DØ Control System Tutorial

J. Frederick Bartlett

Outline

- Terminology
- Control System Components
- Alarm States
- Process Variable Naming Convention
- Significant Event System
- Detector Configuration Management
- Operator Displays

Terminology

- Process Variable (PV)
  - Smallest unit of control data associated with the detector
    - status, readback, setpoint, parameter, …
  - Referenced by name
  - The primary object of the Channel Access Protocol
- Record
  - The mechanism by which a Process Variable is defined in an IOC
    - Data structure that realizes an instance of a PV
  - Composed of fields with a type (behavior), access rules, and value(s)
    - Scan rate, timestamp, value, alarm severity, …
- IOC (Input/Output Controller)
  - A computer running a set of EPICS routines used to define process variables and implement real-time control algorithms

Terminology

- Channel Access (CA)
  - The communication protocol used by EPICS
- Channel Access Server
  - Software that provides access to a Process Variable using the Channel Access Protocol
  - Usually, an IOC
- Channel Access Client
  - Software that requests access to a Process Variable using the Channel Access Protocol
  - Usually, a host-level computer
- Field bus
  - The electrical medium by which a detector element is connected to an IOC

Terminology

- **Device**
  - **A component of the detector, often an entire module in a crate, that performs a unified, high-level function**
  - **Can have 1 to more than 30 associated PV's**

Control System Components

- Process control sub-system
  - EPICS (Experimental Physics and Industrial Control System)
  - Open source
  - Maintained by HEP community
  - Scalable architecture
  - Provides tools and building blocks for constructing a control system
  - Based upon a transport protocol (Channel Access)
  - Extensive collection of host-level support applications
- DØ-specific extensions to EPICS
  - New drivers
    - MIL/STD1553B field bus
  - New record types
    - HV channel state machine
  - New device support
    - Rack monitor
    - AFE boards

Control System Components

- Field buses
  - VME
  - MIL/STD1553B
  - CANBUS (Run IIB)
- Application processes
  - Channel Access clients
  - Significant Event System
    - Alarms
  - Comics
    - Configuration management
  - Graphical resource displays
  - Data archivers

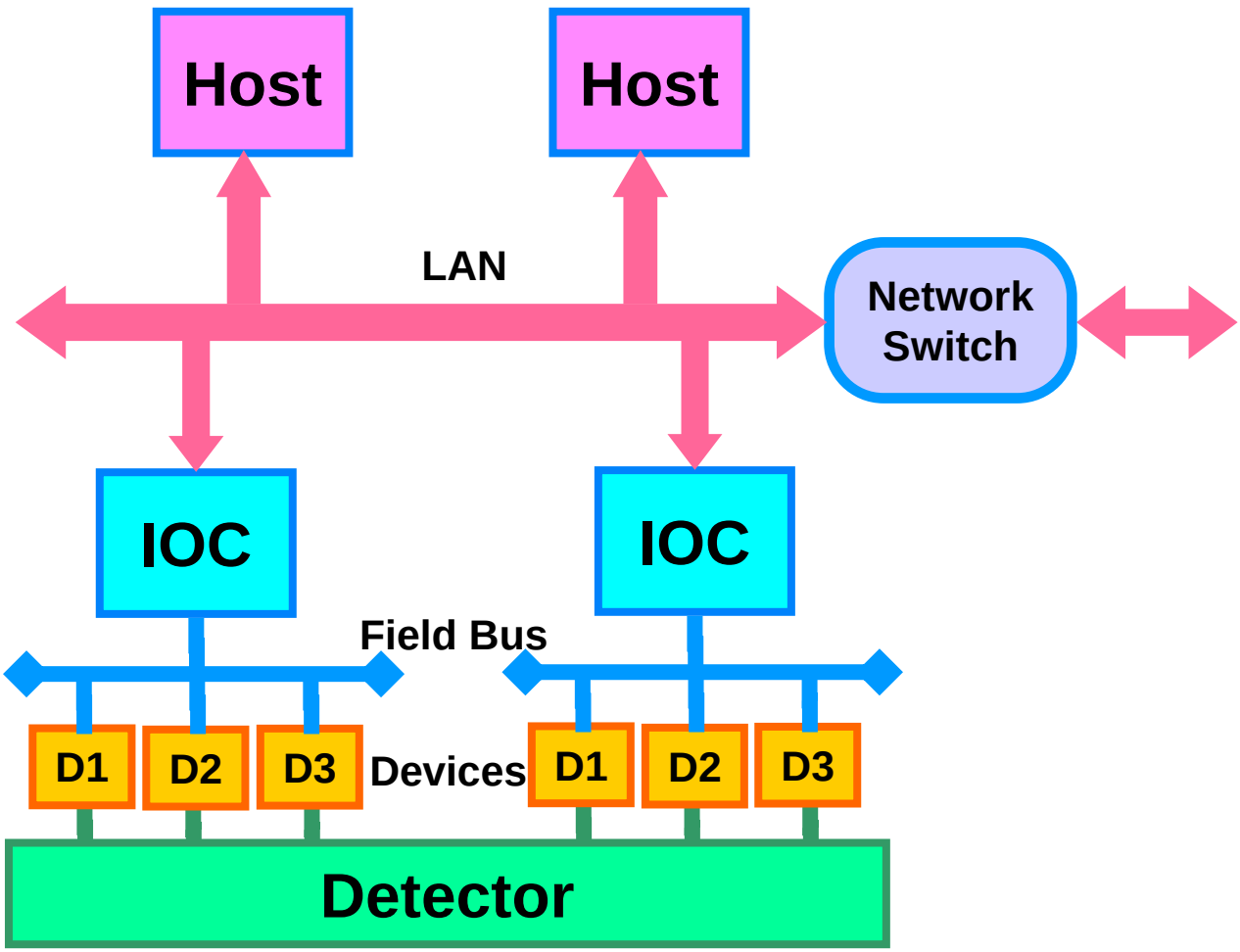
Control System Components

- Size
  - ~15 host-level processors
  - ~150 IOCs (Input/Output Controllers)
  - ~7000 high-level devices
  - ~150000 process variables
- ~20 major detector sub-systems
- Host-Level processes written in Python
- Source management - CVS

Control System Components

- Operating systems
  - Host processors - Linux
  - IOC processors - vxWorks
- Controls staff
  - Core system – 2 ½ FTEs (3 people)
  - Detector-specific components - ~2 FTEs
    - Primarily from other institutions

Control System Components



Host – Linux Servers
IOC – MVME162, MVME23XX, MVME5500
Field Bus – VME, MIL/STD1553B, CanBus

Control System Components

- Accelerator System Gateway
  - Gateway link to ACNET system
  - Bidirectional
  - *Data access only (no control)*
- Cryogenics and Gas Gateway
  - Gateway link to DMACS system
  - Read-only
  - *Data access only (no control)*
- Gateways appear as CA servers (IOC's) to EPICS

Alarm States

- Process variables (EPICS records) exist in one of four alarm states
  - No Alarm [GREEN]
    - Value within normal range
  - Minor [YELLOW]
    - Value outside of normal range but not data corrupting
  - Major [RED]
    - Value outside of normal range and potentially data corrupting
  - Invalid [PURPLE]
    - Value returned by the device is not meaningful
    - Field bus error
    - Network connection lost
- Host-level processes add an additional state
  - Undefined [GREY]
    - Unable to find (connect to) the process variable
- In GUI display fields the background colors indicate the alarm state

Process Variable Naming Convention

Template

<det>[<sub>]_<dev>_<loc>/<attr>[:<io>] [. <field>]

Name Elements

Element Name	Symbol	Example
Detector	<det>	CAL
Sub-det	<sub>	N
Device Type	<dev>	VBD
Locator	<loc>	01
Attribute	<attr>	STATUS
I/O	<io>	W
Field	<field>	SCAN

Example

CALN_VBD_01/STATUS:W.SCAN

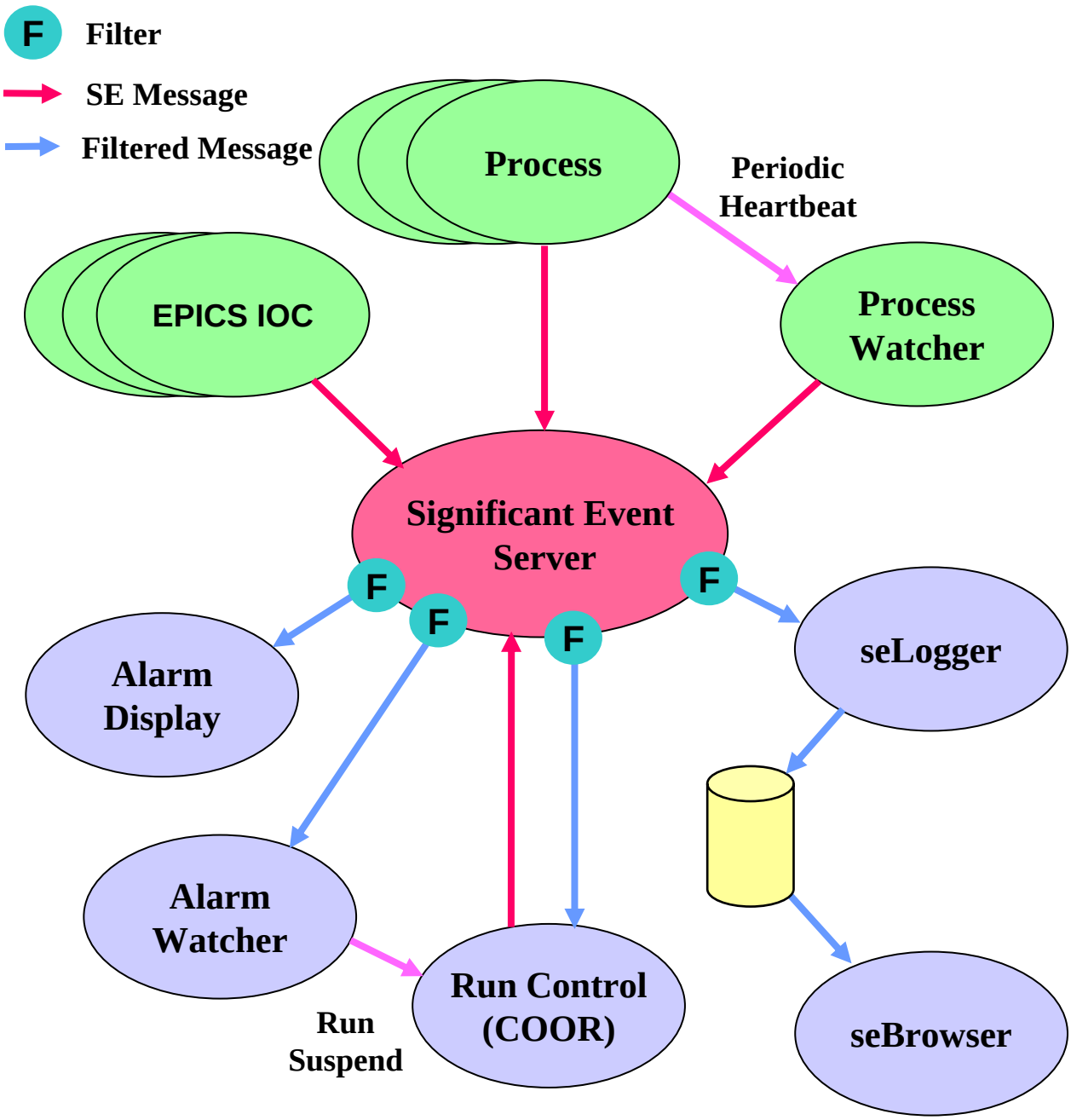
Significant Event System

- The significant event philosophy
  - Alarms are a only a sub-set of the significant events
  - The control system does not generate all of the significant events
  - Alarm utilities enhance reliability
    - Detect impending failures and fix them ***before*** they fail
    - Minimize the time to correct failures
  - Why look at detailed displays until they have something interesting to show?
    - The alarm display shows which detector elements should be viewed in detail
    - No comfort displays, they only clutter the screen
  - Archive all event transitions
    - The archive is a history of the state transitions of the experiment
    - Tools provided to search the event archive

Significant Event System

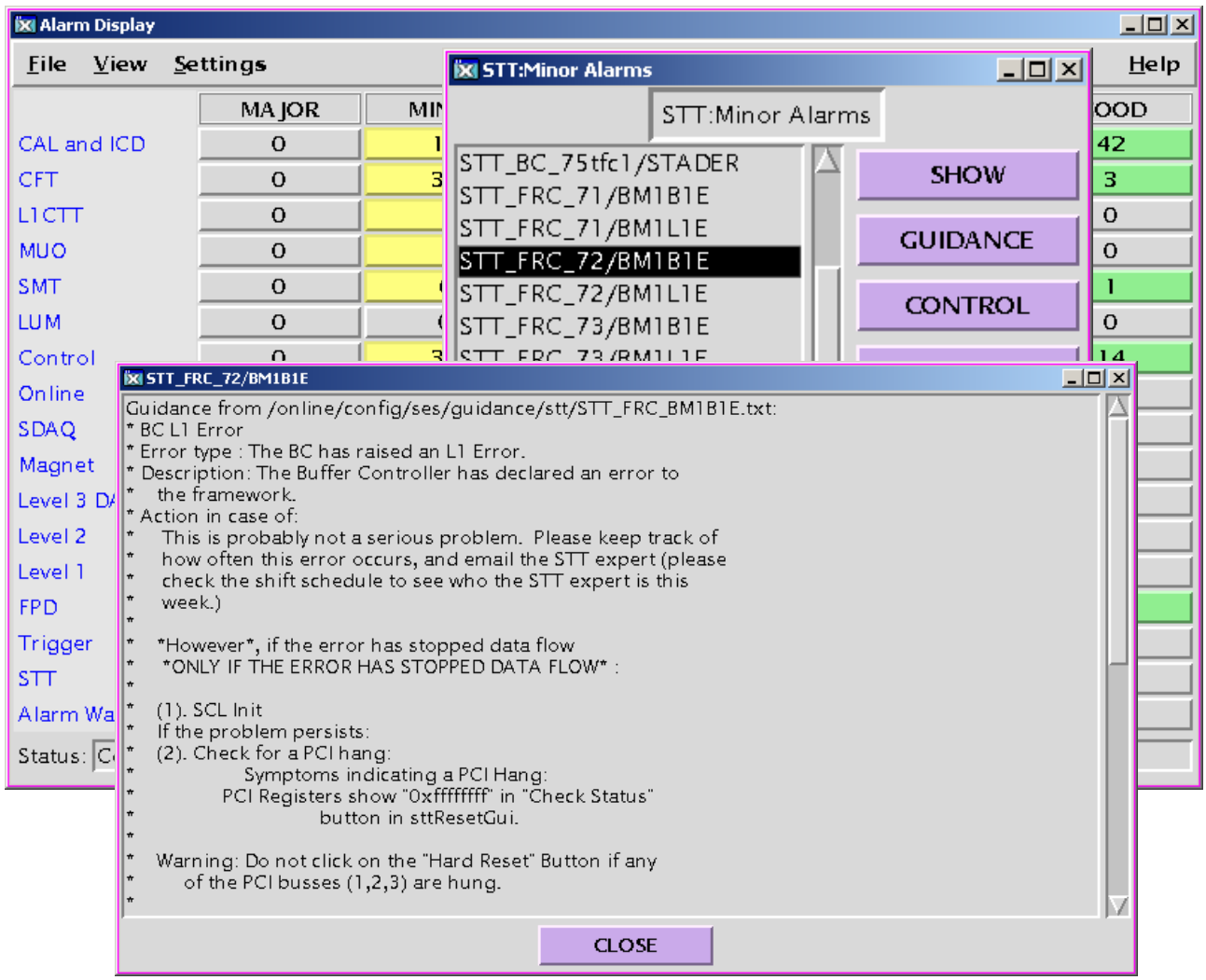
- A server-based event (alarm) system:
  - IOCs and user processes connect to and send alarm transitions to the server
    - Pushed by sources not pulled by the server
  - Server holds the current experiment (alarm) state
  - Server has a filter for each receiving client
    - Makes use of name structure
  - Rapid display startup of receiving clients
  - User processes may also declare events via API (C, C++, Python)
  - Written in Python

Significant Event System



Significant Event System

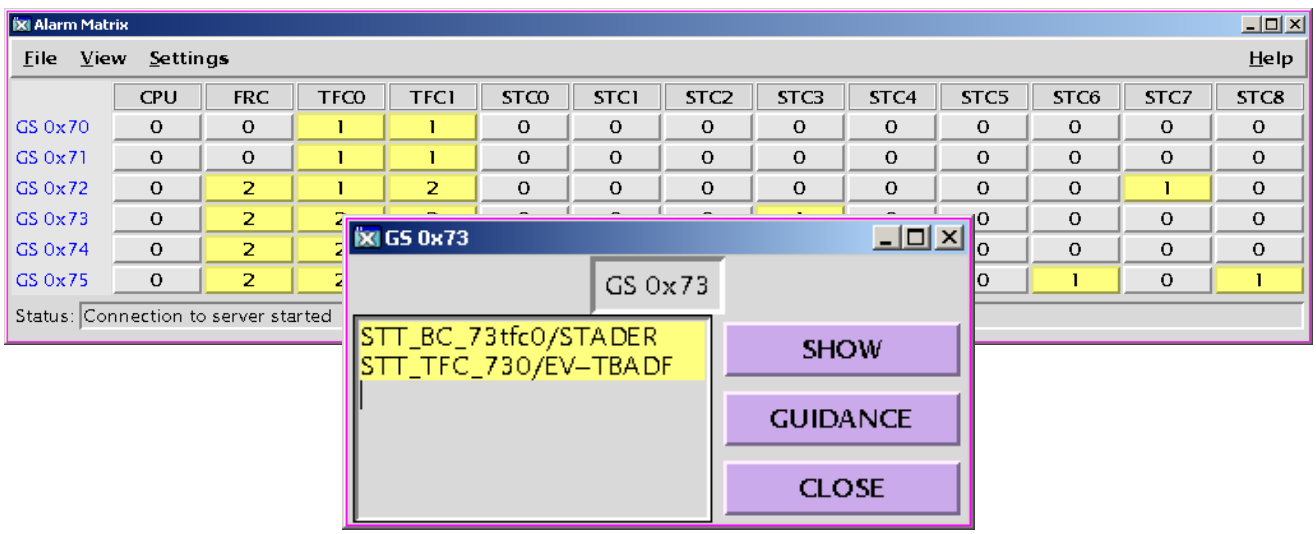
Alarm Table Display



December 29, 2005

Significant Event System

Alarm Matrix Display

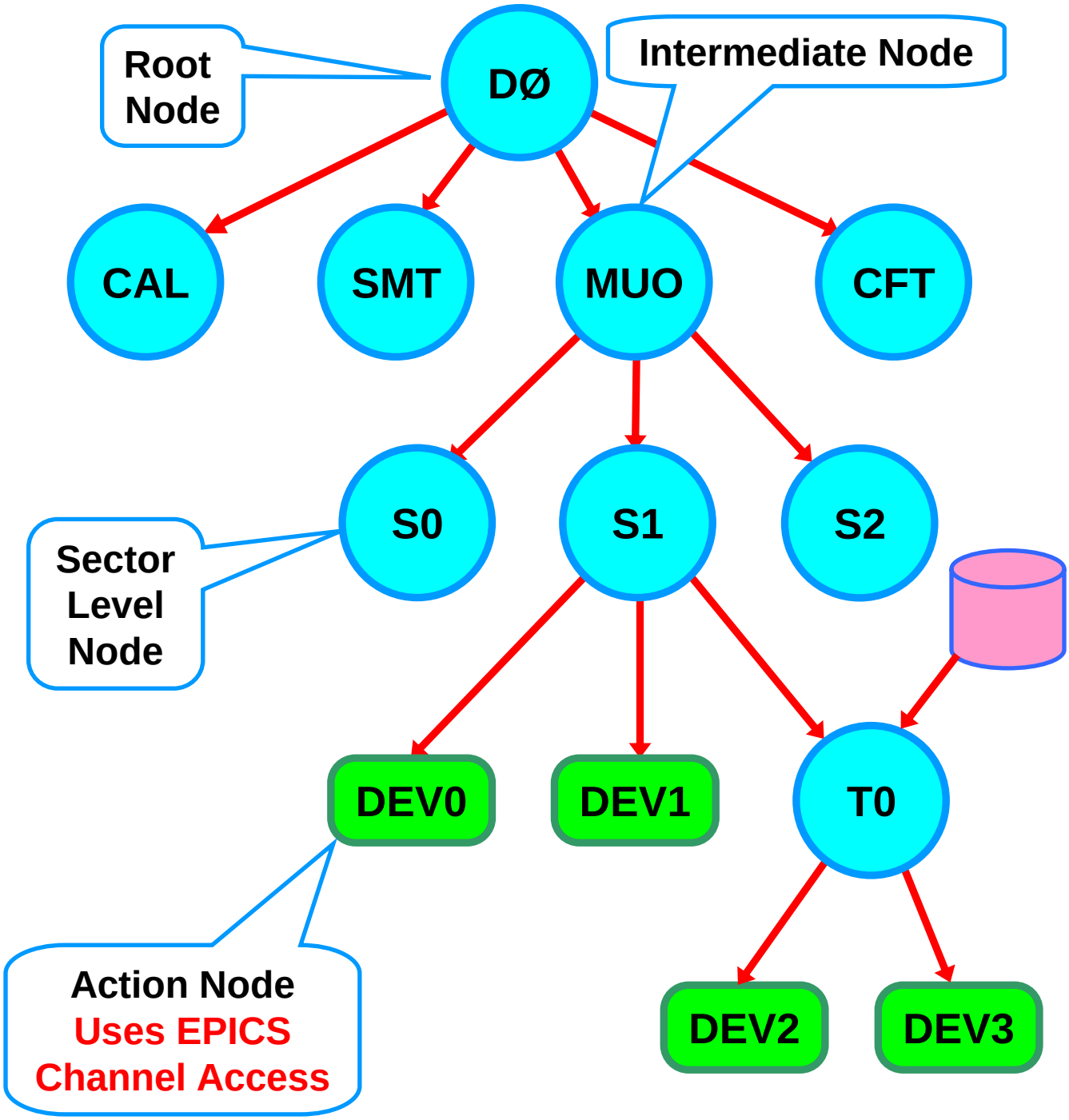


December 29, 2005

Detector Configuration Management

- The COMICS system manages the configuration of the detector
- Configuration map is a tree
  - Directed acyclic graph – no loops
- Tree Nodes
  - Root Node
    - Origin of the configuration tree
  - Intermediate node
    - Establishes a layered hierarchy
    - Establishes an execution order
      - Depth first, left to right
  - Action node (leaf)
    - Performs all control functions (EPICS)
- Constructed on the server model with multiple clients sending commands

Detector Configuration Management

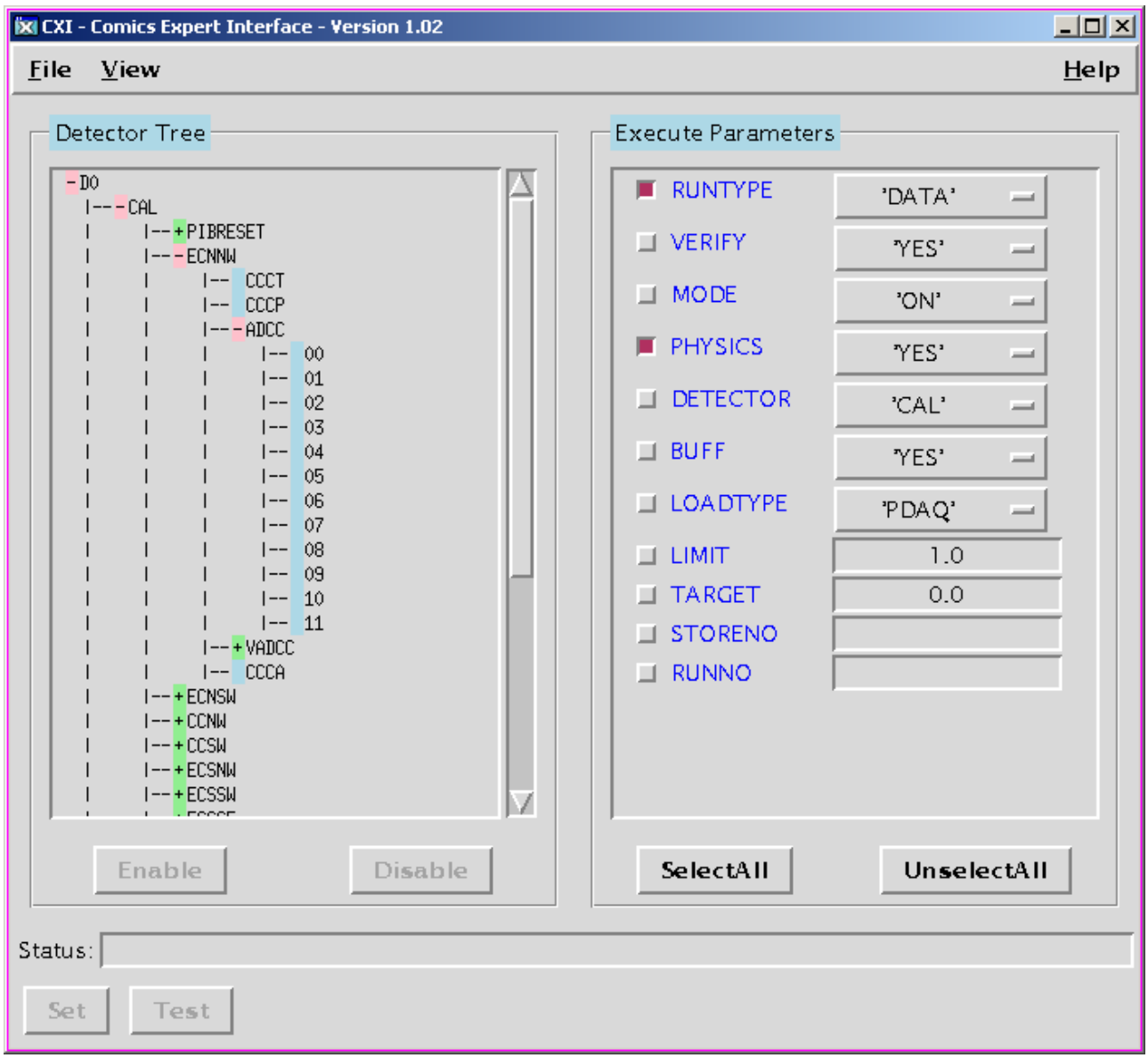


Detector Configuration Management

- Receives sector execution requests from the Run Control Process (COOR)
  - A geographical sector – usually a readout crate -- is the smallest detector component directly managed by COOR
- Server may be activated independently for configuring detector components
  - API (Python only)
  - Shell script (ComicsTalk)
  - Expert Interface (ComicsExpertGui)

Detector Configuration Management

Comics Expert GUI



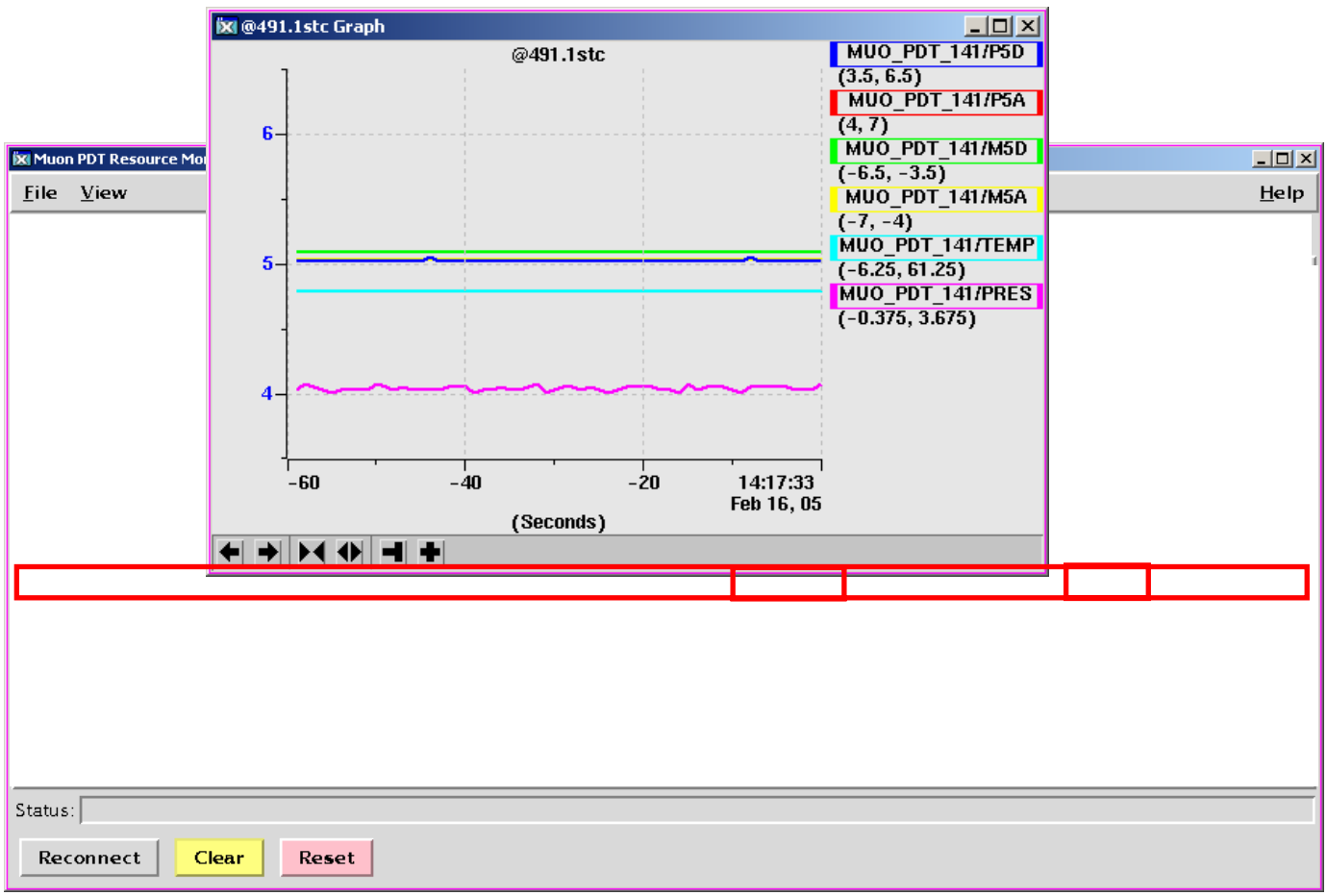
December 29, 2005

Operator Displays

- Standard, process flow (synoptic) displays do not adapt well to the monitoring most of the of the detector components
  - Not related in a serial or sequential fashion like, for instance, a cryogenic plant
- Tabular (spread-sheet designs) are more natural
  - Similar properties for different devices are easily compared
  - Deviations are apparent
- DØ has developed a graphics support library consisting of a series of Python display classes for building tabular displays that collect and display information from EPICS process variables

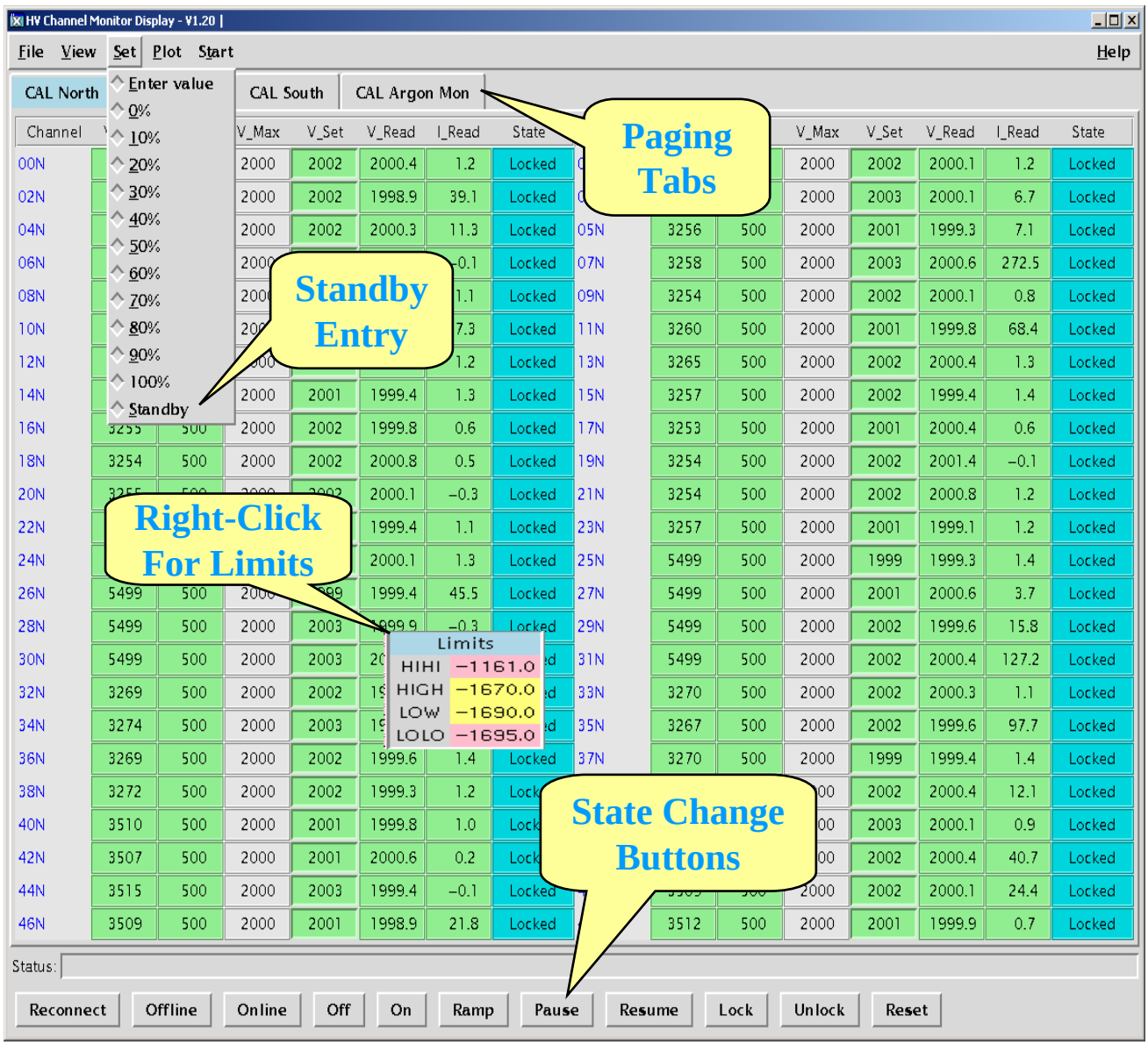
Operator Displays

Resource Display



Operator Displays

HV Channel Display



Operator Displays

Global HV Display