Distributed Control Systems at SSRL

Constraints for Software Development Strategies

Timothy M. McPhillips

Stanford Synchrotron Radiation Laboratory

Overview

Computing Environment at our Beam Lines

- Need for cross-platform development.
- Need for distributed architecture.

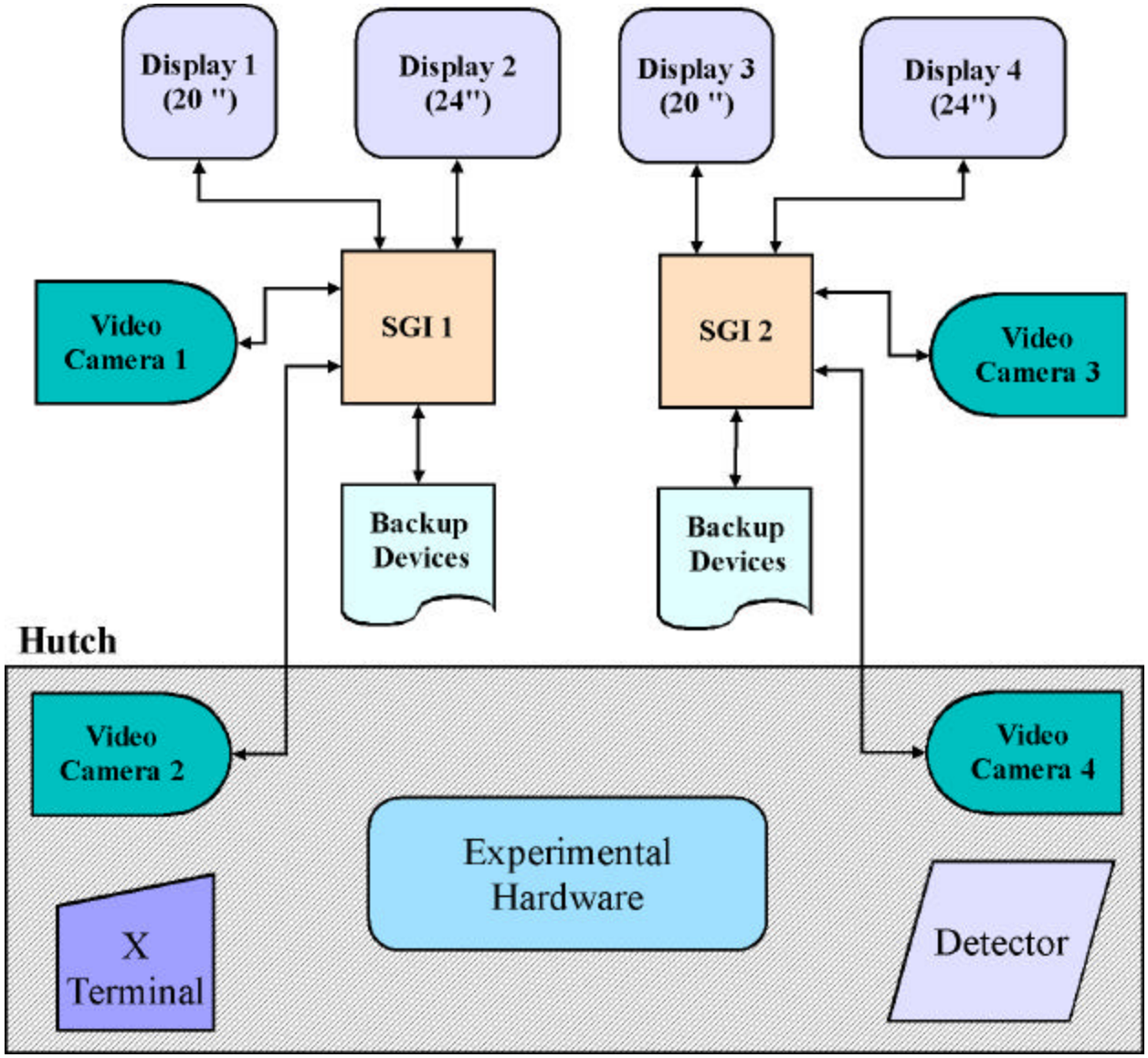
Beam Line Software

- Need for collaborative software.
- Need for high performance software.

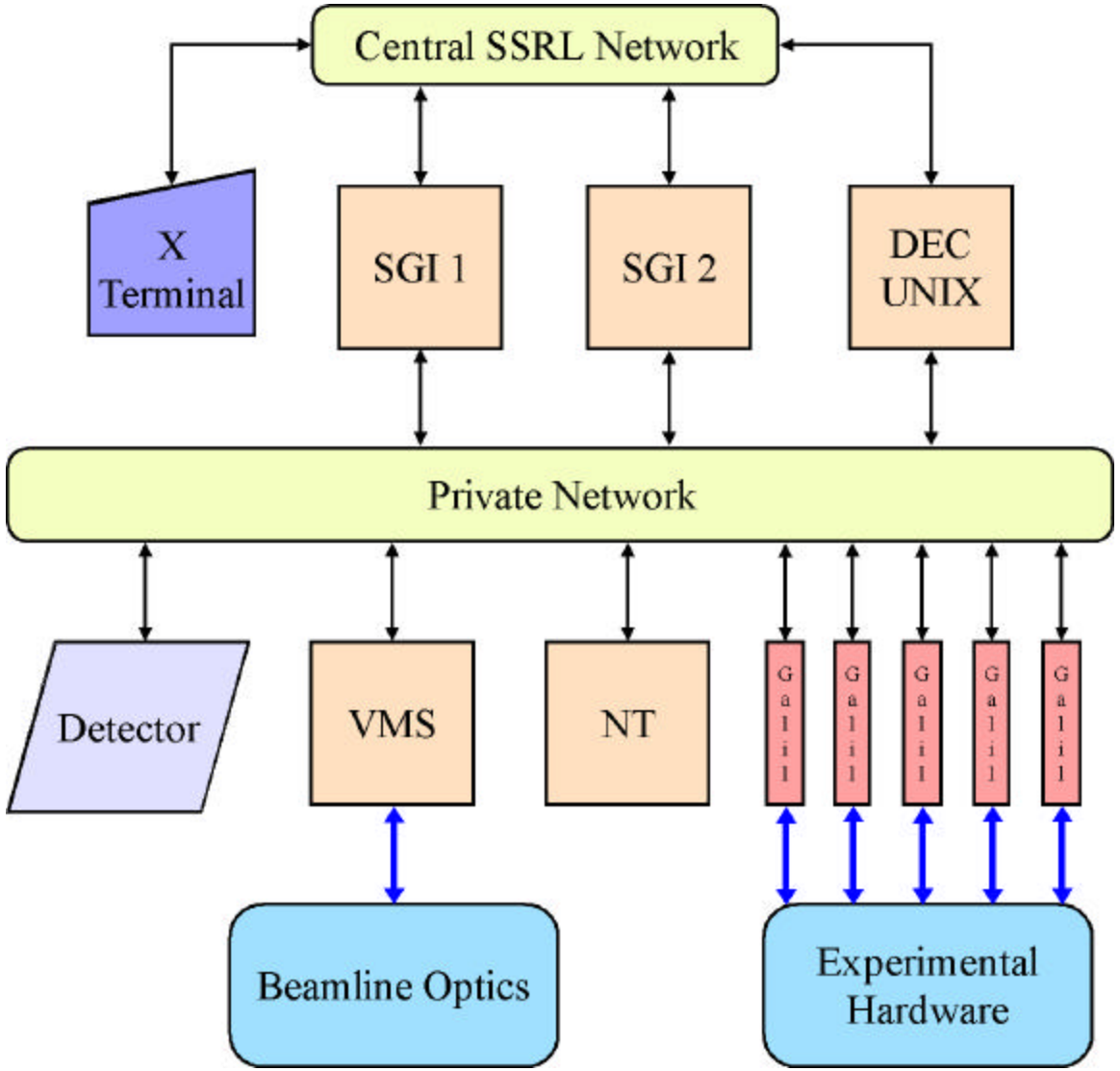
Software Development Strategy

- Constraints on our strategy.
- Our current strategy.

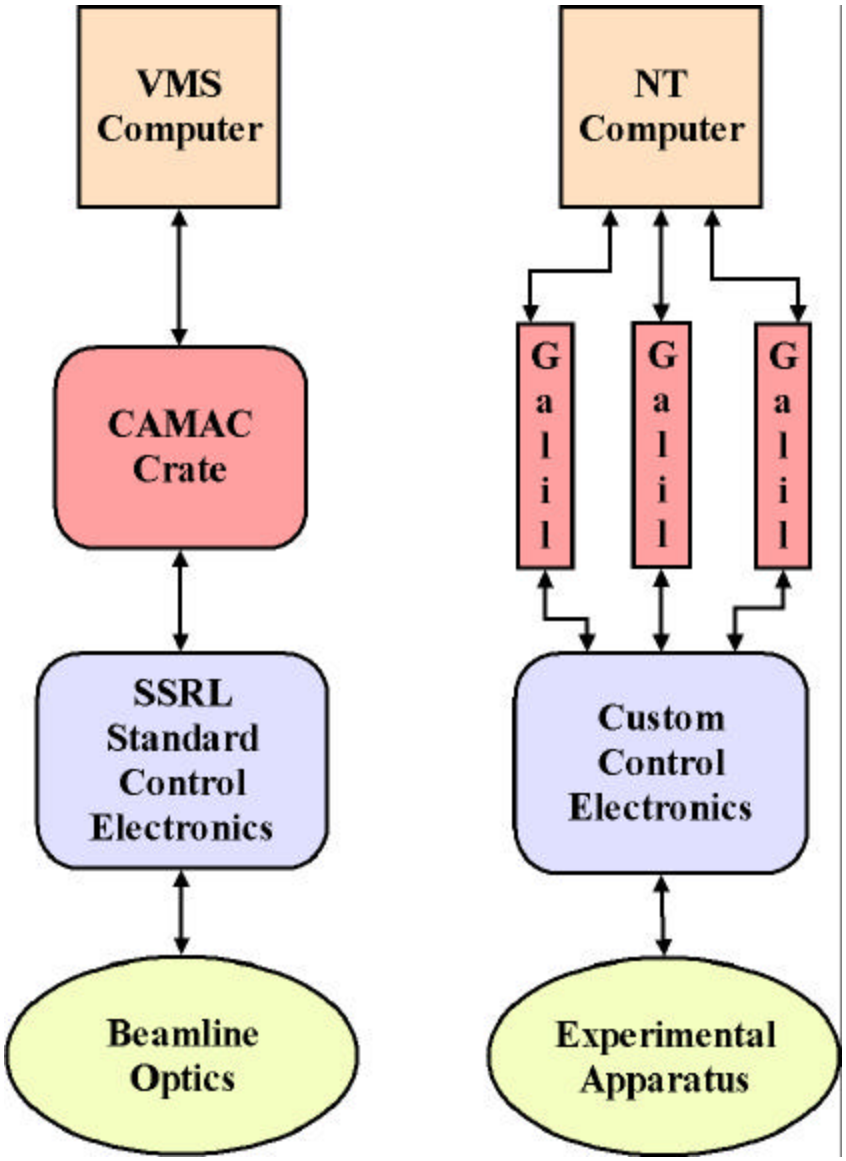
User Environment at Beam Line



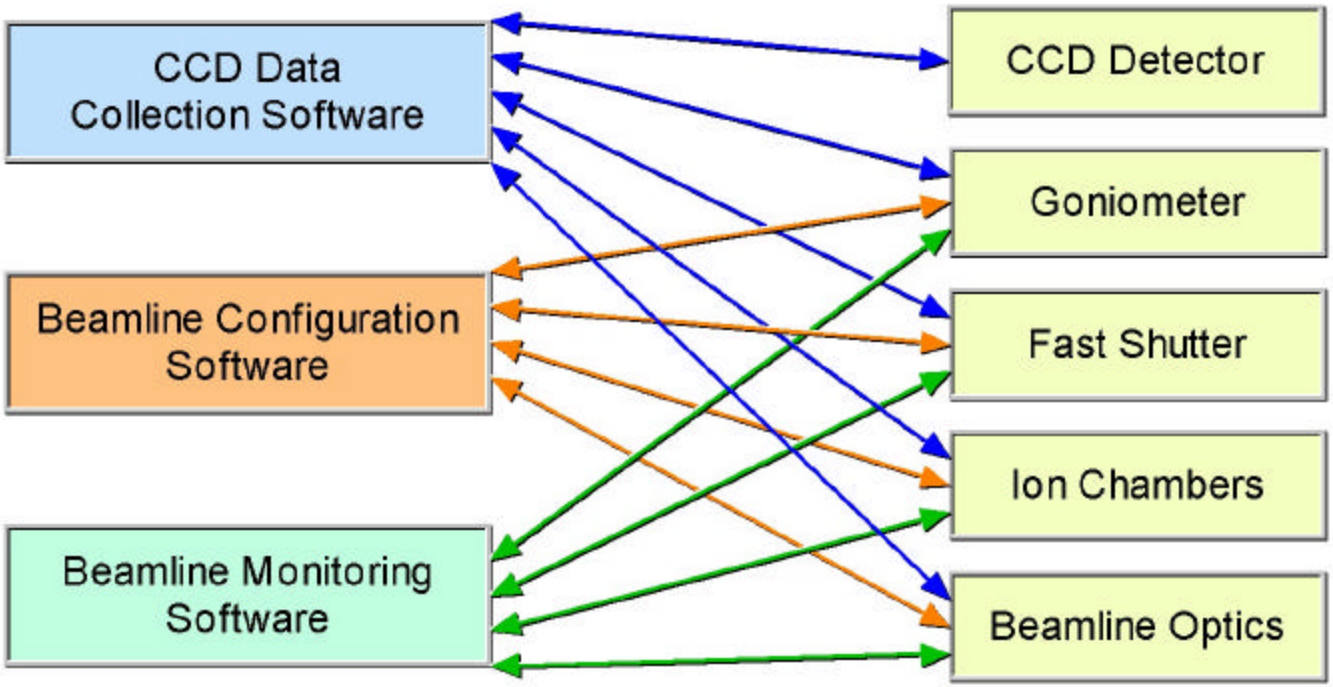
Network Architecture of a Beam Line



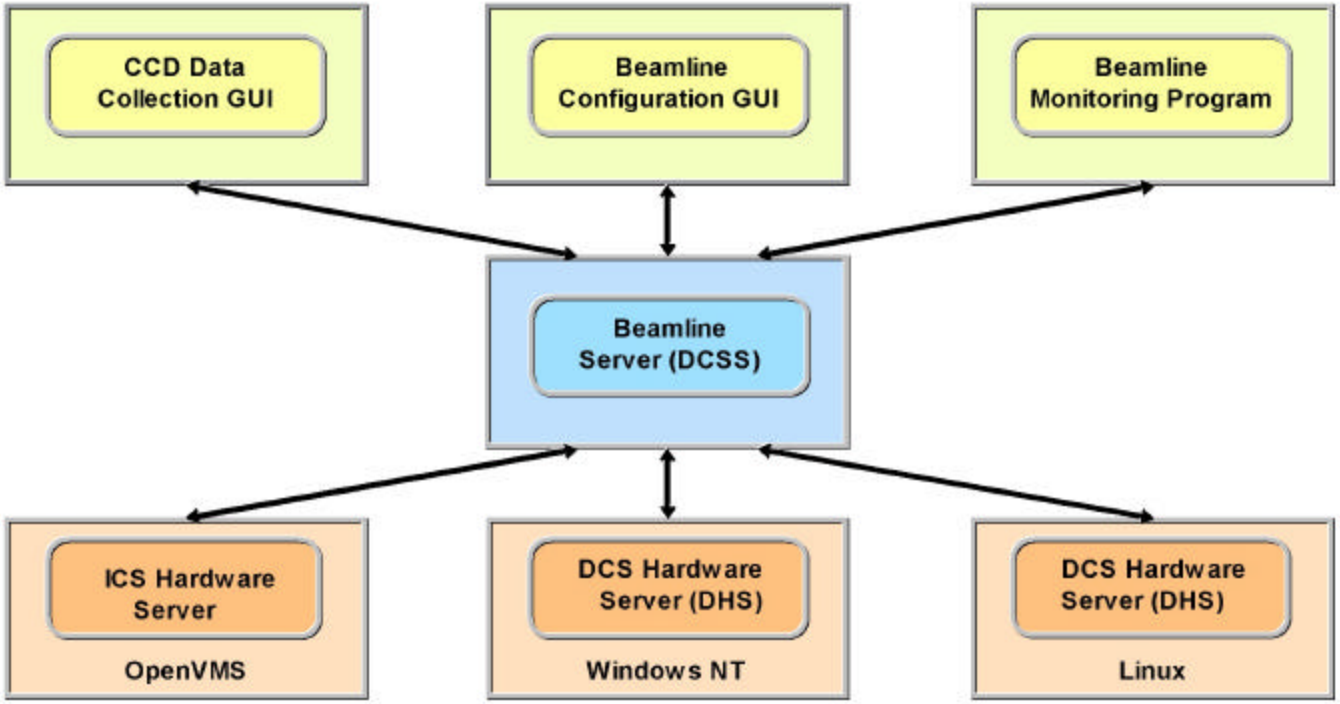
Problem 1: Multiple Hardware Hosts



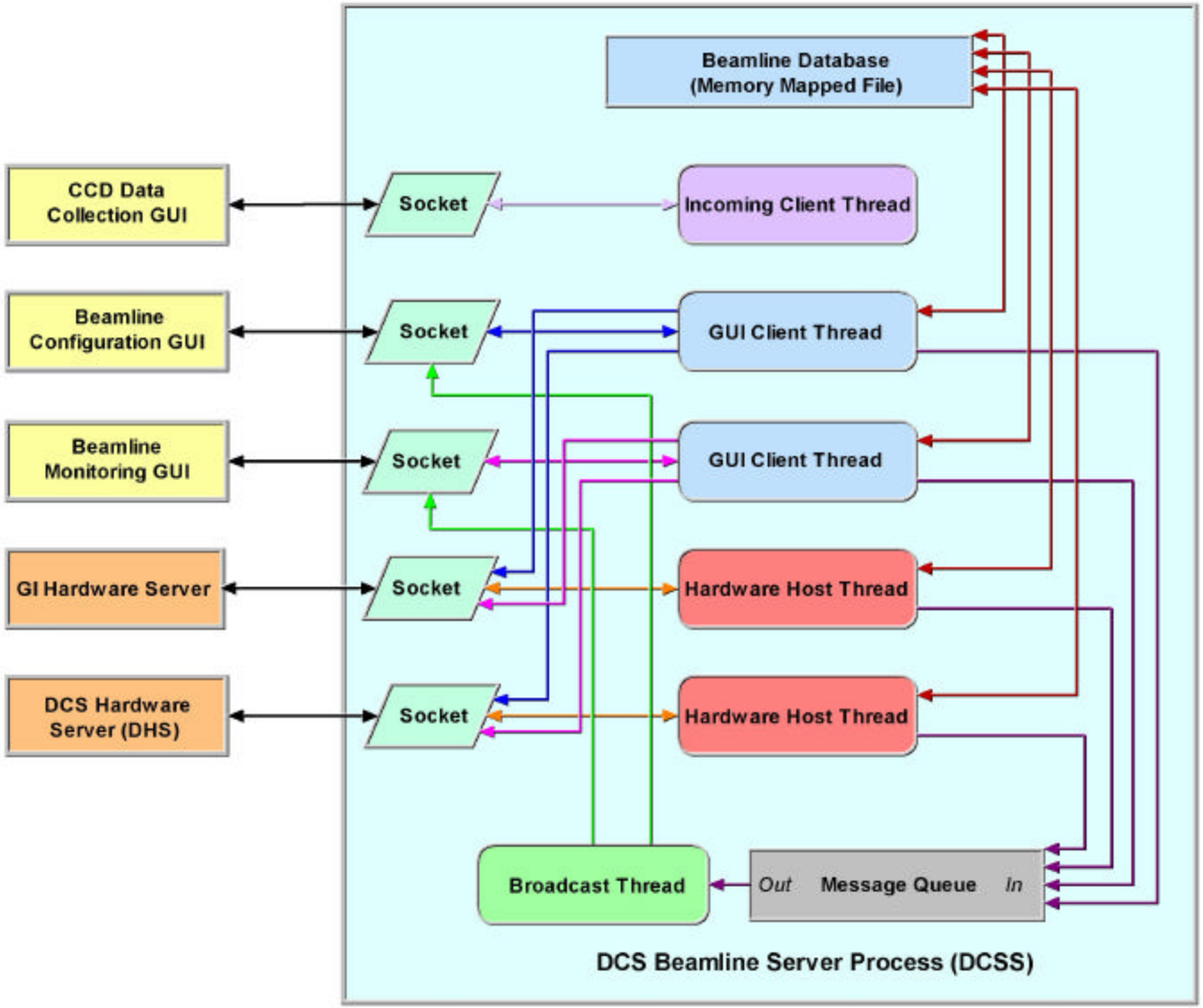
Problem 2: Multiple, Simultaneous User Interfaces



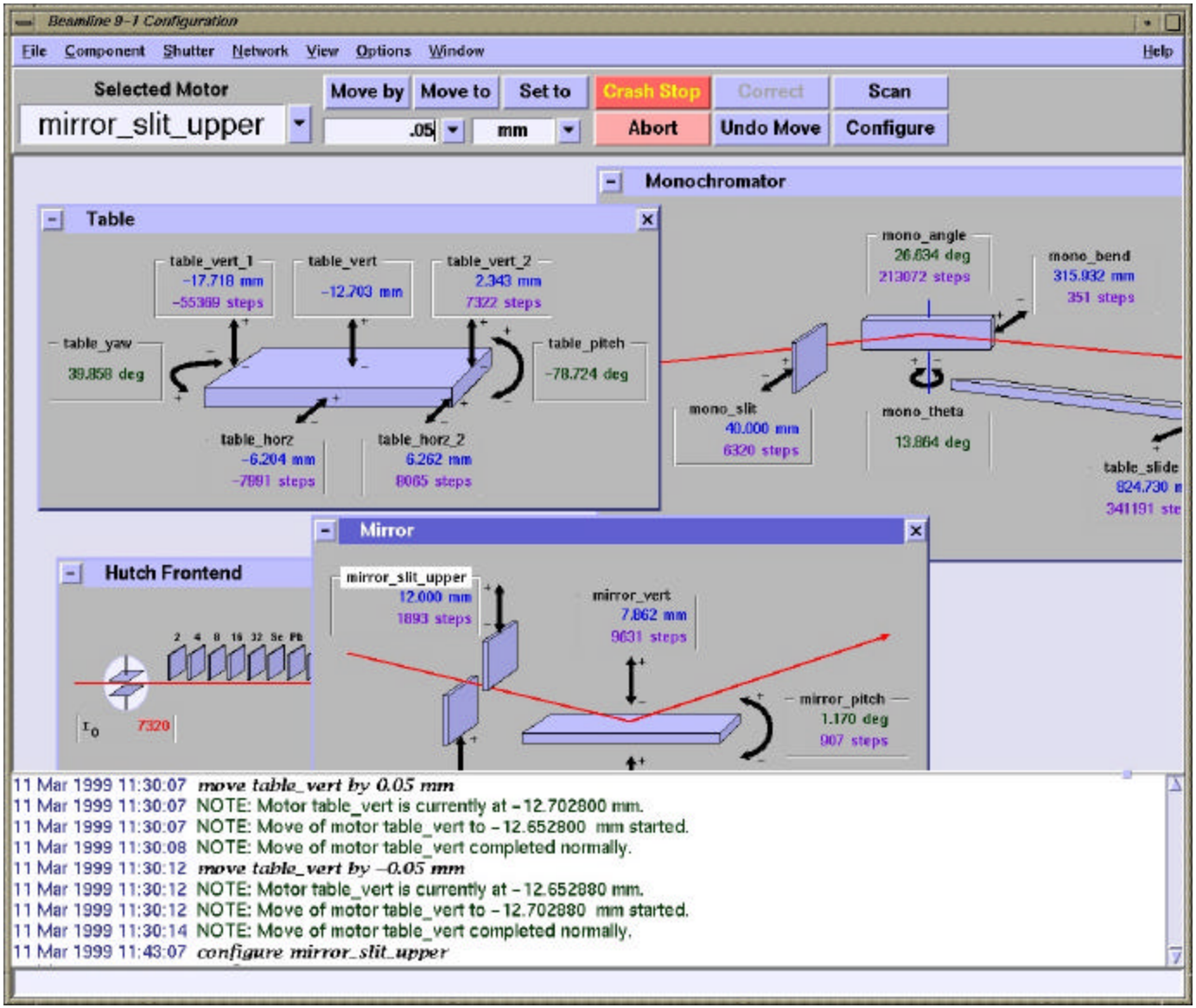
Solution: Distributed Control System (DCS)



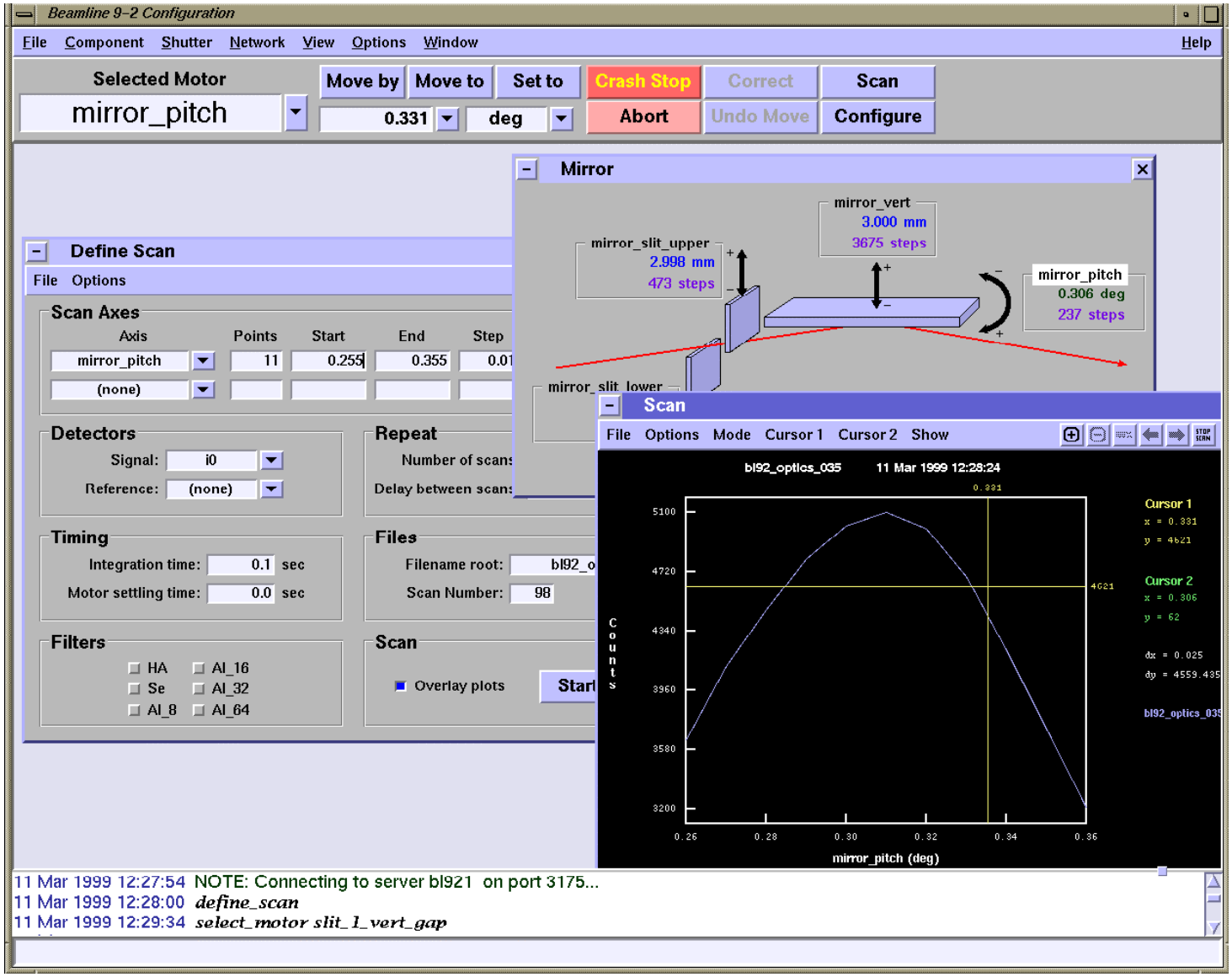
DCS Server (DCSS)



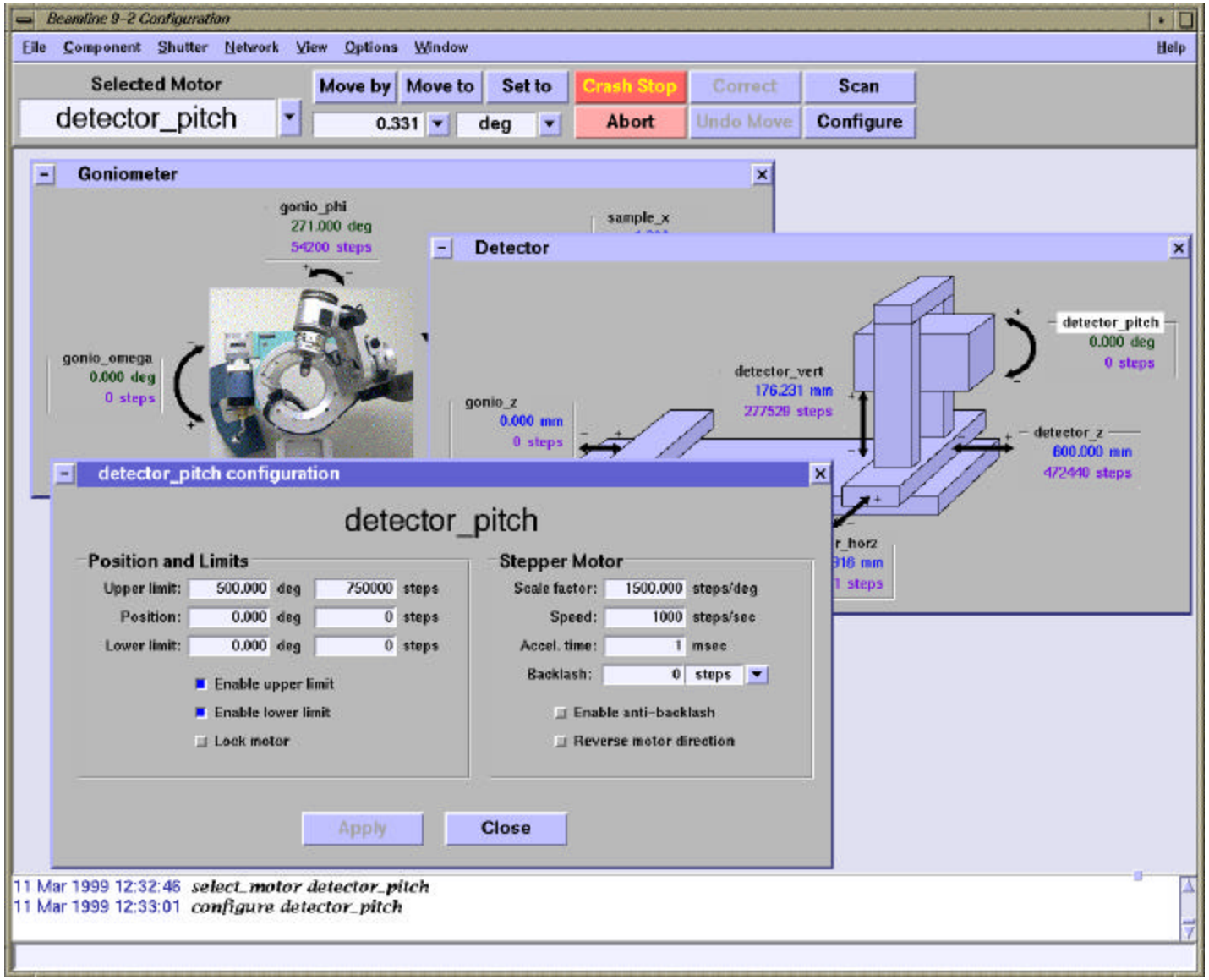
Beam Line Universal - Integrated Configuration Environment (BLU-ICE)



BLU-ICE Motor Scan Windows



BLU-ICE Motor Configuration Windows



Advantages of Writing BLU-ICE in Tcl/Tk

Command Prompt with Scripting

- Tcl was originally designed to be an embedded scripting language, so it is easy to give the user a command prompt and a full featured programming language for scripting.
- User can script any operation in BLU-ICE using control structures, variables, procedures, and even classes.

Platform Independent GUI

- Tcl/Tk runs on any Unix, VMS, Mac, and 32-bit Windows computer.
- Scripts can be distributed without compilation and run on any computer Tcl/Tk has been installed on.
- Scripts can also be bundled with Tcl/Tk binaries and distributed as a single executable file. In this case, Tcl/Tk does not have to be installed on the target machine.

Rapid Development

- Tcl/Tk GUIs can be written with only a fraction of the code necessary in typical system programming languages such as C, C++, or Java.
- GUIs can be quickly written and are easy to maintain in Tcl.
- This characteristic is critical in the rapidly changing environments of our beam lines.

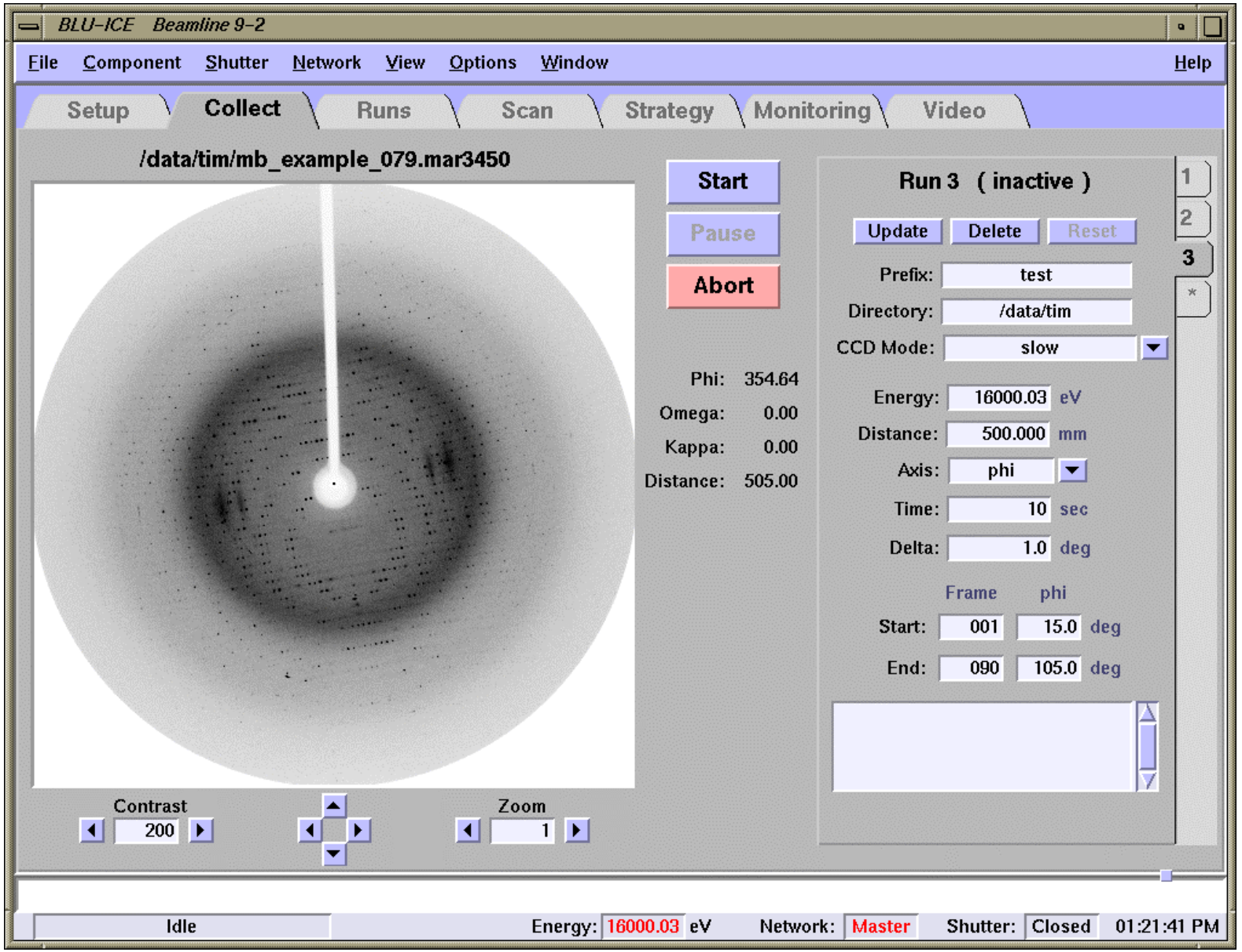
Object Orientation

- The [Incr Tcl] extension to Tcl provides object-oriented features such as classes.
- The [Incr Widgets] extension provides an object oriented framework for building complex widgets from built-in Tcl widgets.

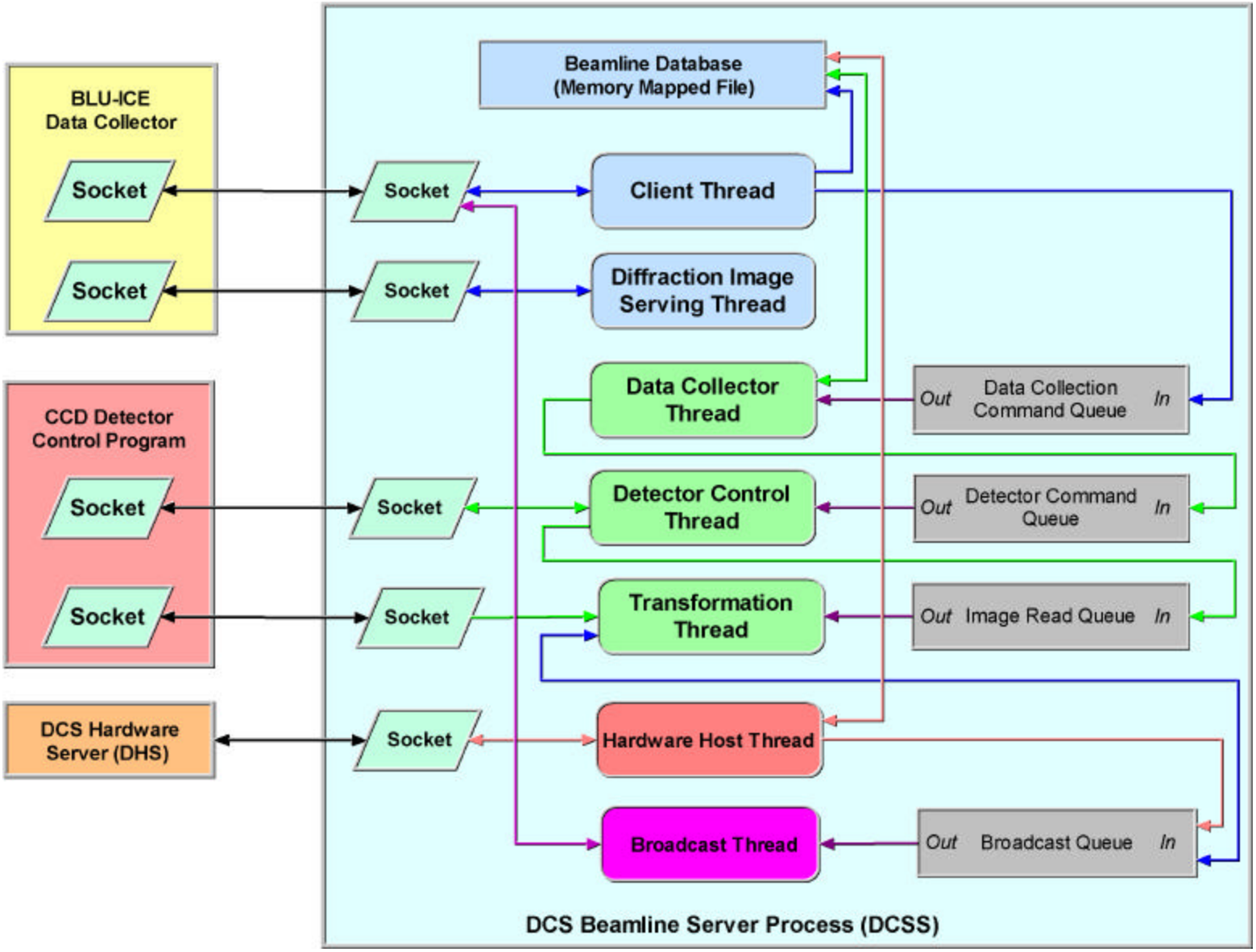
Extensible in C/C++

- Tcl was designed to be extended readily in C. Extensions can be loaded dynamically.
- High performance code, multiple threads and so on are best implemented in extensions.

Data Collection with the New BLU-ICE



DCSS Performance Now Critical



Cross-Operating System Library (XOS)

Features

- **Supports portable, multithreaded, distributed programs**
  - Network communication using a much simplified socket object.
  - Thread creation and synchronization with mutexes and semaphores.
  - Memory mapped files and hash tables.
  - Interthread communication using message queues and Win32-style messages.
- **Compile-time approach**
  - Header file xos.h loads appropriate, system-dependent include files.
  - Objects hide architectural differences.

Advantages

- **Portability**
  - Compile code on Digital Unix, IRIX, OpenVMS, Windows NT/95.
  - Easy to port to new platforms similar to any of the above.
- **Reliability**
  - Simpler APIs leads to more reliable code.
  - Less need to study different platforms.
- **Performance**
  - Native system calls on each platform for maximum performance
  - No runtime overhead for platform independence.

Constraints for Software Development Strategies

Cross-Platform

- Multiple operating systems needed at beam lines; future needs unknown.
- Remote users of the collaboratory may have many different operating systems.
- Other synchrotron labs and even users' home labs may use our software.
- Must support VMS because other SSRL beam lines use it nearly exclusively.

➔ **Use XOS (Cross-Operating System) Library for low-level software and Tcl/Tk for GUI components when feasible.**

Distributed

- Applications must integrate services provided by different computing platforms.
- User interfaces must be kept separate from other components.

➔ **Use TCP/IP socket interfaces between all application components**.

High-Performance Server Processes

- Server software must be extremely fast and take advantage of multiple processors.

➔ **Write multithreaded C++ programs with XOS for portability.**

Open Source

- Must be able to distribute all software freely without licensing issues.
- Installation distributions, source code and documentation should be nicely packaged.
- Document well enough that other groups can use and extend our solutions on their own.

Low Maintenance Overhead

- Write packages in layers that mix and match.
- Wrap packages in clean APIs that do not require knowledge of underlying code.
- Avoid requiring complex infrastructures. Make it easy for novice programmers.