

# A BEAM PROFILE VIEWER DEVELOPMENT WITH OO APPROACH

Tushar K. Das, Amitava Roy, Subrata Dasgupta

Variable Energy Cyclotron Centre, Calcutta 700 064, India

## Abstract

It is a 32 bit MDI application on WIN-95, written in VC++. It is meant to display and log the beam current intensity profile along radius of Cyclotron. It scans the internal beam-current along radius by a current probe, to acquire data through an ISA Add-on ADC/DIO board. In off-line, it is able to read previously scanned data files and for each, displays graphically the radial profile data obtained on three fingers of the probe. Child Windows can be opened to view the on-line and/or off-line data. The off-line data are members of a Document-class attached to a View-class through which the data can be displayed on a child window. The on-line data is a member of another class which interfaces the hardware and this data-member is referenced to the document-class. The application runs in cyclotron control room and helps operators to visually diagnose the unwanted loss of beam current. The application runs in a client server model, communicating through MFC-C++ sockets. In our retro-fitted control system, one control room PC is connected directly to the motor controller and beam-signal cables available in the control room. The user of the application works on another console-PC connected to the control-LAN.

## 1.INTRODUCTION:

The variable energy cyclotron was accelerating light ions for several years and had a beam profile viewer system to visualise the beam current profile along DEE radius, that runs on MS-DOS platform. Presently this cyclotron has been upgraded to accelerate the heavy ions, available from ECRI (electron cyclotron resonance ion) source. In a newer effort, for remote operation control of ECR source and other diagnostics, it was necessary to develop a newer system which will be running on WIN-95 platform where other control softwares are running concurrently. In this paper, design of the new software is described with a brief introduction to hardware set-up.

## 2.HARDWARE SET-UP

One current probe is being used to pick up the beam current signal, which can be moved along the radius of DEE with the help of a motor and gear mechanism. The probe senses the current intensity in three horizontal planes by its three fingers (upper, middle & lower) and these current signals & probe position signal (coming from motor and gear mechanism) are fed to the computer

after being converted to appropriate voltage level. The PC is interfaced to external world through a ISA add-on ADC/DIO board; this board has an ADC with 16 multiplexed analog channels and digital input-output port. Out of these 16 channels, four analog input channels are used for acquiring the beam current & probe position signals and two digital bits of the digital output port, are fed to the probe movement mechanism through opto-isolators, so that the probe movement can be controlled from computer. While in operation the probe is brought to a suitable position and then it moves along DEE radius and computer starts acquiring data in application's data memory area. Probe is stopped at its limit and acquisition is halted. Computer looks upon the whole operation with necessary precautions.

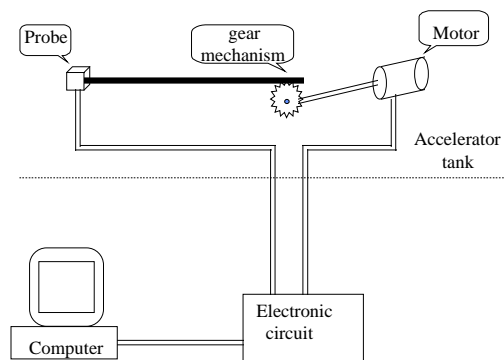


Figure 1: Schematic diagram of hardware setup

## 3.SOFTWARE

This software is developed as an MDI (multiple document interface) application on Micro-Soft Visual C++ package in object oriented style and it runs on WIN-95 platform. Being an MDI application it can display many current profiles on multiple child windows.

### 3.1 Architecture:

This beam viewer software is developed on the MSVC++ standard MDI architecture, shown in figure 2. In this architecture, an object of CWinApp class is the

entry point to the application.

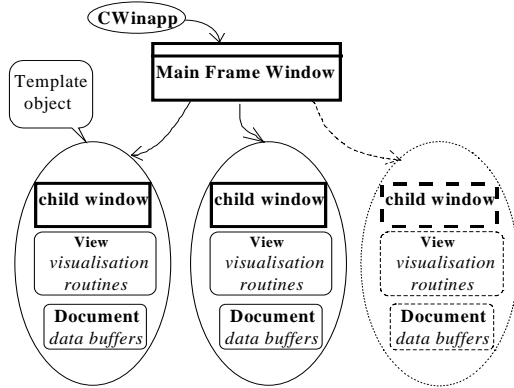


Figure 2: Software architecture.

This object instantiates a main-frame window class and as a result a frame window is displayed with its own menu. All the hardware interface routines are kept in this class because it is instantiable only once. It takes care of the probe movement and acquires on-line data & stores the data on its own data buffer. It displays the necessary hardware related information on its status bar. It does not display the beam profile, for this purpose child windows have been used. MSVC++ package has provided three interrelated classes for this kind of application, document class, view class and child frame class. The document class contains the necessary data buffer, view class mainly presents the data through child frame class. These three classes have been used to handle the on-line & off-line data to display. Basically, the main frame window class registers a template with those three classes along with a menu resource. Instantiation of this template creates those three classes, establishes their inter-relationship and shows a child window, on which beam profile can be viewed. This template is multi-instantiable, ensuring it as an MDI application.

### 3.2 Data Structure:

The data structure is designed in such a way so that user can view on-line data (collected from current probe) and off-line data (read from file) either simultaneously on a single child window or separately on multiple child windows according to the requirement. During on-line acquisition basically we read four integer type data and they are stored in four separate arrays which are put together in a *structure* (C language point of view). One such structure pointer is kept in main-frame class to store the on-line data. Another array of four such structure pointers is declared as private data member in document-class, one of which is always pointing to the structure of main-frame class. Remaining three *structure* pointers can be used to store the off-line data, read from files. This data structure enables the users to view atmost three off-line beam profile and/or on-line beam profile(Obviously one) on a single child window. On demand, memory can be dynamically allocated and assigned to those pointers

and afterwards it can be thrown away, so that unnecessary memory blocking can be avoided.

### 3.3 Remote controllability:

When the application starts working comfortably with the above architecture and data structure, we add few more classes to make it controllable from remote PC connected in network. MFC C++ socket class has been used to access the network and all hardware routines are put to a separate class where all these routines are declared as *virtual* (C++ language point of view).

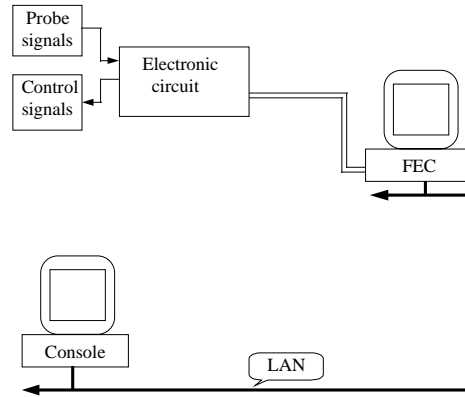


Figure 3: Schematic diagram of distributed control system.

From this class two child class have been derived, one for server application and another for client

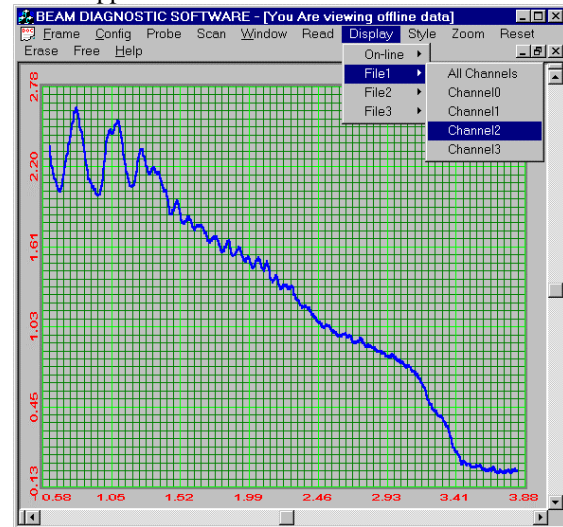


Figure 4: Typical output of this system (profile of oxygene ion beam)

application, those virtual functions are overridden in these child classes. The functions in child class, which is for client application, do not try to access the hardware but send the request to server application to access the hardware and get the status from the server application. Whereas in the other child class, which is for server application, functions are capable of receiving the request from client, accessing the hardware and sending the status

to that client. This client-server model fits into our distributed control system.

### *3.4 General Features:*

The features of this application is listed below.

- Scanning region of the current probe can be specified and software will not allow the user to bring the probe out of this region.
- Position of probe is displayed on the status bar of main frame window.
- During on-line acquisition, user may not take all the signals coming from the probe, any of those selectively can be acquired.
- On-line acquired data can be saved to a data file.
- User can see the signal of individual finger of the probe as well as all signals in superimposition.
- Data can be passed through a averaging filter to remove any kind of harmonic noise.
- Zooming of the profile is possible.

## **SUMMARY AND CONCLUSION**

From the above discussion it is clear that this application can be effectively used to visualise the unwanted loss in beam current and to tune the cyclotron. Being on the WIN-95 platform, concurrently it can run with other applications. The visual tools in MSVC++, helped to develop an attractive user interface.

## **ACKNOWLEDGEMENT**

We gratefully acknowledge the active support and help provided by Prof. Bikas Sinha and Mr. S. K. De. We are thankful to our colleagues for their keen interest in this work.