USE OF A VIRTUAL ACCELERATOR FOR A DEVELOPMENT OF AN ACCELERATOR CONTROL SYSTEM

Noboru Yamamoto
KEK,High Energy Accelerator Research Organization
1-1 Oho,Tsukuba, Ibaraki,JAPAN 305

Abstract

Concept of Virtual accelerator is introduced to develop beam control application programs in the KEKB accelerator control system. A virtual accelerator is a computer process which simulates behavior of a beam in an accelerator and responds to the accelerator control program under development in a same way as an actual accelerator do. Users of the virtual accelerator can test their control algorithm and user interface without affecting the operation of the accelerator.

EPICS(Experimental Physics and Industrial Control System) jointly developed by LANL and APS/ANL will be used as a base of KEKB accelerator control system. In the EPICS based control system, a device is represented as a collection of records in a EPICS runtime database. A control program on a host computer can access devices in the system only through names of record fields, called 'channel'. This abstraction allows us to construct a virtual accelerator control system. Channel access interfaces was implemented into the modeling program SAD[1] to realize this idea. Sample application of the method will be also presented.

1 USE OF VIRTUAL ACCELERATOR CONCEPT.

1.1 Accelerator Model in an accelerator control system.

A concept of a virtual accelerator(VA) had been discussed within the SAD program development team [1] since the early stage of the development of SAD modeling code. It is one of goal in the SAD project to develop a environment in which one can access both an accelerator model and a machine using same user interface and the users cannot easily distinguish them. A user of the program writes a script to analyze data for the model. VA concept in SAD allows to use this program for the operation of REAL accelerator without any modification or with tiny modification.

In the TRISTAN operation, we learned importance of modeling program in the accelerator control system[4]. Comparison of the measured data and results of simulation by modeling code gives us a deeper understanding of the an accelerator and were essential to improve performance of the accelerator. Since the TRISTAN control system was designed long before SAD was developed, data files are the only way to exchange data between two environment. It work, but takes time and human intervention. This experience suggests that tighter integration of modeling program and control system is necessary in the future accelerators such as KEKB[5]. If we can use same operator interface to control both "REAL" and "VIRTUAL" accelerators, we can directly compare the results of measurement and the simulation in same environment.

1.2 VA as a RAD(Rapid Application Development) tool

Tightly integrated modeling code in a control system, or a virtual accelerator, is also useful as a RAD tool in the construction of an accelerator control system. Application programmer and/or an accelerator physicist can develop a code without waiting for the completion of hardware installation. Realistic response using the VA will help the development of high level applications in the control system.

Figure 1 shows a sample application developed with the VA. User can display the horizontal and vertical orbits of KEKB-LER(Low Energy Ring) and can also issue several commands, such as "correct orbit", to the control system. This application, in principle, can be used with the real KEKB-LER when it is ready to operate. MEDM is used to build a user interface in this case. Shell sciripts and SAD scripts are used to describe behaiviour of the application.

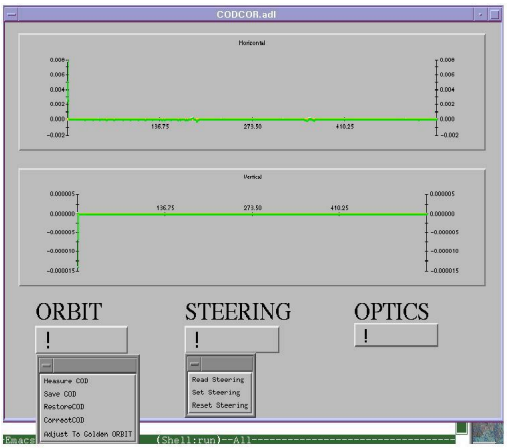

Figure 1: Sample application using VA.

1.3 VA as a "Flight simulator"

It is also useful to use VA as a "Flight simulator". Trainee of the accelerator operation can learn the response of the system even in a situation which should not occur in the real machine.

In the commissioning of the new accelerator, it is necessary to examine new algorithm to optimize performance of the beam. "Flight simulator" allows accelerator physicist to examine his/her algorithm without disturbing the operation.

0-7803-4376-X/98/$10.00 © 1998 IEEE

2 A CONTROL SYSTEM WITH VA.

To realize VA concept in the accelerator control system, a modeling program and control system architecture should have the following features.

1. Control system should have a layer to hide hardware detail from the an application program. Application should access control devices through a logical names but not through hard coded hardware address. Without this layer it is difficult to develop an application which can access both "real" and "virtual" accelerators using same program interfaces.

2. Modeling program can access the information on the control system. And the control system can access data in a process running the model.

3. A control system should allow multiple instance of virtual accelerators at same time. So that multiple users can develop their control application without interference between them.

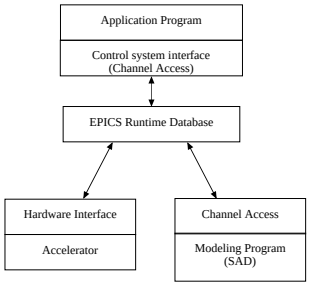

Figure 2: EPICS Database allows application program to access hardware devices and data on a simulation process in a unified way.

2.1 EPICS runtime database and channel access.

We have decided to use EPICS(Experimental Physics and Industrial Control System) as a infrastructure of the KEKB accelerator control system[6]. Runtime database and channel access, which are key parts of the EPICS system, provides a layer which we discussed in the previous section. Figure 2 shows how EPICS runtime database hides a diffrence between "REAL" and "VIRTUAL" accelerators from the application program.

2.2 SAD: an accelerator modeling program

SAD is a computer program complex for accelerator design. It has been developed in KEK since 1986. The major functions are:

1. Structural Definitions of Beam Line and Component[10].

2. Ritch Matching Functions

   (a) Optics matching

   (b) Optical/Geometrical matching

   (c) Off-momentum matching

   (d) Finite-amplitude matching

   (e) Spin matching

3. SAD Script Programming Interface in Mathematica[11] Style. Major functions for lists and structural operations. Built-in, system- and user-defined functions for accelerators

4. 6D full-symplectic particle tracking for Dynamic aperture survey with/without Synchrotron radiation effect.

5. Taylor map by automatic differentiation and Lie algebraic map for Nonlinear Analysis

6. Emittance Calculation using 6D Beam-matrix method[2]

7. Anomalous Emittance Calculation [3]

8. Spin depolarization(SODOM)

SAD has proven to be powerful and useful in designs, simulations, commissioning and improvement of TRISTAN, KEKB, FFTB, ATF, JLC, NLC, and others. Various packages for accelerator studies are written for SAD scripting language. These packages will be used in the operation of KEKB.

2.3 EPICS interface in SAD

EPICS CA interface was implemented in the SAD. Using this interface SAD can directly access data in the control system. This interface is also used to put a response from VA, which is one of process running SAD, into EPICS database. CA interface in SAD has just three functions, $CaOpen$, $CaRead$ and $CaWrite$. These functions can take a list as a its argument to take advantage of asynchronous operation in EPICS CA. For example, if you have a list of EPICS channel names,

$$channels = \{\text{“}BPM1\text{”},\text{”}BPM2\text{”}, \ldots\},$$

a single call of $CaOpen$,

$$CaOpen[chnnels]$$

will return a list of opened channel ids. $CaWrite$ function can take this list of channel ids and returns a list of current values of these channels. This list handling allows SAD to read data from EPICS database efficiently.

However these interface allows to construct VA, you need to prepare EPICS database definition files for each instance of VAs and needs to load these databases onto IOC(Input Output Controller, VME CPU running EPICS core software). We are now developing another mechanism to support VA using the portable CA server program developed at LANL. The "portable CA server" is a collection of

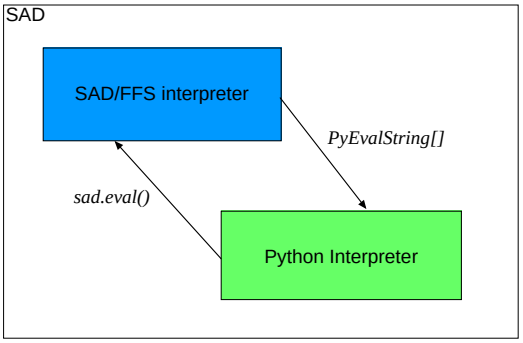

Figure 3: SAD code includes both SAD/FFS interpreter and Python interpreter in it. Two interpreters can exchange a expression as a string for evaluation.

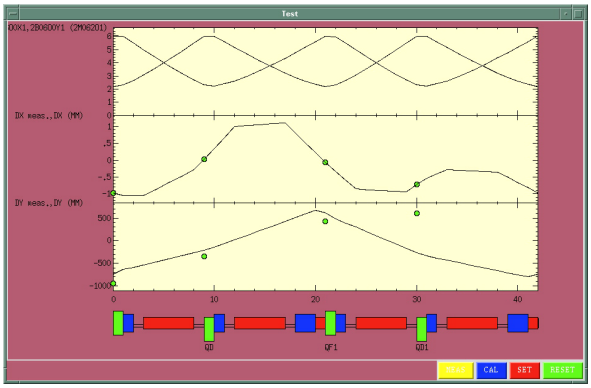

Figure 4: SAD can use Python/Tk to provide GUI.

C++ classes to support development of CA server on Unix platform. Using this framework, you can create SAD-CA bridge software, which takes CA request, get a data from SAD processes and returns these value to the CA clients. The SAD-CA bridge can also create channels connected to the SAD process dynamically. It allows unlimited numbers of VA instances in the control system.

2.4 Python/Tk in SAD

Python is “an interpreted, interactive, object-oriented programming language”[8]. Like Tcl/Tk[9] it can be used as an extension language for applications that need a programmable interface. Python has Tk bindings, Tkinter.py, so that you can create GUI using the Python. Figure 3 shows how Python is embedded in SAD program. SAD passed a string including any Python expression to Python core and Python core evaluate it. Python core can also pass a string which represents SAD/FFS scripts to SAD for evaluation. It provides basic mechanism to exchange data between two interpreters. Wrapper functions are written on this basic mechanism.

K. Oide [12] wrote a sample user interface for the orbit correction as shown in Figure. 4.

3 CONCLUSION

We have implemented a basic mechanism to realize VA concept in the KEKB accelerator control system. Using the this mechanism, a few sample application program were written. Futher development is needed to allow multiple instance of VAs and its dynamic creation.

4 REFERENCES

[1] “SAD home page”, http://www-acc.kek.jp/SAD/sad.html. ; K.Hirata, An introduction to SAD, Second Advanced ICFA Beam Dynamics Workshop, CERN 88-04 (1988).

[2] K.Ohmi, K.Hirata and K.Oide, From the beam-envelope matrix to synchrotron-radiation integrals,Phys.Rev.E49, 751 (1994).

[3] . K.Oide and H.Koiso,Anomalous equilibrium emittance due to chromaticity in electron storage rings, Phys.Rev.E49, 4474 (1994).

[4] ”Integration of the Modeling Program with Accelerator Control Systems in TRISTAN”, Proceedings of the 1995 International Conference on Accelerator and Large Experimental Physics Control Systems, Chicago, Illinois, pp. 423-428

[5] S-I Kurokawa, “ Status of TRISTAN-II Project”, Proceedings of the 1993 Particle Accelerator Conf, pp. 2004-2006; S-I Kurokawa, “KEKB status and Plans”, 1995 Particle Accelerator Conference and International Conference on High-Energy Accelerators, Dallas, Texas,USA, May 1-5, 1995.

[6] T.Katoh et al., “Status of the KEKB accelerator control system development”, Proceedings of the 1995 International Conference on Accelerator and Large Experimental Physics Control Systems,Chicago, Illinois,1995, pp. 899

[7] L.R. Dalesio, et al., “EPICS Architecture”, ICALEPCS 91, KEK Proceedings 92-15, (1992) pp.278-282.; L. Dalesio et al. “The Experimental Physics and Industrial Control System Architecture,” ICALEPCS 93, Berlin,Germany, Oct. 18-22, 1993.

[8] “Python Language Home Page”, http://www.python.org

[9] ”Why Tcl?”, http://sunscript.sun.com/tcltext.html

[10] D.C. Carey and F.C. Iselin, “A standard input language for particle beam and accelerator computer program”

[11] S. Wolfram, “Mathematica: A system for Doing Mathematics by Computer”,Second Edition, Addison-Wesley Publishing Company, 1991

[12] Katsunobu Oide, KEK, in private communications.