A PYTHON BASED INTERFACE FOR THE TANDEM-LINAC CONTROL SYSTEM

Ajith Kumar B P
IUAC, Aruna Asaf Ali Marg New Delhi, INDIA

Abstract

The control system for the Tandem-LINAC accelerator system at IUAC is a client-server design running on a network of PCs under the GNU/Linux operating system. The computers connected to the devices in the accelerator run a server program. The computers providing the user interface runs different kinds of client programs that communicates to the servers over a TCT/IP network to control/monitor the accelerator parameters. Both the programs were written in C language and some programming expertise was required to write the client programs. The addition of a Python language interface has enabled the users to write programs for specific tasks like data logging and partial automation of the operation with minimal effort.

INTRODUCTION

Inter-University Accelerator has a 16MV Tandem accelerator and a super-conducting heavy ion LINAC. The addition of the LINAC increased the number of signals and also demanded features like multiple operator consoles and the ability to run special purpose programs to condition the resonator cavity, automatically setting the amplitude and phase of RF etc. These requirements are met by the distributed control system, were the total number of signals are divided and connected to different computers on a network. This scheme if found to be simple, highly scalable and has been running for more that past 10 years /1/.

THE CLIENT-SERVER DESIGN

Every parameter of the accelerator is specified by three character strings; the name and location, function and unit. For example, CPS031 VC KV uniquely represents the voltage control signal of the charging power supply located in the post acceleration section of the Tandem. Signals of the accelerator are divided into small groups based on their location and connected to different computer running a server program. Details of the signal like the hardware address, resolution, range etc. are known only to the server handling it. Every server maintains a database of all the signals connected to it and listens over the network for commands from the client programs to manipulate any signal. The server programs currently support CAMAC, VME and some home-made hardware interfaces.

The client programs provide the user interface and other functions like alarms and automation. The client programs specify the signal using the three strings that uniquely specify it. The communication between server and clients is mainly through four generic functions. "Get Value" and "Set Value" for analog parameters and "Get Status" and "Set Status" for logical parameters. Several other functions are provided to get more information about the signal, like the minimum and maximum values, from the server.

IMPLEMENTATION

Both the client and server programs were written in C language. Writing a client program required linking it with the communication library to make the executable, a process demanding some programming expertise. To make this process simpler, a client side communication library is written in Python /2/ language that can talk to the servers over the TCP/IP network. Using this feature it is possible to write small Python programs to implement tasks like logging of accelerator parameters, automation of specific operations etc. Python language is chosen due to its clear syntax, ease of learning and the availability of libraries for graphics, networking and scientific computation. The benefits were evident to us due to its usage earlier in the Phoenix project /3/.

The simplicity of this scheme is demonstrated by the example program listed below, which reads the beam current from the Faraday cup and prints it.

```
import pelcon
p = pelcon.pserv()
print p.get_value('FC_021', 'CR', 'A')
```

This program can be easily modified to vary the injector magnet field and plot the beam current as a function of it.

CONCLUSION

The simplicity of the new Python interface has enabled those who working on different subsections of the accelerator to write their own scripts for specific tasks without having any detailed knowledge about the control system. It has been already used in automating some aspects of LINAC operation.

REFERENCES

[1]. Distributed control system for NSC tandem-LINAC. Ajith Kumar B.P. et.al, Indian Journal of Pure and Applied Physics Vol 39 Jan-Feb 2001
[2]. http://www.python.org
[3]. http://www.iuac.res.in