

Operations Analysis Group Software for EPICS Environments

M. Borland
Operations Analysis Group, AOD
March 18, 2004

Reporting on work of
M. Borland, L. Emery, N. Sereno, H. Shang, R. Soliday

Outline

- Brief introduction to OAG
- Basic technologies: Tcl/Tk
 - Why and how we use it
 - PEM automation environment
- Basic technologies: SDDS
 - Why and how we use it
- Data analysis capabilities
- Data logging capabilities
- Process control capabilities

Brief Introduction to OAG

- Group of accelerator physicists and programmers formed in 1995 to “apply the lessons of commissioning to accelerator operation.”
- We automate the operation of APS accelerators.
- We manage the data logging systems.
- We also write accelerator simulation codes.
- We consistently use Tcl/Tk and SDDS.

Tcl/Tk

- Tcl/Tk is our standard scripting language
 - Free
 - Open source
 - Easy to learn
 - Extensible
 - Great for GUIs

OAG Tcl/Tk

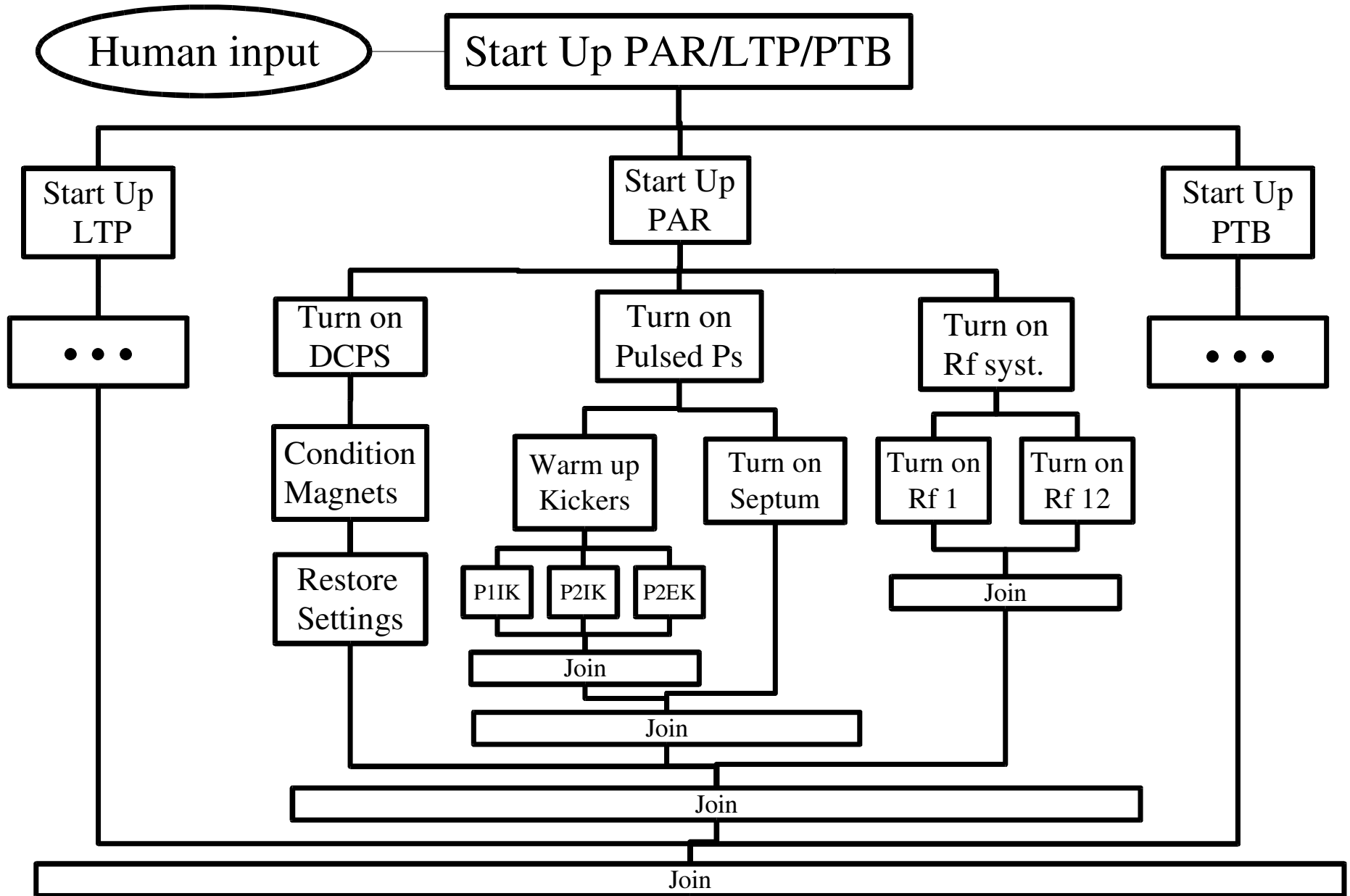
- Tag/value convention for all procedues is critical
 - Instead of
`functionName arg1 arg2 arg3 ...`
 - We use
`functionName -tag1 value1 -tag2 value2 ...`
 - Makes upgrades and reuse easy and robust
 - Code is far more readable
- OAG widget library for common look-and-feel
- Extensions for (among others)
 - Channel access (`et_wish`)
 - SDDS file access
- “Machine procedures” library for accelerator operations

OAG PEM Environment

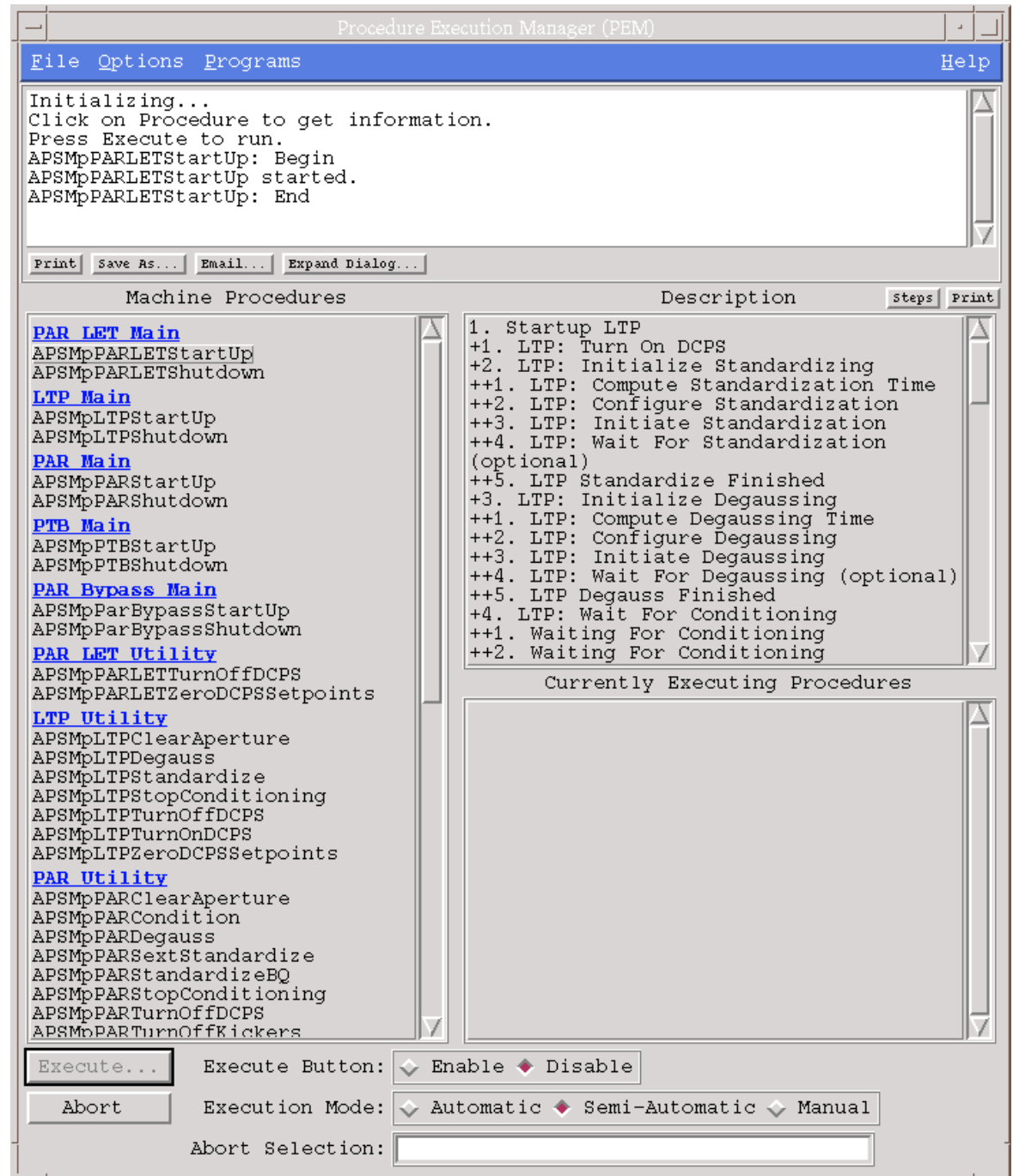
- PEM* (Procedure Execution Manager) is an Tcl/Tk environment used for automation
 - Machine procedure lists
 - Hierarchical, parallel execution
- Critical concept:
 - When a human initiates a procedure, the human must provide “arguments” through a GUI
 - When software initiates a procedure, it must provide the arguments and the GUI is suppressed

*Written by OAG emeritus C. Saunders

Simplified Particle Accumulator Ring PEM Diagram



PAR PEM Panel



SDDS

- SDDS = Self-Describing Data Sets
 - A stable, general-purpose file protocol
 - Generic tools that operate on SDDS files
 - EPICS tools that are configured by SDDS files
 - Libraries for working with such files
- Multiplatform and open-source
 - Solaris, Linux, Windows, OS-X, VxWorks
- Supported languages
 - Shell commandline
 - C/C++, Tcl/Tk, Python, Java, IDL, MATLAB, FORTRAN

Why Use Self-Describing Data?

- Programs that use it are much more robust and flexible
 - Check existence, data-type, units of data instead of crashing or doing something inappropriate
 - Respond appropriately to the data that is provided
 - Exit and warn user, or
 - Use defaults for missing data
 - Data doesn't become obsolete when the program is upgraded
- Data sets can evolve without breaking applications
- Multiple uses for one data set are possible
 - Helps maintain consistent configuration of multiple applications

SDDS File Protocol

- Data model
 - File has a sequence of instances of a structure
 - Structure contains
 - Parameters (scalar values)
 - Table
 - Arbitrarily-dimensioned arrays (little-used)
- All elements are named.
- Meta-data includes units, description, data type
- Options for binary, ASCII, and compressed storage

SDDS Toolkits

- Without the Toolkits, SDDS would be just another boring file format
- Toolkit is UNIX-inspired
- UNIX
 - Everything is a file
 - Programs are “filters” operating on ASCII streams
 - Pipes allow sequencing filters arbitrarily
- SDDS
 - Everything is a self-describing file
 - Programs are operators that transform datasets
 - Pipes allow sequencing operators arbitrarily

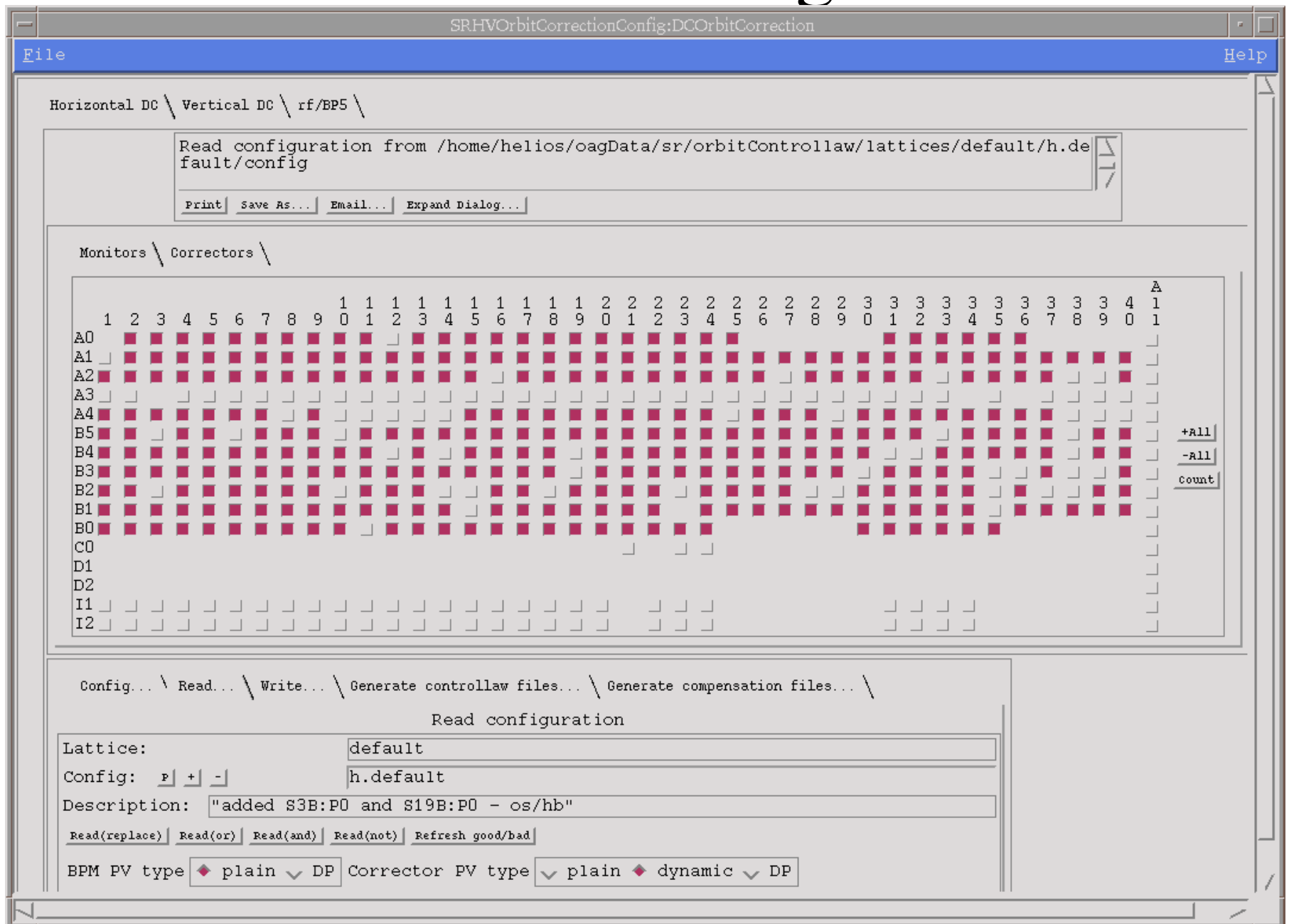
SDDS and Tcl/Tk

- SDDS and Tcl/Tk complement each other
- Tcl/Tk is a good language for GUIs, but
 - Lacks data management capabilities
 - Not great for computation
- SDDS offers data management, analysis, and computation, but
 - Is not a programming language
 - Has commandline user interface

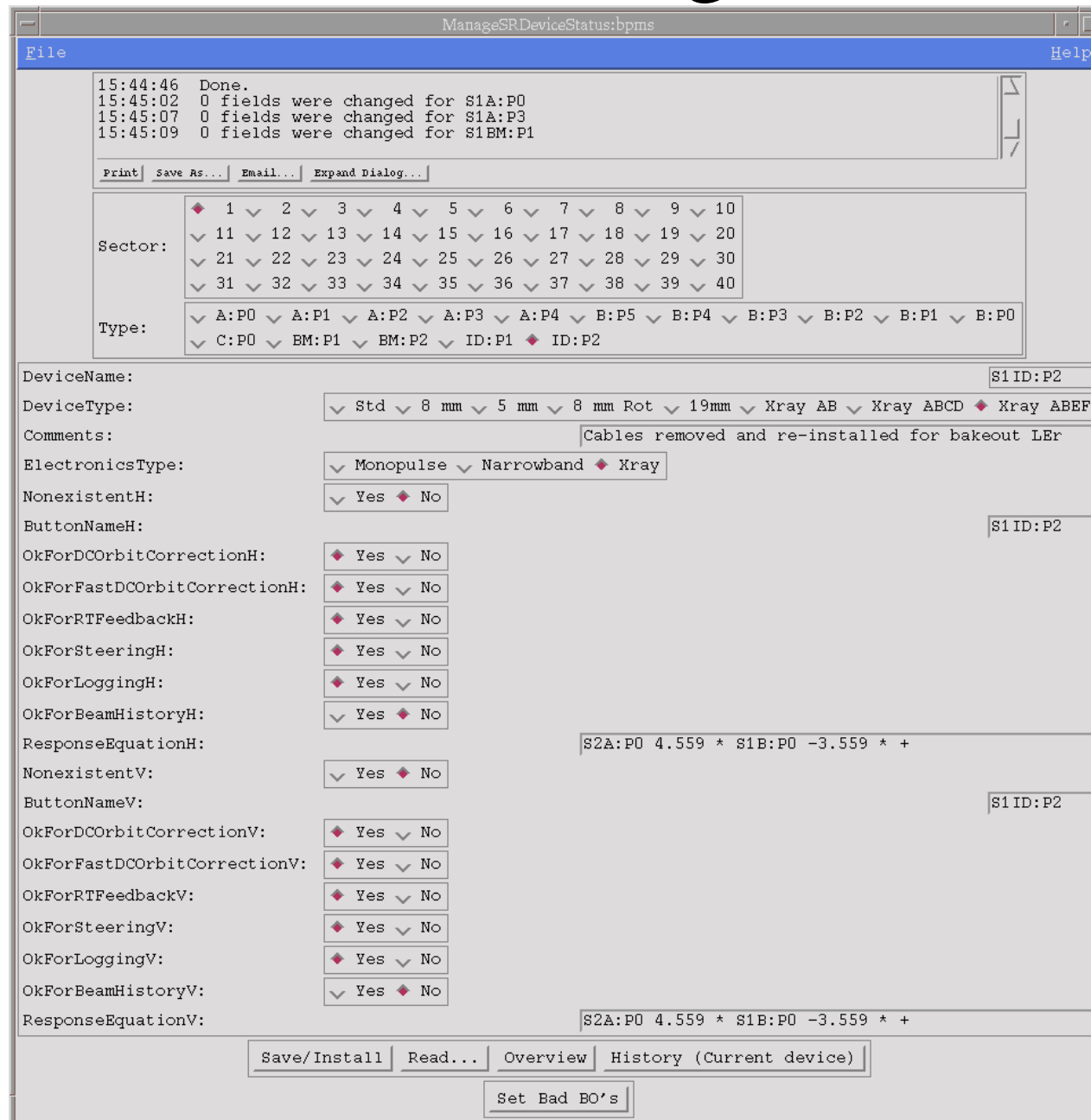
Examples: Orbit Correction

- SDDS-linked Tcl/Tk GUIs for
 - Component status tracking (e.g., “bad BPMs”)
 - Correction configuration management
 - Starting and monitoring processes
- SDDS-configured processes include
 - In-IOC or workstation-based feedback algorithm
 - Permission-to-run testers
 - Feedforward for x-ray BPM correction and fault tolerance
- All data storage and preparation uses SDDS, including
 - Simulation data (response matrix)
 - Measurements (feedforward data)
 - Configurations and configuration history

Orbit Correction Configuration GUI



BPM Status Management GUI



Examples: Save/Compare/Restore System

- SDDS request and snapshot files have PV meta-data
 - System, subsystem
 - Data type (numerical, enumerated)
 - Access mode (read-only, protected, manual-only)
 - Tolerance
- Tag configurations as “preferred” or “reference”
- Compare configurations, or saved state to present state
- Review all or part of a configuration
- Restore/ramp to all or part of a configuration
- 60k process variables are tracked for APS machines

Examples: Setpoint Tracking

- S/C/R only manages discrete configurations stored on demand
- We also monitor changes to “all” setpoints (with deadbands)
- The PVHistory application allows
 - Plot or print history of setpoints
 - Rolling back to any point in the past

MPL Outboard Driver

File Navigate Options

Plot 1 of 1

LTP:V1:CurrentAO

Time starting Sun Mar 14 23:12:41 2004

PV History Tool

File System Help

Loading files
Done loading files
Plotting data
Done plotting data

Print Save As... Email... Expand Dialog...

L5:AUTO:PH:SetpointAO
L5:KY:DC1RF.STTM
L5:KY:DC2ARF.STTM
L5:KY:DC2ARF.TGTM
L5:KY:DC2BRF.STTM
L5:PP:phaseAdjAO
L5:SD:DC1AIQ.TGTM
L5:SD:DC1ARF.TGTM
L5:VX1:TM:ttlTrig2SrcMO
L5:VX1:TM:ttlTrig4PD.DLY
L5:VX1:TM:ttlTrig5PD.DLY
L5:VX1:TM:ttlTrig6PD.DLY
LTP:H1:CurrentAO
LTP:H2:CurrentAO
LTP:H4:CurrentAO
LTP:PH3:SetpointAO
LTP:V1:CurrentAO
LTP:V2:CurrentAO
LTP:V3:CurrentAO
LTP:V4:CurrentAO
PAR:RF1:outerConductor
PAR:RF12:envDet1Ch3V
PAR:RF12:envDet2Ch3V
PTB:H4:CurrentAO

Restore Time Stamp
2004 3 15 15:46:0

BPM ChicaneMovers
 Diag Flippers
 FundamentalRF Gun
 HarmonicRF MainPS
 PulsedPS RF
 RFConstants Scrapers
 SteeringPS SteeringSetpoint
 Timing Water

All None

BB L1 L2
 L3 L4 L5
 L6 LEUTL LINAC
 LTP LTP1 LTP2
 Laser PAR PB
 PCGun1 PTB1 PTB2
 PTB3 rfGun rfGun1
 rfGun2

All None

Rescan PVs Rescan Active PVs

Show PV History Plot PV History

Use tkstdsplot Split Plot

Strip Initial Values

Review Restore System

Enable Restore Button

Date/Time Range of Interest

Starting date/time (year, month, day, hour): 2004 3 15 00:00:00

TODAY -DAY +DAY -WEEK +WEEK -MONTH +MONTH -YEAR +YEAR

Ending date/time (year, month, day, hour): 2004 3 15 23:59:59

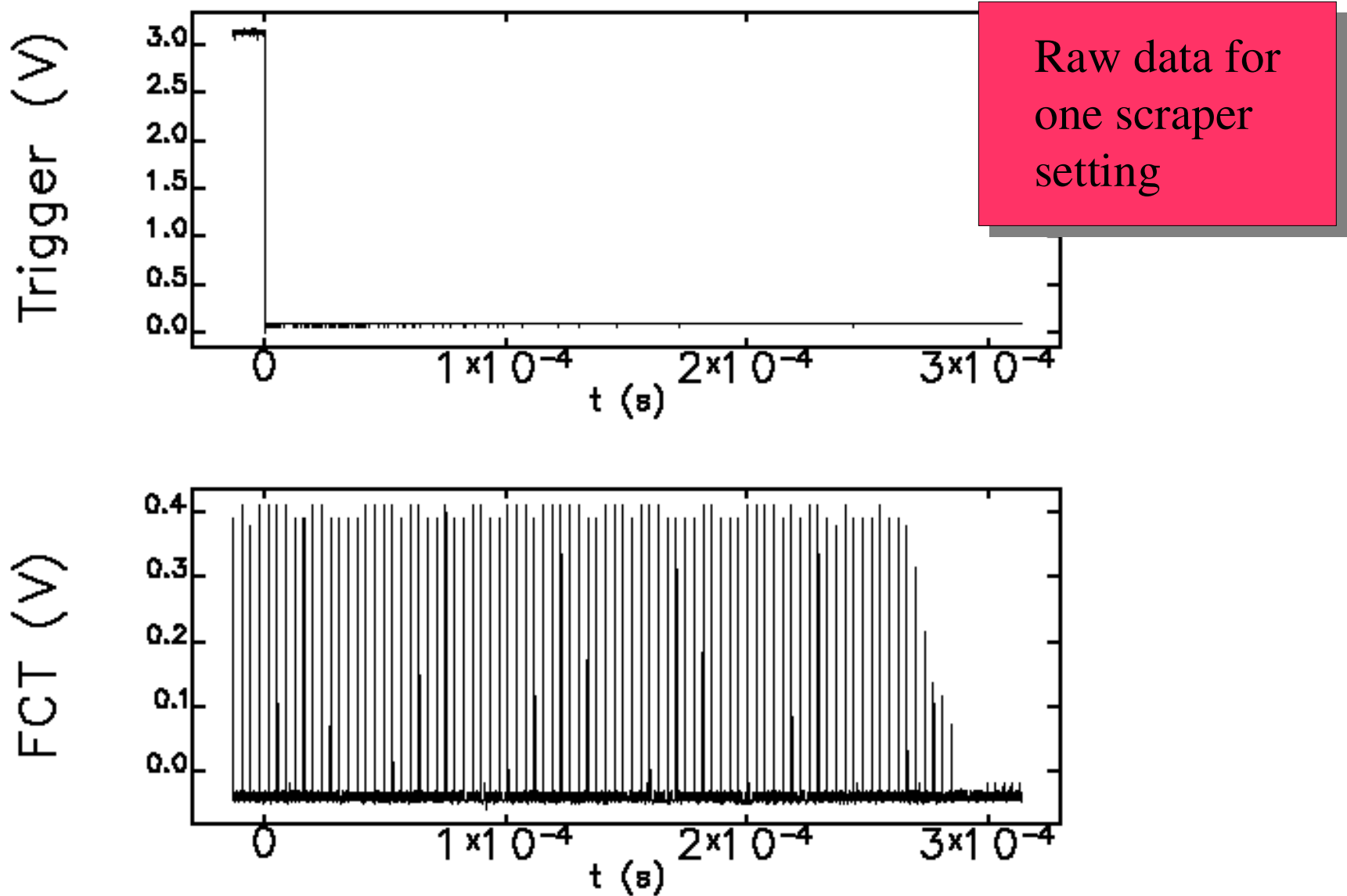
TODAY -DAY +DAY -WEEK +WEEK -MONTH +MONTH -YEAR +YEAR

**Settings History
Review and
Rollback Tool**

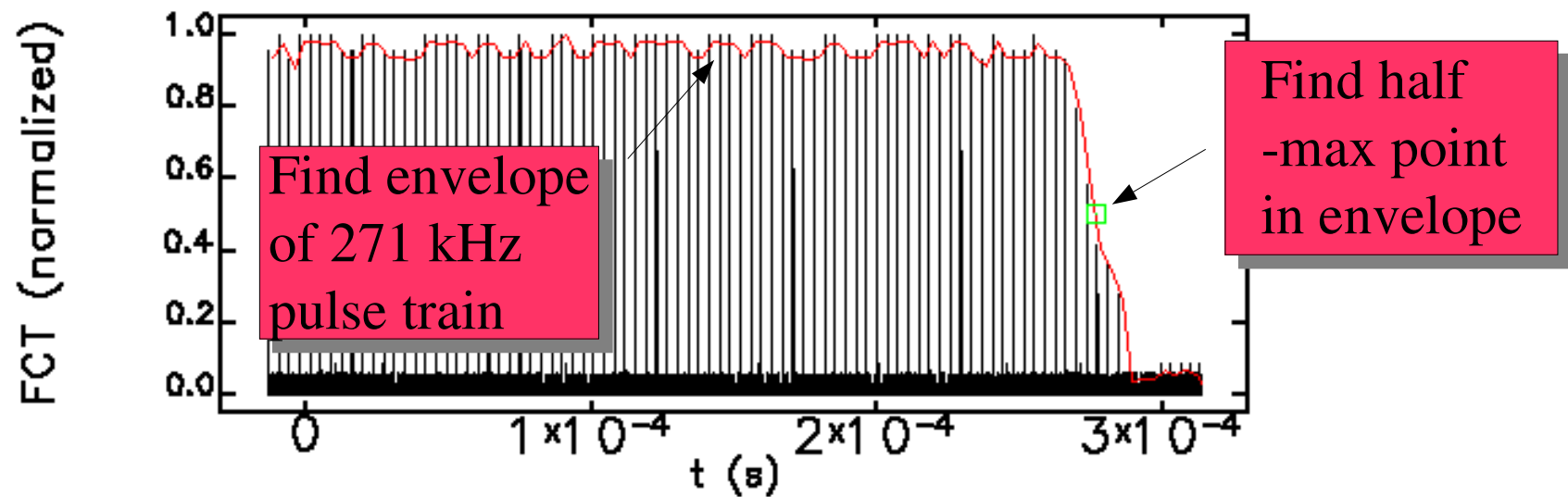
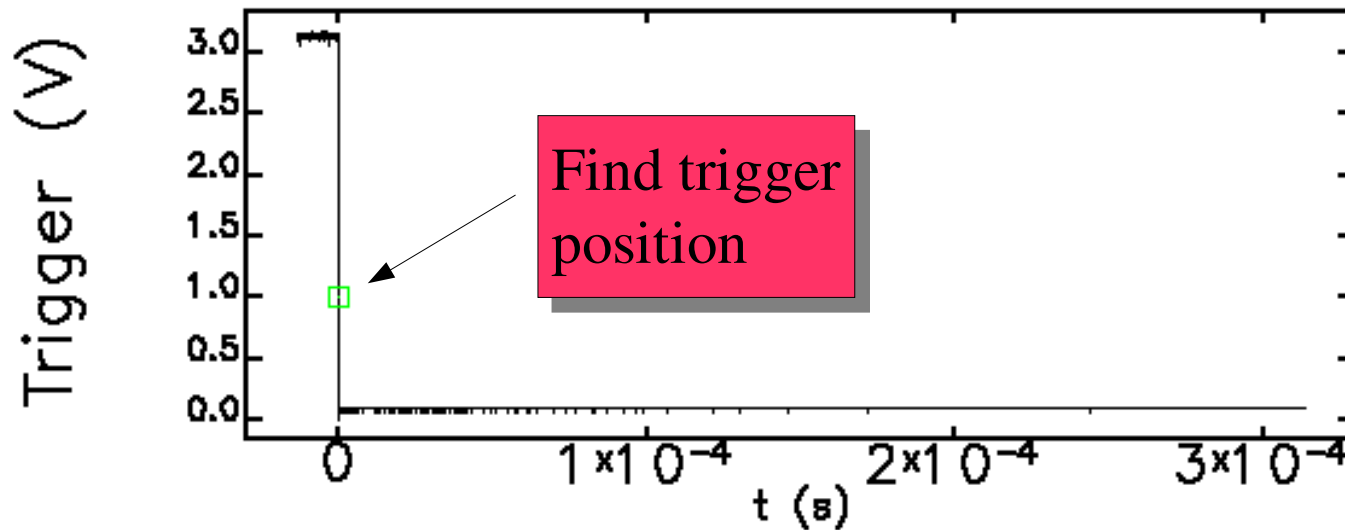
SDDS Toolkit Capabilities

- Display: graphical and text output
- Math: integrate, differentiate, interpolate, normalize, smooth, peakfind, zerofind, evaluate equations, remove baseline
- Matrix: pseudoinverse, transpose, arithmetic operations
- Statistics: correlate, histogram, outlier removal, envelopes, column statistics, running statistics
- DSP: (de)convolution, filtering, FFT, NAFF
- Fitting: exponential, polynomial, gaussian, user-defined
- Manipulation: filter, match, sort, merge, collapse/expand tables, cross-reference/select
- Similar to MATLAB or IDL, but free and open-source

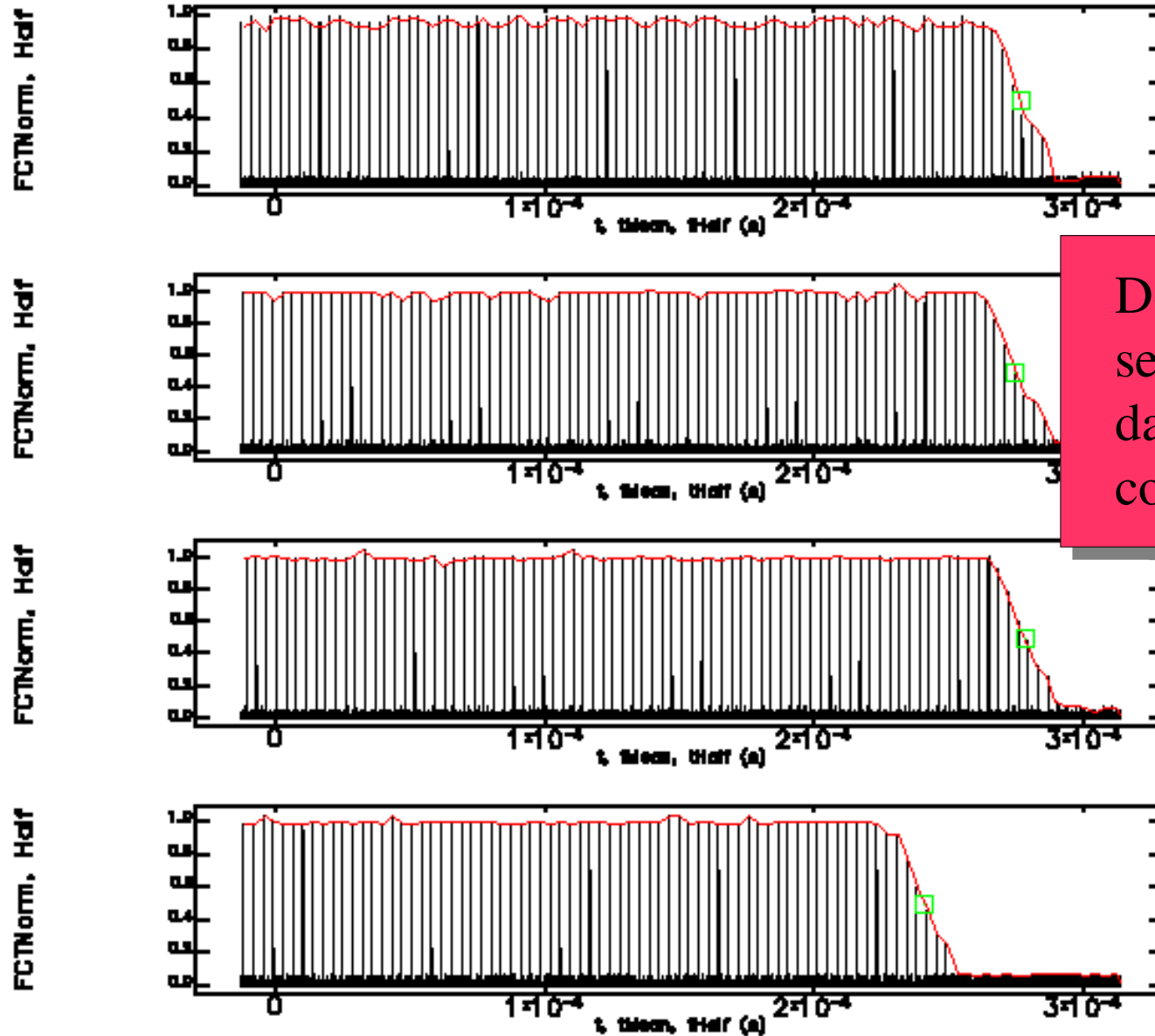
SDDS Processing Example: Beam Dump Analysis



SDDS Processing Example: Beam Dump Analysis

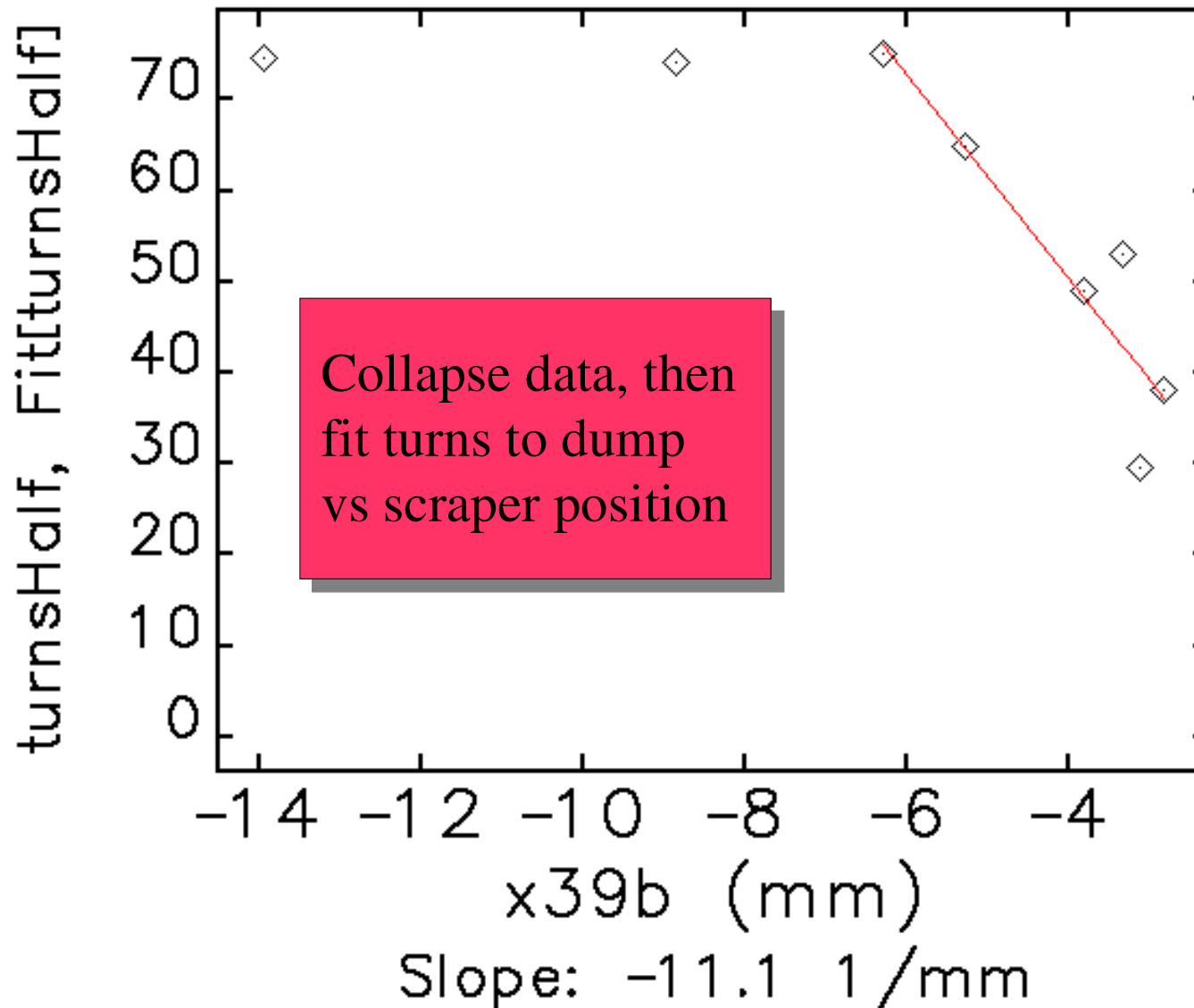


SDDS Processing Example: Beam Dump Analysis



Do for a whole
sequence of
data sets (just
combine files)

SDDS Processing Example: Beam Dump Analysis



SDDS/EPICS Toolkit Capabilities

- Scalar data collection
 - Time-series (`sddslogger`)
 - Time-series statistics (`sddsstatmon`)
 - Glitch- or alarm-triggered with pre- and post-trigger data (`sddsglitchlogger`)
 - Log-on-change (`sddslogonchange`)
 - Conditional logging supported
- These tools are used for most OAG data logging

SDDS/EPICS Toolkit Capabilities

- Synchronized collection (`sddssynchlog`)
 - Does timestamp alignment of high-rate data
 - Optionally collects related, unsynchronized data
 - Supports scalars and waveforms
- Used for on-demand investigation of correlations

SDDS/EPICS Toolkit Capabilities

- Alarm data collection (`sddsalarmlog`)
 - Space-efficient binary files with coded PV names
 - Optional “related PV” logging (e.g., status bits)
- GUI applications for alarm analysis and review, including
 - Alarm rate vs time
 - Alarm counts per PV
 - Alarm overlap
 - Decoding status bits

Alarm Log Review Application

Display File: (/tmp/040317-140813-3685borlan)

```

Linac alarms: minor
2004/3/14@0 - 2004/3/16@24
ControlName          Count
-----
L1:RG2:LFA:StatusCC   3
L1:RG2:QM1:StatusCC   2
L1:RG2:QM4:StatusCC   2
L1:RG2:SC2:HZ:StatusCC 2
L1:RG2:SC3:HZ:StatusCC 2
L1:SC2:HZ:StatusCC   2
L1:SC3:HZ:StatusCC   2
L1:SC3:VL:StatusCC   1
L2:TW:VP:CO1:controllerDG 2
L4:TW:VP:CO6:controllerDG 2
L4:TW:VP:CO7:controllerDG 104
L5:TW:VP:CO4:controllerDG 2
          
```

Indexing... 14:08:13
 Processing done. 14:08:13
 Counting alarms... 14:08:13
 Plotting... 14:08:13

Print | Save As... | Email... | Expand Dialog...

MPL Outboard Driver

File Navigate Options Help

Plot 1 of 2

MAJOR alarms: 2004/3/14@0 - 2004/3/16@24

Number of Alarms

Time starting Sat Mar 13 20:28:55 2004

Linac alarms

System:

- ▼ Booster ◆ Linac ▼ MPS ▼ PAR ▼ SR ▼ SR-BPM ▼ SR-RTFB ▼ SRF ▼ Timing
- ▼ TopUp ▼ ACIS ▼ runControl ▼ SRDCPS-100Hz ▼ CA Beacon Monitor ▼ Custom

Start Year/Month/Day/Hour:Minutes :

End Year/Month/Day/Hour:Minutes :

Directory:

Filename:

Filter:

Data filtering

PV name match:

PV name exclude:

Severity: MAJOR MINOR INVALID NO_ALARM

Histogram control

Bin size:

unit: ◆ seconds ▼ hours ▼ days ▼ weeks

Count control

Threshold (counts):

Sort by: ◆ Name ▼ Counts

Fixed scale: ◆ Yes ▼ No

Printout control

Sort by: ▼ Name ◆ Time

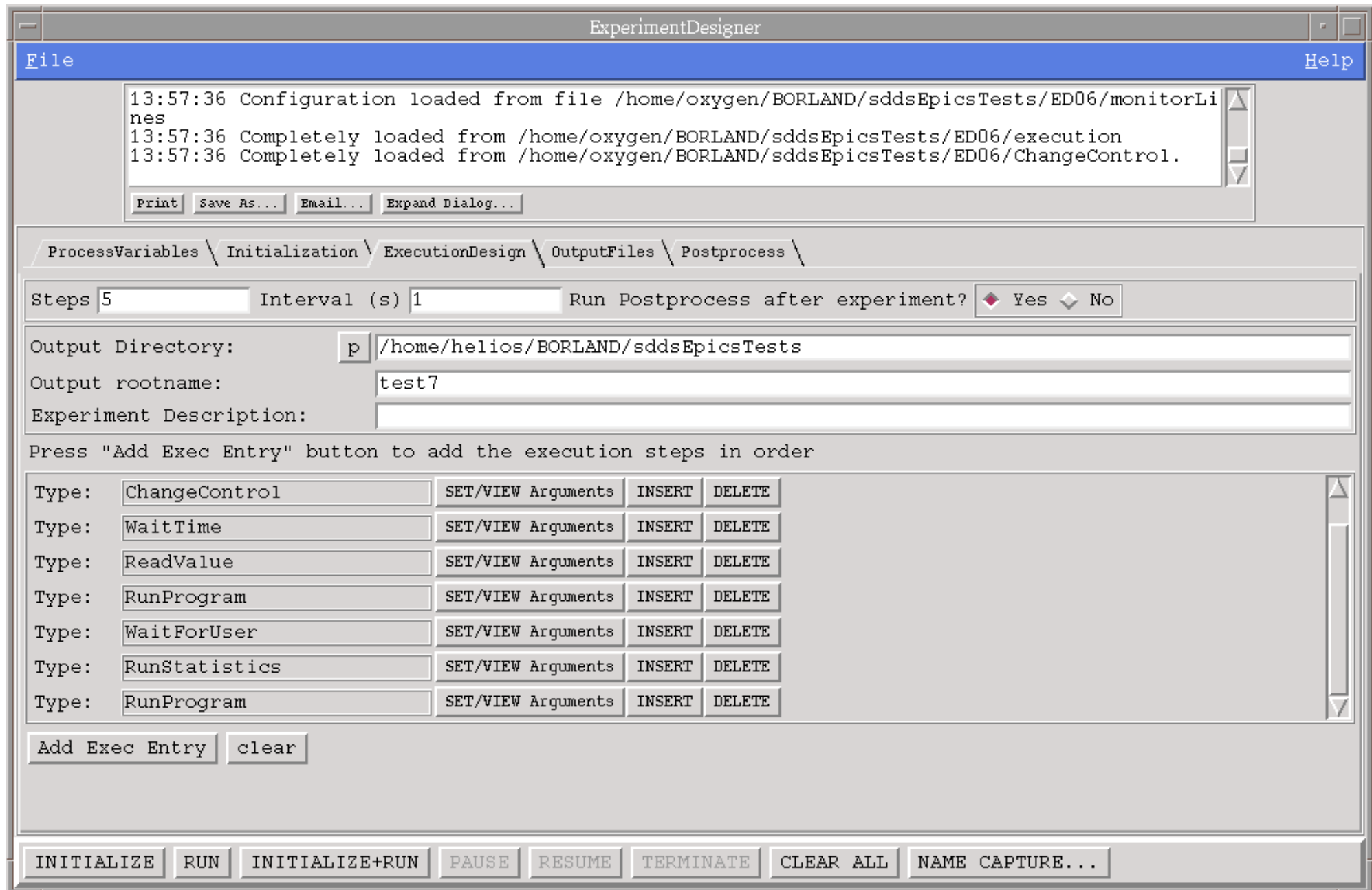
SDDS/EPICS Toolkit Capabilities

- Waveforms collection:
 - At intervals, or when changed (`sddswmonitor`)
 - Simultaneous collection of related scalars
 - Get/put to/from SDDS files (`sddswget`/`sddswput`)
- Used for
 - Beam profile and image capture
 - BPM fast history capture
 - Feedback history capture
 - IOC orbit correction configuration

SDDS/EPICS Toolkit Capabilities

- N-dimensional experiment execution (`sddsexperiment`)
 - Data and statistics collection
 - Validity testing
 - Subprocess execution
- Used for accelerator and hardware characterization
 - Feedback matrix measurement
 - X-ray BPM feedforward data collection
 - BPM offset measurement
- For direct-user interaction, we prefer the new `ExperimentDesigner` script

Experiment Designer



SDDS/EPICS Toolkit Capabilities

- Feedback (`sddscontrolaw`)
 - Proportional or integral mode
 - Validity testing, change limits, deadbands, logging
 - PV controls include locking semaphores
 - Will run under VxWorks
- Applications include
 - Storage ring orbit control
 - Beamline steering
 - Linac energy and trajectory control

Quick One-Variable Feedback GUI

The screenshot displays the 'maintainReadback' GUI with several windows open:

- Feedback Log (Top):** Shows a sequence of operations including reading devices, calling control functions, and setting control devices. It includes a table of statistics:

Stats	Average	rms	mad	Largest
Readback devices	-0.00405	NaN	0	-0.00405 (borland:PM5:X:positionM)
Readback device deltas	-0.0234	NaN	0	-0.0234 (borland:PM5:X:positionM)
Control devices	-1.83	NaN	0	-1.83 (borland:H5:setCurrentC)
Control device deltas	0.00401	NaN	0	0.00401 (borland:H5:setCurrentC)
- Control Parameters (Left):** Includes fields for Change limit (1), Gain (0.99), Action limit (0), Pause (s) (10), Steps (1000000), Samples (10), and Mode (Integral/Proportional). It also has a 'Tests' section with an 'Add' button.
- Beamline Schematic (Middle):** Shows a diagram of the beamline with components H3/V3, H4/V4, and H5/V5. Below the diagram are current and position readings:

Component	I- (mA)	X- (mm)	Y- (mm)
H3/V3	21.86	0.64	-0.00
H4/V4	19.81	0.00	-0.04
H5/V5	17.98	-0.03	-0.07
- maintainReadback: GetGain (Bottom Left):** A dialog box for setting gain parameters:

initial value:	-0.1
final value:	0.1
average:	5
interval (s):	1
post pause (s):	5
steps:	5
gain factor:	0.5
relative to original?	
- Unix Command Execution (Bottom Middle):** Shows the execution of commands to set and ramp the current:


```
Setting borland:H5:setCurrentC to -1.805655e+00
Waiting 5.000000 seconds after changing values
Taking measurements...
* Experiment is 80.00% done.
File updated
Setting borland:H5:setCurrentC to -1.755655e+00
Waiting 5.000000 seconds after changing values
Taking measurements...
* Experiment is 100.00% done.
File updated
Ramping borland:H5:setCurrentC from -1.755655e+00 to
-1.855655e+00
Done ramping.
```
- MPL Outboard Driver (Bottom Right):** Displays a plot of 'Plot 1 of 1' showing a linear relationship between 'borland:H5:setCurrentC (Amps)' on the x-axis and 'borland:PM5:X:positionM (mm)' on the y-axis.

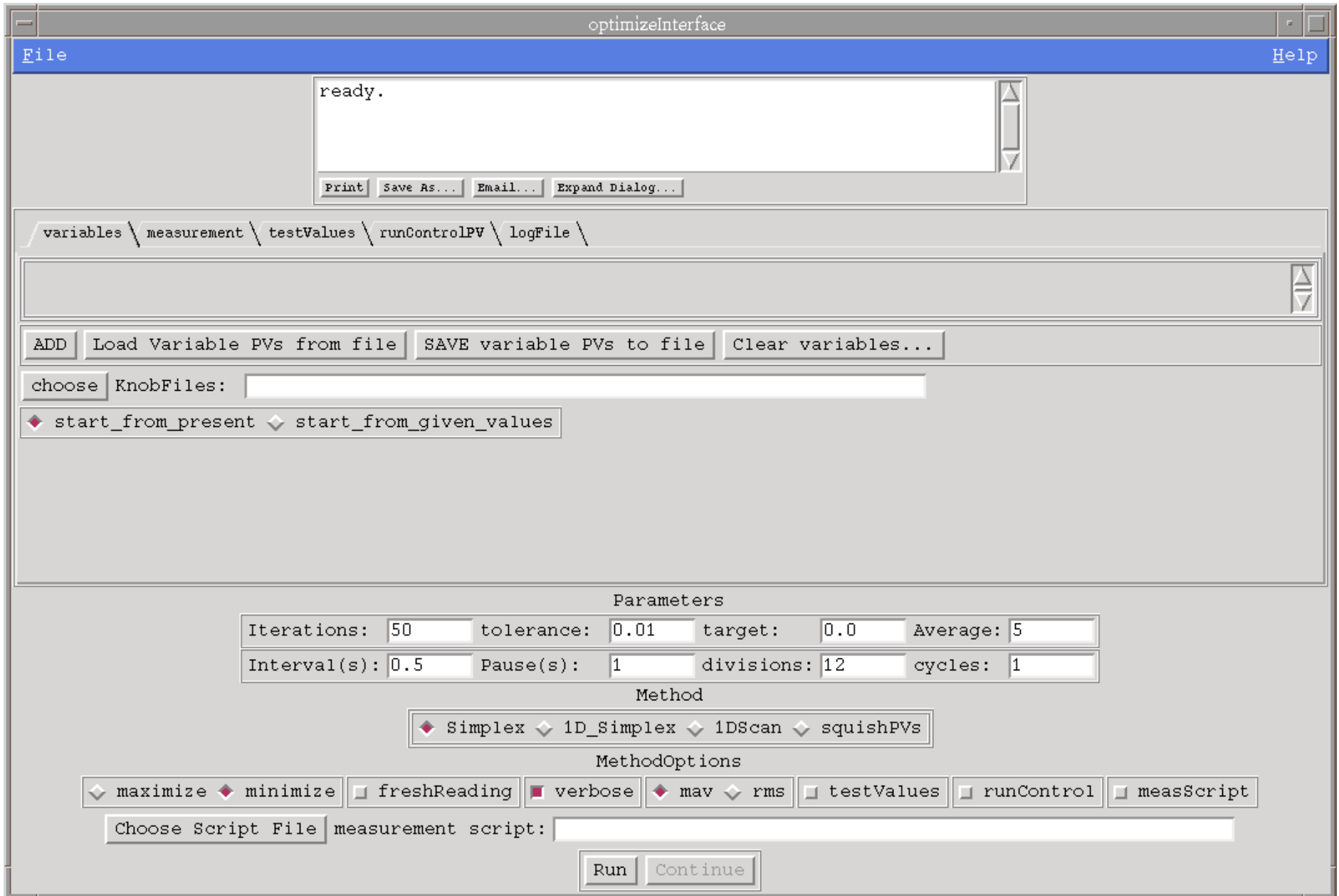
SDDS/EPICS Toolkit Capabilities

- Feedforward (`sddsfeedforward`)
 - Multiple inputs and outputs
 - Locking semaphores
 - Will run under VxWorks
- Applications
 - X-ray BPM gap-dependence compensation
 - Rf BPM intensity-dependence compensation
 - EMW switching correction
 - Septum magnet temperature drift compensation

SDDS/EPICS Toolkit Capabilities

- Generic optimization (`sddsoptimize`)
 - Simplex or successive 1D scan methods
 - Validity testing
 - Script option for setting conditions
 - Script option for computing penalty function
- Applications include
 - Kicker bump matching
 - Coupling optimization
 - Injector efficiency optimization
 - Optimization of simulation results

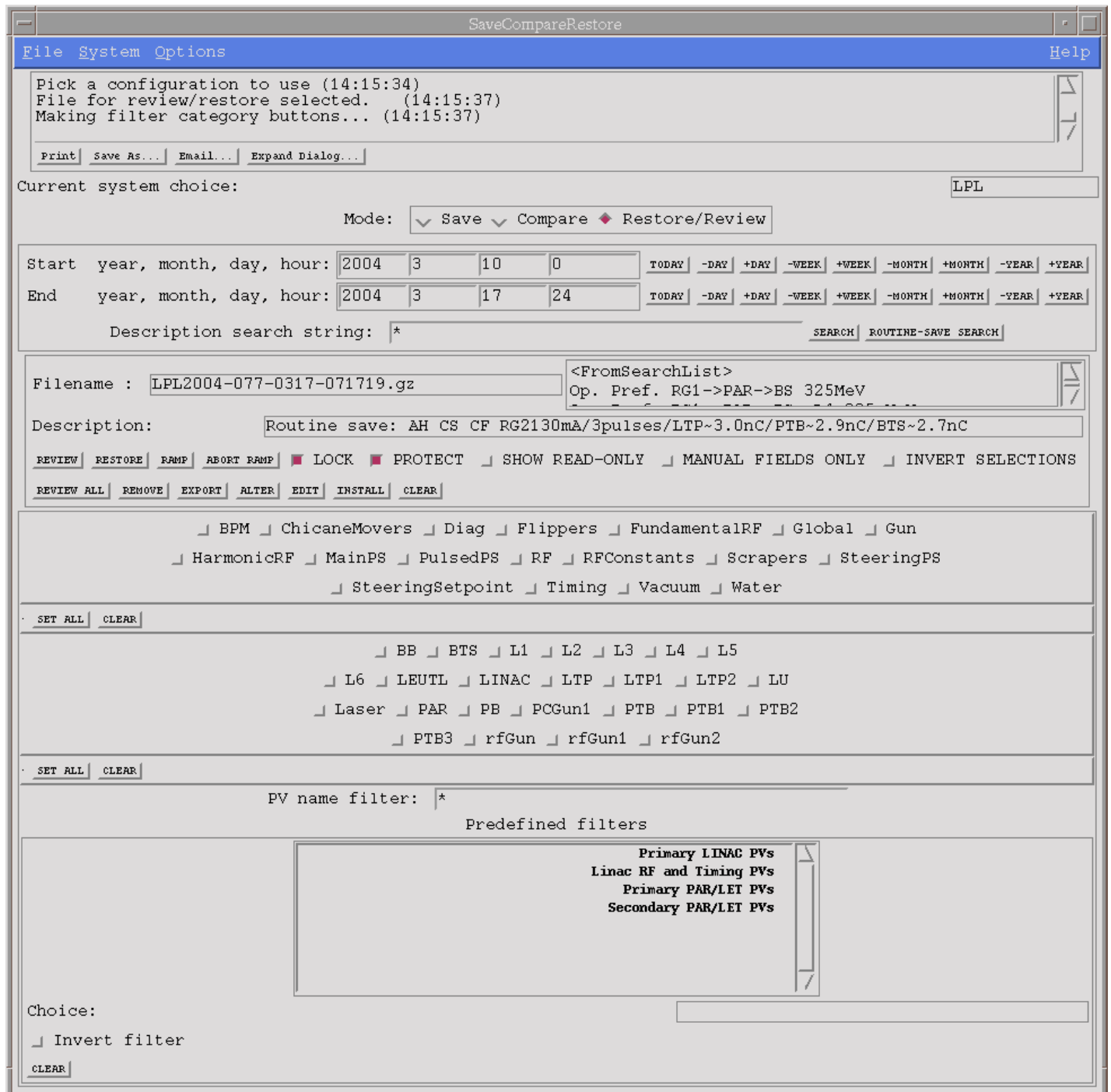
Generic EPICS Optimization Interface



SDDS/EPICS Toolkit Capabilities

- Save/restore
 - Venerable `burt rb/burt wb` pair are (mostly) SDDS-compliant
 - New `sddscasr` program is completely compliant
 - Faster than `burt rb/burt wb`
 - Server mode with PV controls is faster yet
 - Waveform save/restore
 - Program `sddscaramp` ramps through a sequence of snapshots

Save/ Compare/ Restore GUI



SDDS/EPICS Toolkit Capabilities

- PV creation
 - PVs can be created on-the-fly with `sddspcas`
 - SDDS-configured by a file that can also double as
 - Data logger input file
 - Save/restore input file
 - Creates scalar and waveform PVs
 - Checks for existence of PVs before continuing
 - Handy for development work

OAG Data Logging System

- For time-series logging, master (SDDS) configuration file is read by a script on each workstation
 - Identifies the logger, its rate, inputs, triggers, etc
 - Identifies workstations to use for logging, postprocessing
 - Defines how long to keep the data
 - Defines how to postprocess the data.
- Script is run periodically to ensure that all loggers are active
- About 36k PVs are time-series logged
- Intervals from 0.25s to 2 minutes

OAG Data Logging System

- Most loggers use `sddslogger`
 - Multiple input and output file pairs
 - Economize TCP connections
 - Reduce multiple connections to a PV
 - Jobs run forever, creating a new file for each day
 - Supports conditional logging
 - Honors “inhibit PV” that prevents logging and restarts during emergencies (e.g., power outage recovery)
 - Supports use of “data strobe PVs” for quasi-synchronous logging across many loggers and workstations
- Data reviewed through a Tcl/Tk interface or Web



OAG Data Logging: Unique Features

- OAG data logging tools offer more than simple time-series or monitor-based logging
 - Conditional logging
 - Glitch logging
 - Statistics logging
 - Synchronized logging of fast data
 - Alarm logging with related data
- Loggers are SDDS-configured
 - Generate configurations with scripts
 - Process configurations like any other SDDS data
- Custom post-processing and display is easy to develop with SDDS tools

Conclusion

- OAG has a powerful set of EPICS tools based on
 - Tcl/Tk scripting language
 - SDDS file protocol and program toolkits
- This software is used to
 - Automate APS accelerator operations
 - Automate accelerator physics measurements
 - Perform data logging, analysis, and display
 - Perform feedback, feedforward, and optimization
 - Pre- and post-process simulation data
- Software is generic and configurable to diverse applications