Preliminary Design of MUSES Control System Based on RT-CORBA and Java

Toshiya Tanabe*, Toshikatsu Masuoka, Jun-ichi Ohnishi, Motonobu Takano and Takeshi Katayama

RIKEN, Beam Technology Division / Beam Physics and Engineering Lab.
2-1 Hirosawa, Wako, Saitama 351-0198, JAPAN

*e-mail: ttanabe@mailman.riken.go.jp

RI Beam Factory (RIBF) Facilities Overview

- **Existing Facility [RIKEN Accelerator Research Facility (RARF)]**
  - ECR Ion Source
  - RFQ
  - RIKEN Linear Accelerator (RILAC)
  - Charge State Multiplier (CSM)
  - Riken Ring Cyclotron (RRC) [K=540]
  - RIKEN Projectile fragment Separator (RIPS)

- **1st Phase**
  - Intermediate Ring Cyclotron (IRC) [K=950],
  - Superconducting Ring Cyclotron (SRC) [K=2500],
  - Big-RIPS

- **2nd Phase [Multi-USe Experimental Storage rings (MUSES)]**
  - Accumulator Cooler Ring (ACR) [E_{max}=400MeV/u for light ions]
  - e-RI Collider
http://www.rarf.riken.go.jp/rarf/index.html

-> e-RI Collider (2001)
Under this project, three ring cyclotrons (IRC, IRC and SPG) will be constructed in addition to the present RIKEN Accelerator Research Facility (RARF). The purpose is to generate the RI beam of all atoms without limiting to the light ions, as limited so till today, at the world highest intensity. The quality and energy levels of the RI beam will be increased drastically (that is, the energy spread will be reduced extremely fine) thanks to multi-purpose rings (MUSES) to be constructed under this project. The factory will promote the advanced research ranging from the basic to the application.

Welcome to RI Beam Factory

- Overview of RIBF (latest pdf article included)
- New Booster Ring Cyclotrons
- MUSES
- Basic Science at RIBF
- Members(FY2001)

Technical Information

- Control Systems
- Folded Cooaxial RFQ
- ECR Ion Source

Last updated 2001.06.18

Copyright (C) RIKEN, 1998-2001

Access Statistics

URL: http://ribfweb1.riken.go.jp/
RIBF Controls General Policies

- **Vendor/Platform Independence & Portability of Applications**
  - Long-term maintenance and future upgrade
  - Industry Standard (CORBA, Java, etc.)
    - Avoid MS-lead standards (DCOM, Active-X, etc.)

- **Minimization of Own Software Development**
  - Lack of man power (~5FTE)
  - Use of COTS products as much as possible
  - Software sharing with other labs (JSI, KEK, etc)

- **Reliability over Cost on Hardware**
  - VME over PC
  - Optical links wherever necessary
  - Radiation resistant
RIBF Control System Overview

- **Existing Facility [RIKEN Accelerator Research Facility (RARF)]**
  - Partial Upgrade of Upper-Layer System Software Using **EPICS**
  - Keep Legacy HW (Camac and proprietary field bus) in the Controller Level

- **1st Phase (IRC, SRC and Big-RIPS)**
  - Relatively Slow Controls
  - Large and Precise PSs (**IRC PS Controls Currently Done via CORBA**)  
  - High Radiation Dose

- **2nd Phase MUSES (ACR and e-RI Collider)**
  - Fast Synchronization with Varying Speed
  - Large Number of Data Points
  - Various Sub-Devices such as **Stochastic Cooler & Electron Cooler**
Java
✦ GUI and network functions are included in the language
✦ Good development tools
✦ Write once, less debug everywhere
✦ Garbage collection is supposed to reduce memory leaks
× Still slower than native codes (mostly library loading)
× Memory hungry
× Constantly changing / Variations in versions

CORBA
✦ Simplify the development on communications
✦ Platform/Language independent
✦ Variety of freeware available (not so for RTOSs)
× Need good IDL design to be useful
× Services are still somewhat vendor dependent
× Not real-time compliant ➔ RT-CORBA
CORBA Based Facilities in the World

- **ANKA Light Source (Karlsruhe, Germany)**
  - NT+VisiBroker+PC+LonWorks
  - JavaBeans for GUI applications

- **ESRF (European Synchrotron Radiation Facility) Tango**
  - Upper-level Layer Modification with ORBacus and MySQL

- **NIF (National Ignition Facility@LLNL)**
  - Upper-level Layer --> Solaris
  - Lower-level Layer --> VxWorks in VME and ORBexpress

- **Observatories**
  - ESO (European Southern Observatory)
  - DYNACORE (DYNAMically Configurable Remote Experiment monitoring and control)

- **SR Beamlines**
  - ALS (Advanced Light Source)
From socket application to CORBA
   - Make socket based application for GUI (C++) and controllers (VME&VxWorks)
   - Replace communication parts of the programs with CORBA

Server object exchange
   - Make CORBA based application with a server object in VME & VxWorks 5.3.1
   - Replace the server object in CPCI & pSOS2.5 while the client intact
   - Compare the performance

Client GUI exchange
   - Replace C++ GUI with JavaBeans GUI developed by JSI

Interoperability, response time and scalability
   - VisiBroker 3.2 for C++ (Solaris 2.6, NT 4.0, VxWorks 5.3.1)
   - Orbacus 3.0 for C++/Java (Solaris 2.6, NT 4.0)
   - ORBexpress ST for C++ (VxWorks 5.3.1)
CORBA Test System Configuration

MMI (PC)
NT4.0
Various ORBs

CPU : Pentium II  Xeon 450MHz
Memory: 512MB

10/100Mbps Switching HUB

CORBA Server (WS)
Solaris2.6
Various ORBs

CPU : Ultra SPARC-360MHz
Memory: 512MB

Compact PCI
DIO
CPU

VME
CPU
AD
DA
DIO

CPU : AVAL DATA ACP-100
PowerPC 603e 133MHz
Memory: 128MB
pSOS 2.5
VisiBroker 3.2.1 (C++)

CPU : Motorola MVME2600
PowerPC 604 333MHz
Memory: 128MB
VxWorks 5.3.1
Various ORBs
Interoperability & Two-way-call Response Time (1999)

- Interoperability test routine
  - Server: returns a random number (float) when called upon by a client
  - Client: invoke the object 10000 times ($\times 10$) and measure the average elapsed time (in $\mu$sec)

*446 /177 $\mu$sec (Sun-Sun) / (NT-NT) by using interprocess communications

<table>
<thead>
<tr>
<th>Client</th>
<th>VisiBroker (C++)</th>
<th>ORBacus(C++)</th>
<th>ORBexpress</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sun</td>
<td>519 428 734</td>
<td>380 455 389</td>
<td></td>
</tr>
<tr>
<td>NT</td>
<td>492 317 664</td>
<td>413 321</td>
<td></td>
</tr>
<tr>
<td>VxWorks</td>
<td>466 417 768</td>
<td>351 460</td>
<td></td>
</tr>
<tr>
<td></td>
<td>497 327 727</td>
<td>408 328</td>
<td></td>
</tr>
<tr>
<td>Sun</td>
<td>1340 1237 1635</td>
<td>1243 1256</td>
<td></td>
</tr>
</tbody>
</table>

- Using Sockets
  - Without using an event-driven socket library (CAsyncSocket) -> Winsock.dll

<table>
<thead>
<tr>
<th>Client</th>
<th>Socket</th>
<th>Server</th>
<th>Socket</th>
<th>VxWorks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sun</td>
<td>110 [0.0242]</td>
<td>180 [0.0389]</td>
<td>185 [0.0094]</td>
<td></td>
</tr>
<tr>
<td>NT</td>
<td>180 [0.0283]</td>
<td>121 [0.0135]</td>
<td>165 [0.0080]</td>
<td></td>
</tr>
<tr>
<td>VxWorks</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Data Scalability Test

- Comparison of the response Time with varying data size between VisiBroker and ORBexpress-ST for VxWorks 5.3.1 and ORBexpress-RT for VxWorks5.4

- The ratio CORBA::char / socket for two-way calls with varying data size.
Network I/O (NIO)

- VME Master-Slave Board
  - DI/DO 32 pts Each

VME

CPU | NIOC | BRANCH
--- | --- | ---

Optical Fiber Link

RS-485

NIO Slave

DI32pts

DO32pts

Equipment

Optical Fiber Link

RS-485

NIO Slave

DI32pts

DO32pts

Equipment

Optical Fiber Link

RS-485

NIO Slave

DI32pts

DO32pts

Equipment

Optical Fiber Link

RS-485

NIO Slave

DI32pts

DO32pts

Equipment

Optical Fiber Link

RS-485

NIO Slave

DI32pts

DO32pts

Equipment
MUSES Control System R&Ds

- Test of RT-CORBA specifications in real products (VisiBroker & ORBexpress)
  - Processor Resource Control
    - Thread Pools (Static and Dynamic)
    - Priority Mechanisms
      - CORBA priority (0-32767) to native one, VxWorks(255-0), pSOSSystem(0-255)
    - Intra-process Mutexes
  - Communication Resource Control
    - [Protocol properties (other than IIOP]
    - [Explicit Bindings (priority-banded, private connections)]
  - Memory Resource Control
    - [Buffering Requests in Queues]
    - [Bounding the Size of Thread Pools]
- Programmable Logic Controller (PLC) CORBA Object
  - PLCs are Japanese Manufacturer’s Favorite!!
  - Create Abstraction Layer for Future Replacement
**CORBA Mutex**

- `rtMutex->lock()`: Only the “owner” can unlock
- Comparison with VxWorks Native Semaphore functions

<table>
<thead>
<tr>
<th>Mutex</th>
<th>VxWorks Semaphore</th>
<th>VisiBroker</th>
<th>ORBexpress</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTCORBA::Mutex_ptrrtMutex; rtMutex = rtORB -&gt; create_mutex(); for( k = 0 ; k &lt; 1000 ; k ++ ) { rtMutex -&gt; lock(); rtMutex -&gt; unlock(); }</td>
<td>intiSemID; iSemID = semMCreate( SEM_Q_PRIORITY</td>
<td>SEM_INVERSION_SAFE ); for( k = 0 ; k &lt; 1000 ; k ++ ) { semTake( ( semaphore * )iSemID, WAITFOREVER ); semGive( ( semaphore * )iSemID ); }</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>VxWorks Semaphore</th>
<th>VisiBroker</th>
<th>ORBexpress</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average for 1000 times of operations (µsec)</td>
<td>2.092</td>
<td>2.334</td>
<td>2.742</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>0.00625</td>
<td>0.00833</td>
<td>0.005</td>
</tr>
</tbody>
</table>
Thread Pools
- Static Threads (pre-fixed number)
- Dynamic Threads (maximum number can be set)
- Default Priority of These Threads

CORBA Priority Mechanism
- Map CORBA Priority (0 [low]-32767 [high]) to Native Priority
  (ex. 255-0 in VxWorks, 0-255 in pSOSystem)
- Server Declared Priorities
  - PriorityModel policy in Server POA
- Client Propagated Priorities
  - Clients Declare Invocation Priorities
- Check Whether one VxWorks Task with Higher Priority Interrupts the Other with Lower one
CORBA Priority Test (VisiBroker only)

Fig. 1: Task Switching Due to Priorities in CLIENT_PROPAGATED model
### Connection Time with Thread Pool

- **create_threadpool()** *(VisiBroker only)*
  - Time To Connect Between a Client and a Servant *(Average for 10 times).*
  - For Dynamic Threads, Default Timeout is 300 Seconds.
    - Connection Time with Existing Threads and One After Their Disappearance.

#### (A) Server Declared Model

<table>
<thead>
<tr>
<th>Static Thread</th>
<th>Initial Connection (IC)</th>
<th>Subsequent Connections (SC)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average (msec)</strong></td>
<td>4.623</td>
<td>3.308</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>0.008</td>
<td>0.143</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dynamic Thread</th>
<th>IC</th>
<th>SC w. Threads</th>
<th>SC w.o. threads</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average (msec)</strong></td>
<td>4.800</td>
<td>3.326</td>
<td>3.581</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>0.012</td>
<td>0.125</td>
<td>0.152</td>
</tr>
</tbody>
</table>

#### (B) Client Propagated Model

<table>
<thead>
<tr>
<th>Static Thread</th>
<th>Initial Connection (IC)</th>
<th>Subsequent Connections (SC)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average (msec)</strong></td>
<td>4.621</td>
<td>3.491</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>0.016</td>
<td>0.175</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dynamic Thread</th>
<th>IC</th>
<th>SC w. threads</th>
<th>SC w.o. threads</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average (msec)</strong></td>
<td>4.801</td>
<td>3.545</td>
<td>3.801</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>0.010</td>
<td>0.190</td>
<td>0.195</td>
</tr>
</tbody>
</table>
## Comparison bet. VisiBroker & ORBExpress

<table>
<thead>
<tr>
<th>Program example</th>
<th>VisiBroker for Tornado 4.1a</th>
</tr>
</thead>
</table>
|                 | ```
|                 | int argc; 
|                 | char *argv[]; 
|                 | CORBA::ORB_var orb; 
|                 | orb = ORB_init( argc, argv ); 
|                 | ``` |
|                 | ORBExpress-RT 2.3.2 |
|                 | ```
|                 | int argc; 
|                 | char *argv[]; 
|                 | CORBA::ORB_var orb; 
|                 | CORBA::Environment _env; 
|                 | orb = ORB_init( argc, argv, "", _env ); 
|                 | ``` |
| Implementation for stub and skeleton | Create new classes inherited from a generated skeleton code |
| ORB's Proprietary Method | PriorityMapping:Instance() |
| Unsupported Methods | |
| try~catch sentence | VISTORY |
|                    | ```
|                    | { 
|                    | : 
|                    | } 
|                    | VISCATCH( CORBA::Exception, e ) 
|                    | ``` |
|                    | VISEND_CATCH |
|                    | ```
|                    | TRY 
|                    | ```
|                    | { 
|                    | : 
|                    | } 
|                    | CATCH( CORBA::SystemException, e ) 
|                    | ``` |
|                    | CATCHANY |
|                    | ```
|                    | { 
|                    | : 
|                    | } 
|                    | ENDTRY(-1) ``` |
CORBA wrappering for PLCs
- OMRON SYSMAC CVM1/CV Series PLC
- Used for Vacuum Monitoring, RF Controls, and Other Monitoring Device Controls
- Read/Write Shared Memory in PLC by UDP
Software Sharing with JSI
- Create Abeans® Plug-in Program for RIKEN Server
- Localization (Japanisation) of the GUIs
  - Use “Unicode Escapes”
  - Property File as a Dictionary

Connection with Mathematica®
- JLink allows communication with Java program and Mathematica
- Some cares required regarding Main()

Use of More Recent CORBA Specification (2.5 as of 09/2001)
- In 1999, ORB for VxWorks complied with only CORBA 2.1
- Portable Object Adapter (POA) – 2.2
- Object By Value (OBV) – 2.3
- Interoperable Naming Service (INS), Asynchronous Messaging – 2.4
“Japanisation” now easier thanks to Unicode (but not complete)

- `PropertyResourceBundle` class to Switch the Languages
  - English: `<class>.properties`
  - Japanese: `<class>_ja_JP.properties`
- Unicode Escapes to Send Japanese via CORBA
  - Example: “abcあいう” → “abc\u3042\u3044\u3046”
- `native2ascii` Command to Modify “properties” File
Connection with Mathematica® (JLink)

Jlink 1.1.2

– Calling Java from Mathematica

If the class is a subclass of java.awt.Window

If not, must create a Frame to hold the content
Connection with Mathematica® (JLink)

- **Jlink 1.1.2**
  - Use Mathematica from Java Program
    - `import com.wolfram.jlink.*`
    - `KernelLink.class`
      - Subclass of `Mathlink.class`
    - `MathLinkFactory.class`
      - Create link objects
    - `Expr.class`
      - Java representation of Mathematica expressions
    - `MathCanvas.class`
      - Subclass of `AWT Canvas` class
      - Display Mathematica graphics
(Before) URL Naming to write Interoperable Object Reference (IOR) files.
- How to manage many IOR files?
- Web server is always a bottleneck

(After) Interoperable Naming Service (INS)
- Naming hierarchy
- Web server no longer needed

*Currently Java client cannot be used with VisiBroker for Tornado 4.1a due to a serious bug (ORB.destroy() would hang the server)*
Database & XML

- **Merits of Using DB in Accelerator Controls**
  - Keep Data Consistency
  - Search Functions
  - Back-Up
  - Common Client Applications

- **Relational Database <-> Object-Oriented Database**
  - Reliability, Market Share, Applications <-> Scalability, OO affinity

- **RDBMS Test (Mainly the Use of XML)**
  - Oracle 8i & MySQL for Miracle Linux
  - JDBC (Unified SQL Commands)
  - XDK tools for Java for Oracle
  - XML Schema, too new?

- **Store Equipment Information in XML**
  - Currently the Info. for Each Equipment is partly hardcoded and rest stored in CSV files.
  - Get the Data in terms of XML from DB
  - Parse in VME with DTD or XML Schema
  - DOM/valuetype CORBA Object
Before
- Data are stored in CSV files
- Properties and attributes are hardcoded

After
- Data values and attributes are stored in XML files
- DB connection more seamless
Conclusions

► Java/JavaBeans for Applications
  – Better portability ensures greater software sharing among labs.
  – Good development tools available
  – Localization easier thanks to “unicode”

► RT-CORBA for Communication Layer
  – Good enough abstraction layer for RTOS environment
  – ORB level asynchronous communication still problem

► CORBA Wrapper for PLCs
  – No unified way for various PLCs
  – Ethernet capabilities also varies for different PLCs

► Others
  – Mathematica® -Java Link
  – Oracle XDK test
  – Interoperable Naming Service (Fatal Bug found in VisiBroker for Tornado 4.1a)