STATUS OF THE CORBA BASED SOFTWARE ENVIRONMENT FOR BEAM DYNAMICS APPLICATIONS AT THE SLS

M. Böge, J. Chrin

A CORBA based client-server framework has been in active service since the commissioning of the SLS, providing an uniform interface to a variety of objects required by beam dynamics applications. The framework underwent significant development during the course of the year, motivated by advances in computer hardware capabilities and recent releases of CORBA software packages. In addition, several more software modules were developed and integrated into the framework. They include event processing agents which serve to aggregate low-level hardware data to produce complex events that offer a personalized view of a given component of the control system. The events are propagated to applications through designated event channels, managed by the CORBA Event Service. A specific set of persistent objects, aimed at facilitating the tune measurement and retrieval of the resulting data, has also been implemented.

INTRODUCTION

The Common Object Request Broker Architecture (CORBA) [1] forms the middleware layer and access point to several essential packages required by beam dynamics applications at the SLS [2]. Reusable components are developed as CORBA objects that can be readily accessed by client programmes. The use of CORBA further serves to realize the potential benefits of distributed computing, an important consideration given the computer intensive accelerator modelling procedures, and provides for the interoperability between objects implemented in different progamming languages. This latter feature, unique to CORBA, has allowed developers to program high-level applications in their preferred language.

The CORBA framework [3], in active service since the commissioning of the SLS, underwent significant development during the course of the year. Modern computer hardware was commissioned to meet with the increasing demands imposed on the server hosting the CORBA objects, referred to as the “Model Server”, and recent versions of CORBA software packages were likewise installed. In addition, further software development served to consolidate the manner in which selected data from the low-level hardware are aggregrated by intelligent agents. These act to trigger *complex* events that are propagated to high-level applications through event channels supplied by the CORBA Event Service. Clients need only subscribe as a consumer to the appropriate event channel to passively receive updated values. They are consequently relieved from the communication details and the need to establish callbacks. A specific set of persistent objects, aimed at facilitating the timely retrieval of results from the computer intensive tune measurement algorithm, has also been implemented.

HARDWARE AND SOFTWARE PLATFORMS

Beam dynamics applications are constructed using a client-server framework, wherein client applications typically operate on consoles located in the accelerator control room and connect to the “Model Server”, that negotiates access to the controls hardware, the accelerator model and the database on the client’s behalf. With the increased demands from high-level applications (e.g. the fast orbit feedback [4]), the “Model Server” was replaced by a dual-processor Dell PowerEdge 2650 Server running Red Hat Linux v. 7.3. The server features two 2.8 GHz Xeon processors, 2 GByte of RAM, a 1 GByte Ethernet card and a RAID controller that duplicates data onto a second, hot-plug, 36 GByte SCSI hard drive, providing additional data security. It is also equipped with two hot-plug power supplies. A second identical server is available to provide redundancy. Maintenance of the server machines has, as of this year, been delegated to the SLS Controls group.

The introduction of the high performance hardware was combined with the installation of recent releases of our CORBA products [5], namely MICO (C++ mapping), ORBacus (Java) and Combat (Tcl). The new MICO release, our principal ORB, reflects a tighter implementation of the specifications formulated by the Object Management Group (OMG) [1]. In particular, a more robust Event Service was introduced and is now extensively employed for the propagation of logical sets of controls data to high-level applications. Updates of software packages are provided to the SLS Controls group through the Red Hat Package Manager (RPM).

AN EVENT SERVICE FOR DATA PROPAGATION

The synchronous request/response exchange is the standard means of communication between a client and server. However, in certain instances, the optimal means of data transfer is through an alternative, *reactive*, form of programming wherein clients are notified *en masse* of updated values. The CORBA Event Service provides such a delivery mechanism. It supports decoupled communication between multiple suppliers and consumers and has been effectively employed for the propagation of low-level hardware data. Fig. 1 illustrates how control data are aggregrated by an event processing agent (EPA) to produce a *complex* event which is propagated to those applications subscribing as consumers to the designated event channel. A typical EPA uses the CDEV [6] Application Progamming Interface (API) to establish a callback mechanism to the EPICS [7] based control system, the communication

protocol of which is channel access. In general terms, the EPA is a simple object that consists of event pattern rules, comprised of a *trigger* and a *body* of actions, and local variables whose values form its state. The EPA monitors its input to detect instances of the rule triggers. When a match is detected, the agent executes the action of the rule's body. With reference to Fig. 1, an invocation of the CDEV callback function causes the EPA to change its local state variables and its output event. When the trigger is satisfied, a *complex* event (i.e. an aggregration of the low-level hardware event) is formed and routed to its designated event channel.

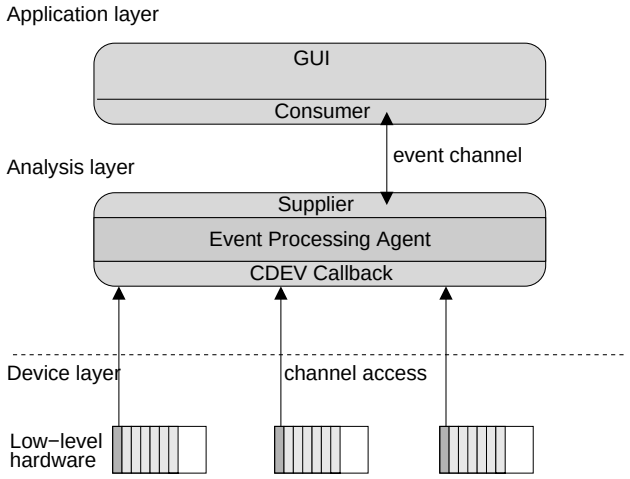

Fig. 1: Data aggregration and propagation.

EPAs exist for the aggregration of several types of devices, including Beam Position Monitors (BPMs) and various magnet groups. The corresponding event data thus provides a personalized view of a given component of the control system.
The event channels are the primary source of information for users and are optimized to satisfy their reporting needs and so that no further data manipulation is required on the client side. Indeed, much of the data on display in beam dynamics applications are received in this way. It is also interesting to note that this work demonstrated that errors were often easily detected by viewing higher level events and that drill down techniques could be used to locate the cause of an error at low-level. The presense of an error would otherwise go undetected if only low-level events were monitored!

CORBA TUNE OBJECTS

A set of persistent CORBA objects has been developed with the aim of decoupling the measurement of the critical tune parameters from the time consuming I/O (input/output) operations involved in writing large data volumes to EPICS records. The computer-intensive calculation of the vertical and horizontal components of the machine tune parameter has been incorporated into a dedicated EPA. On completion of the measurement, the data, including those of the tune BPM waveforms, are stored in virtual memory space (avoiding I/O operations) and the measured tune values are immediately fed into an event channel, to which any interested client is able to subscribe. A dedicated tune server provides methods that enable a client to both regulate input parameters to the tune calculation and to retrieve the results from the data store. A further program, acting on a trigger from the designated tune event channel, retrieves the full complement of results from the tune server and directs them to EPICS records for archiving. There are a number of advantages to this three-tiered approach:
(i) the latest tune values are made available to interested clients as soon as they become available,
(ii) input parameters to the tune calculation can be optimized on the fly, and
(iii) the large volume of data from the resulting calculation can be retrieved by clients without jeopardising the flow with which the latest tune values are determined.
The new framework has been effectively employed by the Java based tune application described in [8].

SUMMARY

An updated CORBA-based software framework has been operating on recently commissioned modern, high-performance, hardware offering an improved response time and greater stability. Recent software developments have served to consolidate the CORBA Event Service as the standard mechanism for the propagation of aggregrated control data to high-level applications. Several new CORBA objects have been developed to optimize the timely retrieval of results from the computer intensive tune calculation.

ACKNOWLEDGEMENTS

The authors wish to thank M. Dach, B. Keil, A. Luedeke, T. Schilcher and D. Zimoch for their assistance in matters concerning the low-level hardware. In particular, the provision to faithfully mirror EPICS records onto the PSI network has greatly eased the task of testing software.

REFERENCES

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

[2] M. Böge, J. Chrin, M. Muñoz, A.Streun, SLS-TME-TA-2001-0182;
M. Böge, J. Chrin, SLS-TME-TA-2003-0225.

[3] M. Böge, J. Chrin, SLS-TME-TA-2000-0162, SLS-TME-TA-2001-0183, SLS-TME-TA-2002-0217.

[4] M. Böge et al., *Orbit Stability at the SLS*, this report.

[5] MICO v. 2.3.8, http://www.mico.org/
ORBacus v. 4.0.5, http://www.ooc.com/
Combat v. 0.7.2, http://www.fpx.de/Combat/

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

[7] EPICS, http://www.aps.anl.gov/epics/

[8] M. Muñoz, *Improvements to the Tune Measurement*, this report.