A Framework for Java Application Programs in the CERN PS Control System

Michel Arruat, Jan Cuperus, Marine Gourber-Pace, Roger Hoh, Eric Roux

CERN, 1211 Geneve 23, Switzerland
Objectives of the Framework

- Control 5 interconnected accelerators with an **application layer in Java**.
- **Generic** system, driven by data in a relational database.
- Assembled with proven and **reusable components**.
- **Easy design**, with and without a graphical design tool.
- Application life cycle controlled by an **application manager**.
- Design and operation both in **MS Windows and Linux**.
Class Diagram for an Applic Frame

- JFrame
  - ExtFrame
    - Interface Abstract Frame
      - AbstractApplic
        - Applic
  - Internal or External Frame
    - IntFrame
      - TopZone
      - BottomZone
The Default Empty Frame

The components of the default toolbar are:

- Freeze/Unfreeze action
- One-shot action
- Show/Select dependent frames action
- Date+Time of latest action or update
- Present beam and cycle
- Beam and cycle trigger selector
- Print the frame action
- Popup html browser with applic help text action
A class **Display** has static methods for reporting:
- showMessage(message, source)
- showWarning(message, source)
- showFault(message, source)

According to **source** this is displayed in priority in:
- 1. the bottom zone associated in any way with source
- 2. the bottom zone of the top manager
- 3. as a message pane or
- 4. if there is no graphical interface (yet) on the output stream

In addition, faults are **logged** on a database with JDBC
Application Environment

Accelerators and accelerator devices

Subscription and access service in middleware

Directory service for accelerator devices and parameters

Accelerator device interface

Timing and beam sequence interface

Relational database with JDBC

Reference Parameters for accelerator devices

Error reporting

Frame services

applying and component library
Design Tools

- A text editor is sufficient but a powerful design tool is more effective. **JBuilder** was chosen because:
  - It has powerful text editing capabilities
  - It works on Linux and MS Windows
  - Graphical editing produces clear code
  - It has good debugging facilities
  - It can maintain many projects and switch between them
  - It does not lock you in
Deferred Initialization

- Beans may come to life in JBuilder if completely initialized
- They call static method `deferredInit()`
- In operational environment all these calls are put on a stack with initialization thread as key
- At end of initialization, the AbstractApplic class knows all these beans and can call them with any parameters required by their class and may even automatically interconnect pairs of beans.
Templates

- For **myproject**, skeleton files for main applic can be generated from templates:
  - `Myproject.java`: main applic
  - `MyprojectFrame.java`: for graphical design in tool
  - `MyprojectHelp.html`: shows when help button is pushed
  - `Myproject.jpx`: defines project for JBuilder

- Any time later, skeletons for dependent applics can be added:
  - `Mydep.java`
  - `MydepFrame.java`
  - `MydepHelp.html`
PM Select Project Page

Select or Create a java project name
Select a project category.
Then create a project name or select an existing project in:
- your local project path
- both your local and the central path
- the central repository path
Finally, push 'Goto Project'

Project Category
- ade - Application for accel. ADE
- cps - Application for accel. CPS
- ctf - Application for accel. CTF
- iso - Application for accel. ISO
- lin - Application for accel. LIN
- ln3 - Application for accel. LN3
- mcr - General PS application
- psb - Application for accel. PSB
- app - General application
tools - Designer/Operator tools
lib - Application Interface Library

Project Selector
- tgm
- projectmanager

Selected or created project name: projectmanager

Process - Run local project mcr/test - Terminated successfully
PM CVS Page

<table>
<thead>
<tr>
<th>Design</th>
<th>CVS</th>
<th>Test+Install</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Repository</td>
<td>Add Directories</td>
<td>Add Files</td>
</tr>
<tr>
<td>Update Working Copy</td>
<td>CVS Commit</td>
<td>Revert Working Copy</td>
</tr>
<tr>
<td>Show Summary Diff</td>
<td>Show Selected Diff</td>
<td>Show All Logs</td>
</tr>
<tr>
<td>Show Selected Logs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Status and Conclusions

- **Early version in use** for over a year as stand-alone java programs under C++ console manager.
- Gradual move to applications in a **single Java Virtual machine** under Java console manager.
- JVM must **run for days** without rebooting.
- JVM must update **hundreds of parameters/second**.
- We are confident this will be possible with our **new framework, components, and middleware**.
- Java console manager can call C/C++ programs but ultimately we want a **homogeneous Java application layer**.