pa-34

Finding Universal Objects for Accelerator MMI

S. Dasgupta* and I. Abe

High Energy Accelerator Research Organisation
1-1 Oho, Tsukuba-shi, Ibaraki 305, Japan

*Deputed from VECC; 1/AF Bidhan Nagar, Calcutta 700 064, India

Abstract

The accelerators over the world show a great variety, and the control systems of these are also varied. In the 3-tier control systems, the lowest levels are implemented according to the availability and historical situations regarding the hardware devices. The middle level is also influenced by the types of computers and the lower level devices.. The top level (Man-Machine Interface) shows a considerable similarity because of necessities of human interactions. The article discusses an attempt for finding of the most universal objects of such an MMI. This should enable the designers to apply modern tools for objected oriented designs of accelerator MMI. The Java language and its tools can be utilised to develop framework of MMI's which are muti-platform. It will then become suitable for producing re-usable codes for tailoring and fast reproduction at different accelerator.

Introduction

Accelerators, depending upon their family, purpose and energy, use subsystems and equipments from multiple sources and of wide variety. For this reason, and also because of varying personal tastes of designers, their control systems at different installations are much different. But the physics parameters and physical principles are broadly alike. With a global view towards accelerator control, it is possible to select several sets of common parameters and methodologies.. Such similarities can be encapsulated together to form 'control objects'. Having such 'objects', with similar properties and functionalities identified, the techinique of object oriented programming is applicable for the construction of control programs.

The well-known benefits are

(a) if a general aggrement on the rules and naming conventions are found, the softwares are portable to other installations and hence sharable.

(b) application of an object oriented language like Java for program development and subsequent ease in networking .

Among the several usual levels of a control system, the Man Machine Interface has the most universal 'objects', since human beings conceive such machines in most similar ways. Therefore it is worthwhile to analyse the common MMI objects of accelerators for their common behaviours.

The Case

The work reported here pertains to a specific case of two different kinds of accelerators eg., the linear accelerator PF LINAC of KEK, Japan and the Variable Energy Cyclotron(VECC), Calcutta. The MMI of an accelerator needs diverse appearances and functionalities in its different uses as console for design, for operation, for tests or for maintenance. In the present example, it is the operation console and its objects that are principally dealt with. The MMI objects have the responsibility of carrying messages to and fro the console operator on one hand, and the various objects of the accelerator control system on the other.

As the first step towards forming of MMI classes and sub-classes, the report here describes an work to produce a detailed view of the underlying classes and objects of an accelerator control system.

Objects of Accelerator Control System

An object oriented accelerator control system can be conceived as a collection of objects (of primary conceptual importance) of the following classes.

(a) Vacuum : to take care of vacuum pumps, gauges, valves, status indicators etc..

(b) Magnet : to take care of all magnetic elements eg. Solenoid, Quadrupole, Main, Trim, Magnetic channels their drivers and monitors, Analyser etc..

(c) Rf : dealing with High Frequency generators, their amplifiers, their measuring apparata, their accessories etc..

(d) Diagnostics : dealing with beam diagnostic parameters eg. beam current, beam phase, beam profile, orbit seperation, time structure etc..

Few other objects which also take part in the control and may need messages from console MMI objects are :

(a) Services : to take care of water-flow, electric power, air conditioning etc..

(b) Auxiliary : Few other types of entities, working as direct accessories for the above eg. Caxial Switches, Ion puller, RF-trimmers etc..

(c) Devices : power supplies, instruments, hardwired controllers eg., beam position stabiliser.

(d) Alarm : to take care of storage of alarm settings, processing of alarms etc..

In order to activate the accelerator control objects, by communicating to them, it is necessary to specify the public interfaces (ie., the public methods) of the classes of the control objects.

Based on studies and operation experiences, a list of variables and a list of methods on which these methods act, are under construction. The list cannot claim to be complete and exhaustive and is under construction.

Variables List

The following is a representative list that shows the class name in underline with their respective variables below.

Class Magnet

Setting, AverageField, FieldComponent, PeakAmplitude, FWHM, PowerSupply (Device class), FlowSwitches(Services class), etc.

Class Rf

Setting, Mode(CW/Pulse), PeakAmplitude, Frequency, MacroFrequency, Harmonics, Phase, SWRatio, Co-ax. Switch(Auxiliary class), FlowSwitches(Services class), PowerSupplies (Device class).

Class Vacuum

Pumps (Device class), FlowSwitches(Services class), OilLevel(Service class), Gauges, GateValves, Status etc.

Class Diagnostics

Particle, AverageAmplitude, PeakAmplitude, MacroWidth, FWHM, MacroFrequency, Power, HarmonicPower, OscillationFrequency, OrbitSeparation, Size, Profile, Phase, Emittance etc.

Class Device

On/Off Switch, Inc./DecControl, Setting, Measured, Range, FlowSwitches (Services class) etc.

Class Auxiliary :

Special Switches, Motors, Instuments etc.

Methods List

A preliminary investigation has led to unearthing a number of 'methods' necessary for each type of class. A few of these have been found to be of universal type and have been listed to represent the generic 'method's necessary for accelerator control objects.

(a) Operate( ) : to operate the on-off controls eg., switch off the cathode-bias voltage of the amplifier (an Rf object).

(b) Regulate( ) : to increment / decrement other types of controls eg., increment the current in an analysing magnet.

(c) Set( ) : to complete setting of any number of pre-specified values for certain members. eg., set all the beam-line magnets to a known set.

(d) TakeSnap( ) : to measure a parameter at an instant eg., measure the average magnetic field in a Magnet object.

(e) MonitorVsTime( ) : to measure and display a parameter of interest at specified time-intervals.

(f) MonitorVsSpace( ) : to measure and display a parameter of interest at specified distances.

(g) Plot( ) : to make graphical output on screen.

(h) Archive( ) : to log selected sets of data to permanent storage.

(i) GetHistory( ) : to get sets of data from storage.

MMI Objects

The specifications of the responding objects having been built, one can proceed to build the MMI object interfaces. An assortment of standard MMI components are readily available in many of the OO languages. The looks and structures may be sufficient to choose from. But it is necessary to tailor their variables and public interfaces to accommodate the common functionalities of accelerator MMI readily.

Recalling the methods necessary to send messages, the following are some of the 'Button' component types that can be universally useful.

Operate, Regulate, Set, TakeSnap, Standby, MonitorVsTime, MonitorVsSpace, Plot, Archive, GetHistory .

Pressing of any of these buttons should produce a 'Checkbox group' MMI component, tabulating the control objects to which the above button should send a message.

For example, clicking of 'Plot' button should produce a 'checkbox' with names of all acceleartor objects in which a plotting is possible. eg., objects of the classes Rf, Magnet, Diagnostics, Device etc.. On selecting an object from this 'checkbox', a next level 'checkbox group' objects must appear. The members of this new 'checkbox group' can be the variables of the class selected in the earlier checkbox. Thereafter a 'textarea' may take up values typed in from console.

The above example illustrates how , a detailed specification of the control system objects, is necessary for analysis of the MMI objects for accelerators and their classification.

After this analysis, it will be possible to find out and classify MMI objects of common functionalities.

Plans

The details of variables and functions to work as a common set for several accelarators can be found and therefrom, it will be possible to write codes to implement specific MMI objects of such accelerators. The use of a platform-independent language like Java is envisaged to make these codes portable to other installations. Also, the ease of interfacing to network browser, this way, will produce distributed system codes. The functionalities of attractive commercial GUI components can be thus extended to suit the requirements of the accelerator world.