SLS-TME-TA-1999-0015
September, 1999

A CORBA Based Client-Server Model for Beam Dynamics Applications at the SLS

Michael Böge, Jan Chrin

Paul Scherrer Institut
CH-5232 Villigen PSI
Switzerland

Contributed Paper (MC1P61) to the International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'99), 4-8 October 1999, Trieste, Italy

A CORBA Based Client-Server Model for Beam Dynamics Applications at the SLS

M. Böge†, J. Chrin††
Paul Scherrer Institut, 5232 Villigen PSI, Switzerland

†*email: Michael.Boege@psi.ch*
††*Presenting author, email: Jan.Chrin@psi.ch*

Abstract

A distributed object oriented client-server model, based on the Common Object Request Broker Architecture (CORBA), has been established to interface beam dynamics application programs at the Swiss Light Source (SLS) to essential software packages. These include the accelerator physics package, TRACY, the Common DEVice (CDEV) control library, a relational database management system and a logging facility for error messages and alarm reports. The software architecture allows for remote clients to invoke computer intensive methods, such as beam orbit correction procedures, on a dedicated server running the UNIX derivative, Linux. Client programs typically make use of graphical user interface (GUI) elements provided by specialized toolkits such as Tk or Java Swing, while monitored data required by procedures utilising the TRACY library, such as beam optics parameters, are marshalled to the model server for fast analysis. Access to the SLS accelerator devices is achieved through a generic C++ CDEV server. The architectural model components are described and a prototype application within the beam dynamics environment is presented.

1 INTRODUCTION

The Swiss Light Source (SLS) [1] is a 2.4 GeV electron storage ring currently under construction at the Paul Scherrer Institute, Switzerland. Electrons from an injector booster synchrotron, fed by a 100 MeV linear accelerator, are transferred to the main ring at full operating energy. Scheduled for operation in August 2001, the SLS will provide synchrotron radiation of high brilliance to experimenters from a variety of disciplines. A considerable number of high-level beam dynamics application program interfaces (APIs) are required for the commissioning and operation of the SLS accelerator complex and for machine physics studies. These APIs typically share a number of generic tasks including:

- access to an accelerator physics package,
- accelerator device control,
- database access and management, and
- logging of error messages and alarms.

With the aid of object-oriented methodology, common functions can be identified and developed as reusable components. Furthermore, a distributed system allows optimal use of available resources, an important consideration given the computer intensive physics algorithms employed by the accelerator modelling procedures. To this end, a distributed client-server model, based on the Common Object Request Broker Architecture (CORBA) [2], is presented; client programs readily access shared services, either locally or across the network, through CORBA objects.

2 ARCHITECTURAL MODEL

In the evolution of object-oriented distributed computing systems, CORBA is a recent standard that provides a mechanism for defining interfaces between distributed components. Its most distinguished assets are platform independence, in so far as the platform hosts a CORBA Object Request Broker (ORB) implementation, and language independence, as ensured through the use of the Interface Definition Language (IDL). The latter feature is of particular interest to SLS beam dynamics API developers as it provides for the option between high-level application languages. For instance, the client component of the prototype closed orbit correction API has been implemented in Tcl/Tk [3] using the BLT extension, a package that is an appropriate match to the requirements of this particular application. The server components, on the other hand, have been implemented in C++ for high performance and run on a dedicated server machine. It is interesting to note that in this multi-language scenario, the Tcl/Tk client program is comparatively short in length and, therefore, quite managable. Optimal use of the Tk/BLT package is made for building the graphical user interface (GUI) component of the API, while the more complex components are routed to server processes by means of CORBA objects.

2.1 Server Hardware and System Software Components

The chosen operating system and platform for server applications is RedHat Linux version 6.0 on a server housing dual 550 MHz Intel Pentium III processors. The use of Linux and the GNU project C++ compiler

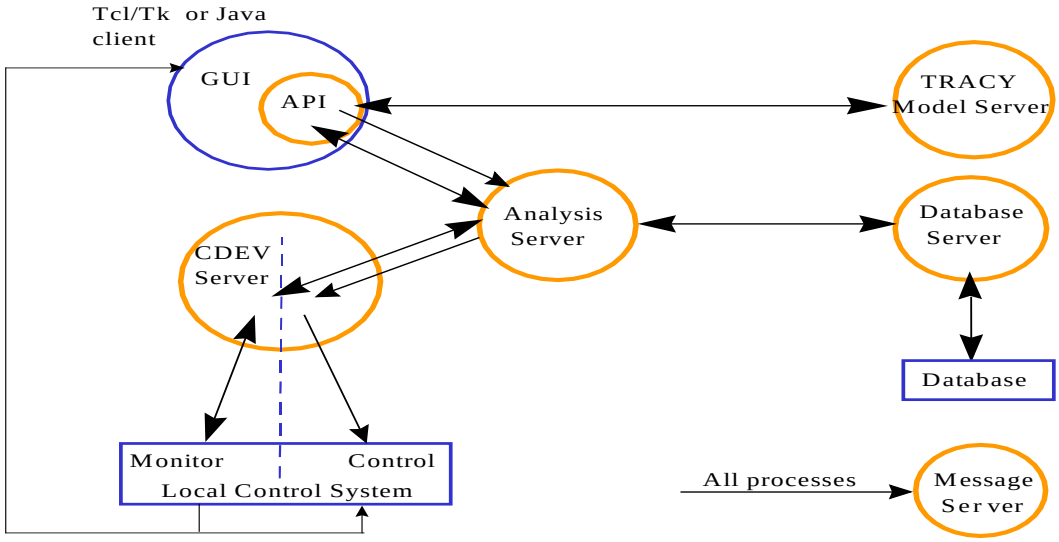

Figure 1: Software Architectural Model for SLS Beam Dynamics APIs

(egcs) avoids vendor dependency; compilation with egcs further reduces the dependency on the operating system thereby increasing the portability of applications. A second identical server is permanently available to provide redundancy. Client programs typically run on local Linux PCs.

The principal CORBA product employed is MICO [4], a fully compliant CORBA 2.2 implementation, available free of charge under the GNU public license terms. Use is made of the Naming Service and Interface Repository facilities provided by MICO. Significantly, in addition to the given IDL to C++ mapping, a Tcl interface to MICO [5] that provides CORBA client and server functionality to Tcl scripts has been incorporated. Noticably absent from MICO at present, however, is mapping for the Java programming language [6]. Since Java client components will be an integral part of SLS beam dynamics applications, another CORBA product, namely the Java-based JavaORB package [7], provides IDL mapping to Java. Applications involving permutations of client-server processes written in C++, Tcl/Tk and Java have all been tried and tested, further verifying the interoperability between the different CORBA 2-compliant products[1].

2.2 Server-Client Software Components

Fig. 1 illustrates the conceptual design of the software model employed for the retrieval, analysis and display of pertinent data for a specific beam dynamics API, namely the closed orbit display. The prototype Tk/BLT client GUI is shown in Fig. 2.

A CORBA interface to the C-based TRACY library [8] provides users with convenient access to the accelerator physics routines. This capability in itself provided strong motivation for the use of CORBA as it allows access to the same machine model as used in offline simulations; procedures tested in simulation can thus be effectively employed for the optimization of the accelerator *online*. In this present example, measured beam positions are marshalled to the dedictated TRACY model server for analysis. A new set of corrector values is calculated and presented to the client together with the predicted orbit. The corresponding hardware settings required to achieve the improved orbit are handled by the Analysis and Database Servers.

Synchronous and asynchronous interaction with the EPICS-based local accelerator device control system [9] is achieved through use of the Common DEVice (CDEV) C++ class library (version 1.6.2) [10]. A generic CDEV Server employs a CORBA server object that responds to CDEV-type verbs. The “set”, “get” and “monitor” verbs are accompanied by a CORBA sequence of objects containing the parameters required to, respectively, download setpoints, readback device attributes and monitor selected channels. The configuration information is held in a SQL92 compliant relational database [11] which is accessed through a CORBA wrapped Database Server. The Analysis Server further retrieves monitored data from the real-time system, through the CDEV Server, for recalibration and analysis. The application API embedded in the client GUI component polls the Analysis Server for updated values and displays them.

All client-server processes are able to report error messages and alarms to a dedicated Message Server through a CORBA interface. Presently the server employs the UNIX syslog message logging facility, incorporating a variety of priority levels. Syslog entries are further converted to SQL insert queries for immediate entry into

[1]Interoperability is made possible by virtue of the CORBA 2.0 requirement that the Internet Inter-ORB Protocol (IIOP) be the standard protocol for communication between ORBs

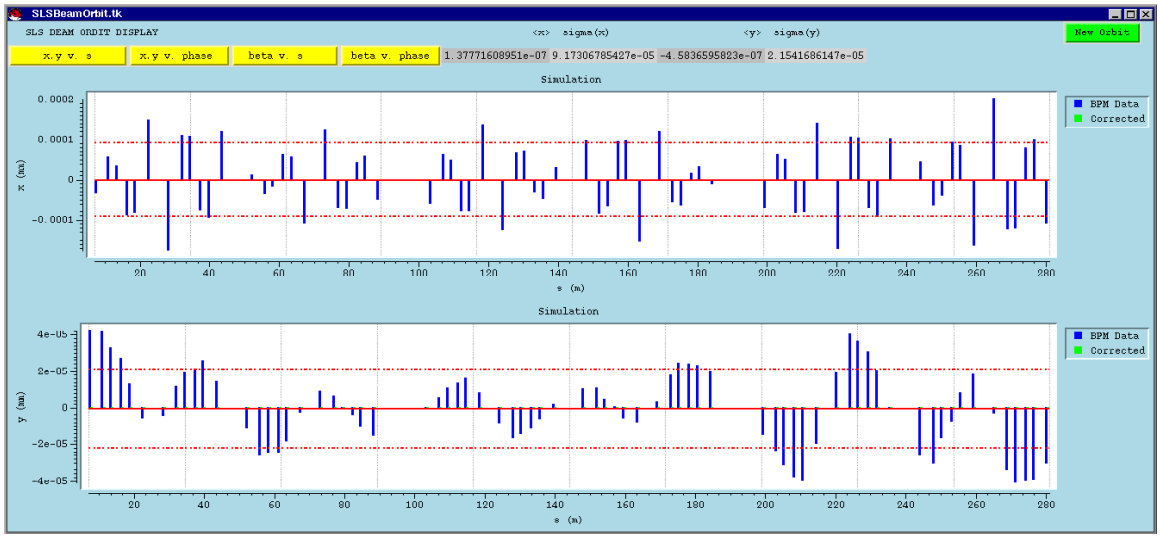

Figure 2: A prototype client application displaying measured beam positions

the database. Error messages are viewed either through a Tcl/Tk based browser or the native database browser.

The server-client components featured are typical of the requirements of several anticipated applications. The framework further allows for critical code components to be better tested through their eventual use in different APIs.

3 FUTURE DEVELOPMENTS

The model presented here represents work in progress and, as such, a number of developments are foreseen. Since several of the servers have write privileges to sensitive software and hardware channels, it is intended to add authentication procedures that identify and authorize the client, e.g. through use of the Secure Sockets Layer (SSL), a protocol also supported by MICO. Server diagnostic tools will also be added to provide a synopsis of usage and performance.

It is envisioned that configuration, calibration and other data will be held in an Oracle database. Work is in progress within the CDEV community to interface Oracle to the CDEV service layer allowing easy database access through the CDEV device/message paradigm [12].

Of the CORBA facilities and services that are becoming increasingly available, particularly appealling is the Event Service which offers a convenient channel for distributing data to one or more consumers. Data from a supplier is distributed to consumers, on a push or pull basis, without the supplier requiring knowldege of the receiving objects. Such a service would be usefully employed in the distribution of calibrated data to client consumers.

The use of Java is noticably gaining momentum in the accelerator physics community; its unique features of garbage collection, exception handling and integrated thread support are desirable assets for building large-scale distributed systems. Java Swing and Java Beans further offer components for building GUI operator interfaces (OPIs). In this respect, an effort to coordinate activities with the aim of releasing a standard Java OPI is on the Software Sharing Workshop agenda [13].

4 CONCLUSION

An object oriented client-server model in which dedicated C++ servers provide essential services to clients by means of CORBA objects has been presented. A prototype client application has demonstrated that the proposed architectural model offers an appropriate framework for application programmers to develop APIs within the beam dynamics environment.

5 REFERENCES

[1] M. Böge *et al.*, “The Swiss Light Source Accelerator Complex: An Overview”, Proc. 1998 6th European Particle Acc. Conf. (EPAC-98), Stockholm, Pub: IoP, UK, p. 623

[2] OMG, CORBA, http://www.omg.org/

[3] J.K. Ousterhout, “Tcl and the TK Toolkit”, Pub: Addison-Wesley; Tk/BLT, http://www.tcltk.com/blt/

[4] A. Puder, K. Römer, “Mico is CORBA”, Pub: Ap Professional; http://www.mico.org/

[5] F. Pilhofer, “TclMico, A Tcl Interface to Mico”, http://www.informatik.uni-frankfurt.de/~fp/Tcl/tclmico/

[6] Java, http://java.sun.com/

[7] JavaORB, http://www.multimania.com/dogweb/Projects/JavaORB/javaorb.html

[8] J. Bengtsson, “TRACY-2 User’s Manual”, SLS Internal Document (1997); M. Böge, “Update on TRACY-2 Documentation”, SLS Internal Note, SLS-TME-TA-1999-0002 (1999); http://slsbd.psi.ch/pub/slsnotes/

[9] M. Dach *et al.*, “Control and Data Acquisition System of the Swiss Light Source”, ICALEPCS’99, Invited Paper MA1I01

[10] CDEV, http://www.jlab.org/cdev/

[11] PostgreSQL, http://www.pgsql.com/

[12] W. Watson, T. Pal, private communication

[13] SOSH’99, http://www.jlab.org/sosh/sosh99/