers) and mini-computers for the KEK PF, which has been operating for more than ten years, had to be renewed during 1993 because there was no longer support from the hardware manufacturers. In 1994, the B-Factory project at KEK was approved and will start from 1998, and this requires the Linac to be upgraded from 2.5GeV to 6GeV. The control system also needs an increase in size for the additional klystrons, magnets, vacuum systems and so on At the same time, the console programs must be modified to be suitable for B-factory operation. In the modification phase, in order to reduce the maintenance and development costs, some of the device controller SBCs (Single-Board-Computers) will be replaced by PLCs (programmable logic controllers). Investigations[1] were launched late in 1994 in order to determine the I/O stability and network possibilities. In this paper, the software for the device layer controller (PLC) and the human interface layer to the B-factory are discussed on the basis of the Object Oriented Concept.

1. Introduction

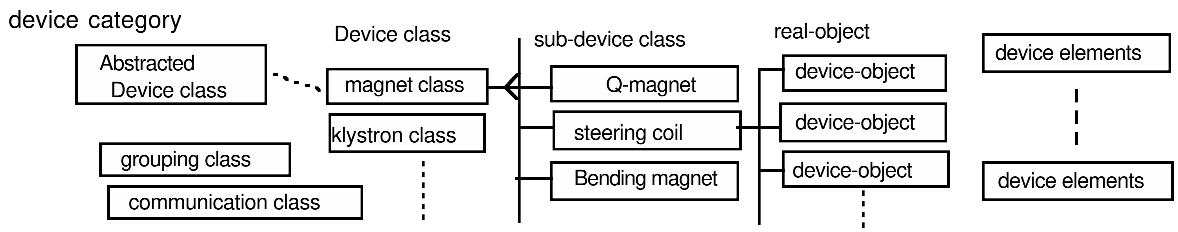
An object-oriented analysis and design[2] of a human interface (operators console system) have been adopted for the PF Linac. The old console system has a graphic display which was written using FBHG (Fujitsu Basic High Grade) on DOS/Windows (Japanese version). It has not been very simple to change the display and its programs, which were written in structured FBHG language. To change the conventional language to a new OOP one requires a big paradigm shift and there are two aspects which are both good and not so good. One is higher productivity with the OOP, although the learning cost for users is also higher. Some disadvantages of the OOP are also becoming clear. We are therefore providing a root, or super class, from which one can derive objects for users (developers / operators) without any programming on the new GUI (graphical user interface) windows. Users can make control and display windows very easily and quickly by just copying a visual object from its mother class, while inheriting methods and properties. After naming the object and changing some properties if necessary, it can be run as a flexible user program on the Linac console.

2. Classes in an accelerator domain

For an entire accelerator domain, a rough object-oriented analysis was made,and then the control-system design proceeded step-by-step based on an OOA (analysis) . The OOA and OOD (design) were repeated several times for each field in the accelerator domain. As a result, we divided the accelerator control system into to two major categories: 1) device category and 2) generic category. The generic category can be divided into classes: the data and information process class, the generic tasks class, the beam physics class, and the GUI class [3]. In this paper the device class and GUI class are mainly discussed.

3. Device class

The Linac is composed of many devices, such as magnets (beam transport), klystrons (RF system), injection system, beam monitors (Beam Position Monitor, Core Monitor, Profile Monitor, etc.), vacuum system, trigger system, control system and safety system, which are located along its length. These grouped systems, which have physical-device objects, belong to the device category. The device category has several root classes and/or super classes. Generally, it has subsystems as a child class. In most cases, the super class has a definite behavior, properties and messages to be handled independently. When there are many similar device objects, they can sometimes be classified in the same group in a bottom-up approach. Each object is derived from its mother class. The magnet and the power supply, which are geographically located along the beam line, are typical examples. Once they are well defined, these classes and objects do not have to be changed very often; it is only necessary to install or remove objects when the Linac is physically modified. We may find a common or standard class which might be shareable in several laboratories if it is well analyzed. In 1989, the device and access procedures in an accelerator domain were well analyzed and defined at CERN as reported [4] at ICALEPCS'91. The CERN device protocols were installed in the device-layer computers. In our system, we will attempt to analyze objects in the near future, so as to distribute them on the two layers of the computer system through a network, using the emergent CORBA, OLE or something similar. The root or super class, sub-class, and device element relations are shown below:



4. Generic category

4.1) Data and information class

The data and information (with commands) exchanged between the devices or equipment of the accelerator are defined as objects separate from device objects. Data which is sent to or received from device classes are supposed to be processed in some way at the operator console or by a data-base system or generic task class. Normally, data must be processed in a way that is already known, or decided by an expert based on experience or knowledge. As an example, magnet and vacuum data are commonly displayed as real-time data or a history graph at the operator's console, and then sent to an upper/lower level check filter for the alarm system. Since it is already known how to process most of the accelerator data, we can make data-process classes rather easily abstracted as re-useable ones so that data can be collected as encapsulated objects in a data base.

4.2) Generic task class

There are no actual devices in the generic task class; it is a type of heuristic knowledge class: how to operate an accelerator, or how to run each device. There are many ways to run the Linac and each person can have his own methods to handle the devices or the accelerator to obtain beam. Although it is difficult to find a standard or common class in this generic task, a typical procedure must be defined for the actual operation of the Linac. There are very few derived sub-classes in this category. Each accelerator must have its own class for every operation or diagnosis; there will then be a few shareable common classes among the different laboratories concerning generic tasks.

4.3) Beam physics class

Mathematical models are commonly used for beam dynamics calculations and their handling on the networked computers. Some people use OOP methods for modeling, while others do not. Since these are carried out at the accelerator-design phase, they are not considered here.

4.4) GUI class

In the GUI class, the object-oriented approach has been a most effective way to make an easy-to-use and flexible graphical user interface on Windows. The accelerator GUI requires some graphical icons, which are for abstract devices on the Windows display. The graphical icon has derived properties and methods based on its mother class. Two layers of the mother and child classes represent a more convenient way in actual operations or in the development phase of the GUI. A deeper class should be related to the mother class, behind which it does not appear on the GUI. Simple changes in the properties and methods are easy to make in the property lists or windows. There are necessary GUI elements, which should be abstracted objects or those encapsulated for the Linac operator:

1) Device_Icon, Operational_sw/lamp_Icon,
2) History_graph, Two-dimensional_graph, Charts
3) Help_menu(www viewer), Text_display,
4) Video_display,

These Icon or GUI elements must be a re-useable mother class which can inherit many useful properties. Some commercial software packages exist which could match our needs.

5. Device layer controller

The magnet and vacuum systems used to be controlled by SBC (single board computer) local controllers which were connected to CAMAC. CAMAC had been running until it was replaced by the VME system. Since we have tried to reduce maintenance costs, the replacement of the old SBC with a PLC was investigated. Although the PLC was originally not intended for a high-speed multi-purpose machine interface, being used in industry as well as other fields, the DAC and ADC modules are sufficiently stable for use with power supplies. The performance of the PLC is almost good enough to control not only the magnet power supplies, but also the vacuum systems and klystron modulators as a low-speed control. The cost is less than that of other interfaces, such as VME or CAMAC, and maintenance is easier. Since most of the PLCs can be connected to Ethernet, it is now possible for them to be controlled from host computers, even though each maker has its own network. The PLC which we have selected has two CPUs: one is for I/O and the other is for communications. TCP/IP and UDP are available protocols which are used to communicate with other computers on Ethernet. The PLC has its own internal programming and tools, and acts as an intelligent controller using ladder logic. Even from Windows 3.1 with a Pentium 90MHz CPU, a 10 to 20 per second refresh rate is possible with the PLC. That is adequate scan speed for an operator when he adjusts the magnet current or other variable.

6. PLC class

PLCs are produced by several companies and there are no standards among the manufacturers. A PLC can support various types of I/O modules: such as DI/O, DAC, ADC and IRQ. The PLC can be a member of the super class at a device-layer computer and the I/O modules are defined as belonging to the sub-class. The PLC super class must communicate with the PLC communication class or friend classes. In the PF Linac, the abstracted device classes are also set on the human-interface layer so the PLC communication class could communicate with the PLC device objects which are distributed at the device layer through the network.

7. Implementation

The PLC used for the Linac device controller will be connected to about 400 sets of magnets, 100 sets of vacuum systems and 60 sets of klystrons, which are slow devices to control. In late 1995, a prototype will run the vacuum system at the 2nd sector of the PF Linac. In advance of the hardware system, the first version of the software development has been completed, based on object models using the OMT (Object Making Technique). Device classes (klystron, magnet, vacuum, gun, monitor and others) are defined and coded as a super class on the control pack. The instances (real objects) are derived from their control packs on the GUI windows: for example, the magnet class has its sub class (such as Q-magnet, STC and bending-magnet). Each device class may use a PLC object for intercommunication. It is also related to the other super (or friend) class. Since Visual BASIC has a simple inheritance system, a common mother class has been coded based on the OMT using Visual C++ and was made as *.VBX tools on Visual BASIC. These classes were developed as a control pack which can appear in the toolbox in Visual Basic when it is in interpreter mode. Visual BASIC has limitations in memory size and execution time when many objects are created in the Windows derived from the mother class. A more objective language having better OOP features is now being tested in order to solve these problems.

8. Results and conclusion

Adequate speeds for the control devices were achieved by object oriented programming in the GUI and at the device layer. We could actually make common classes for the PLC device objects and the abstracted GUI objects. The standard class has resulted in a flexible control system at the console of the PF Linac. The OOP also provides users an easy way to make modifications and to produce a highly productive control system for accelerator users or machine operators.

References

[1] Feasibility study for PLC as a device controller, A. Shirakawa, Linac conference 1995, Osaka

[2] Object-Oriented Modeling and Design James Rumbaugh, Prentice Hall, Inc.

[3] GUI object model in Accelerator control
I.abe, K.Nakahara, M.Mutoh, Y.Shibasaki, Linac conference 1995, Osaka

[4] Control Protocol: The proposed new CERN standard access procedure to accelerator equipment.
G. Baribaud and etc. ICALPCS91, p591