XAL Applications, Correlator and Framework

Thomas Pelaia II

SNS Accelerator Physics R&D Staff

June 18-20, 2003

Active Developers

- Chris Allen (LANL)
- Chungming P. Chu (ORNL)
- John Galambos (ORNL)
- Wolf-Dieter Klotz (ESRF)
- Craig McChesney (LANL)
- Dan Ottavio (BNL)
- Thomas Pelaia (ORNL)
- Andrei Shishlo (ORNL)

Acknowledgements

- Nikolay Malitsky (BNL)
- Peregrine McGehee (LANL)
- Nick Pattengale (formerly at LANL)
- Cosylab Group
- EPICS Community

Contents

Part I

XAL Applications

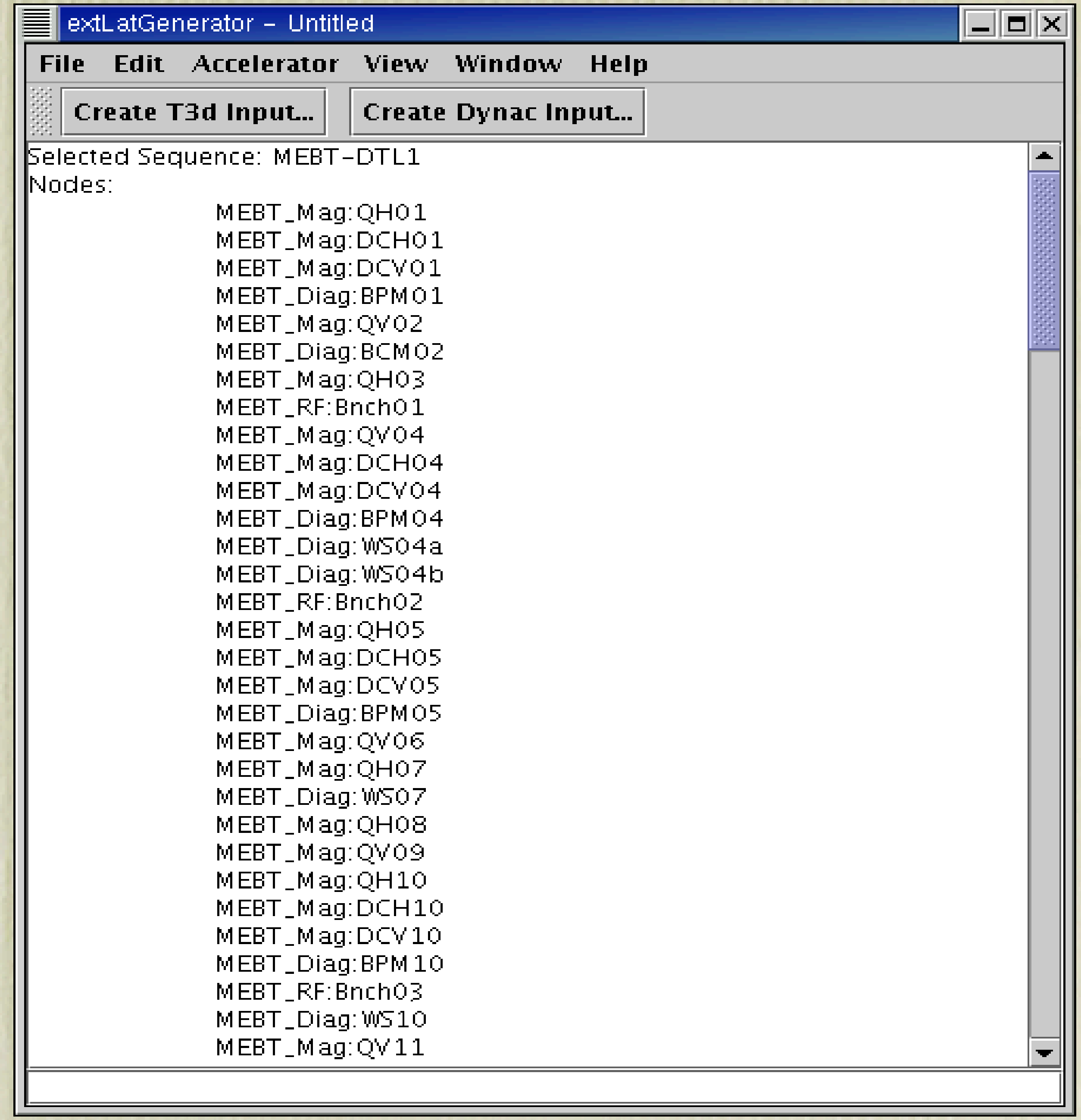
Applications

Application	Author(s)	Description	Application Framework
ExtLatGenerator	C.P. Chu, W.D. Klotz	Generate external lattice files: Trace 3D, Dynac	✔
MEBT Cavity Scan	A. Shishlo	Controls amplitude and phase of MEBT cavities	
MPS Post Mortem	J. Galambos	Monitor and sort (by time stamp) machine protection trips	✔
MPX Main	W.D. Klotz	Runs the model and gets parameters at elements	✔
One-D Scan	A. Shishlo	Monitors one PV against another as it is scanned	
Orbit Correction	T. Pelaia	Monitor and correct the orbit as needed in the background	
Orbit Difference	C.P. Chu	Verifies magnet wiring against the model	
Scope	T. Pelaia	Displays waveforms on a common time base	✔
xio	N. Pattengale, C.P. Chu, J. Galambos, D. Ottavio	Monitor PVs in table, line plot and waterfall plot	✔
xyz correlator	C.P. Chu, J. Galambos	Plots correlated, live PV scalars relative to each other	✔

External Lattice Generator

C.P. Chu, W.D. Klotz

- Exports the XAL lattice to an external format
- Supported Formats:
  - ✓ Trace 3D
  - ✓ Dynac

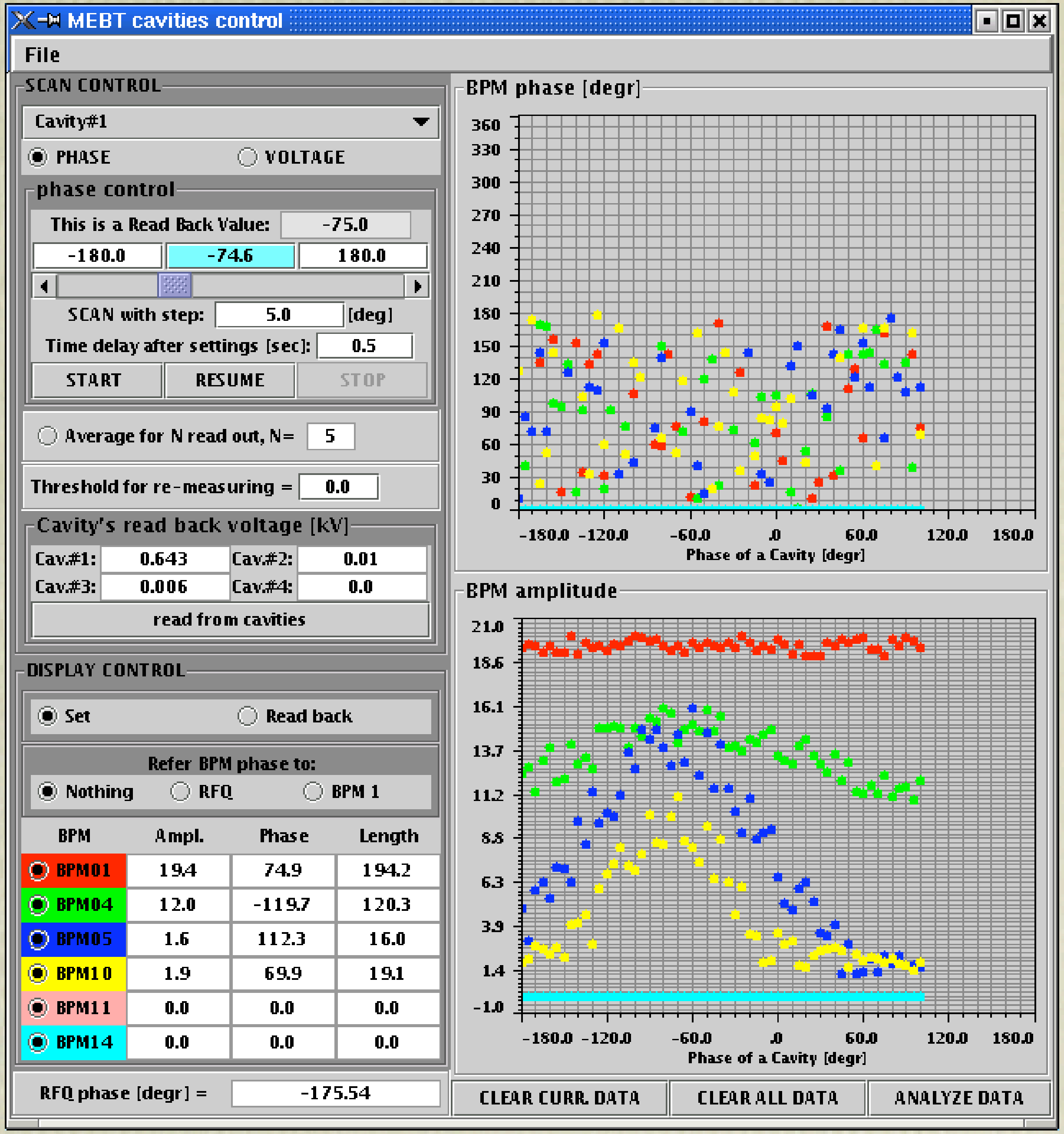


MEBT Cavity Scan

A. Shishlo

- Scan amplitude and phase of MEBT rebuncher cavities to find optimal settings based on BPM response

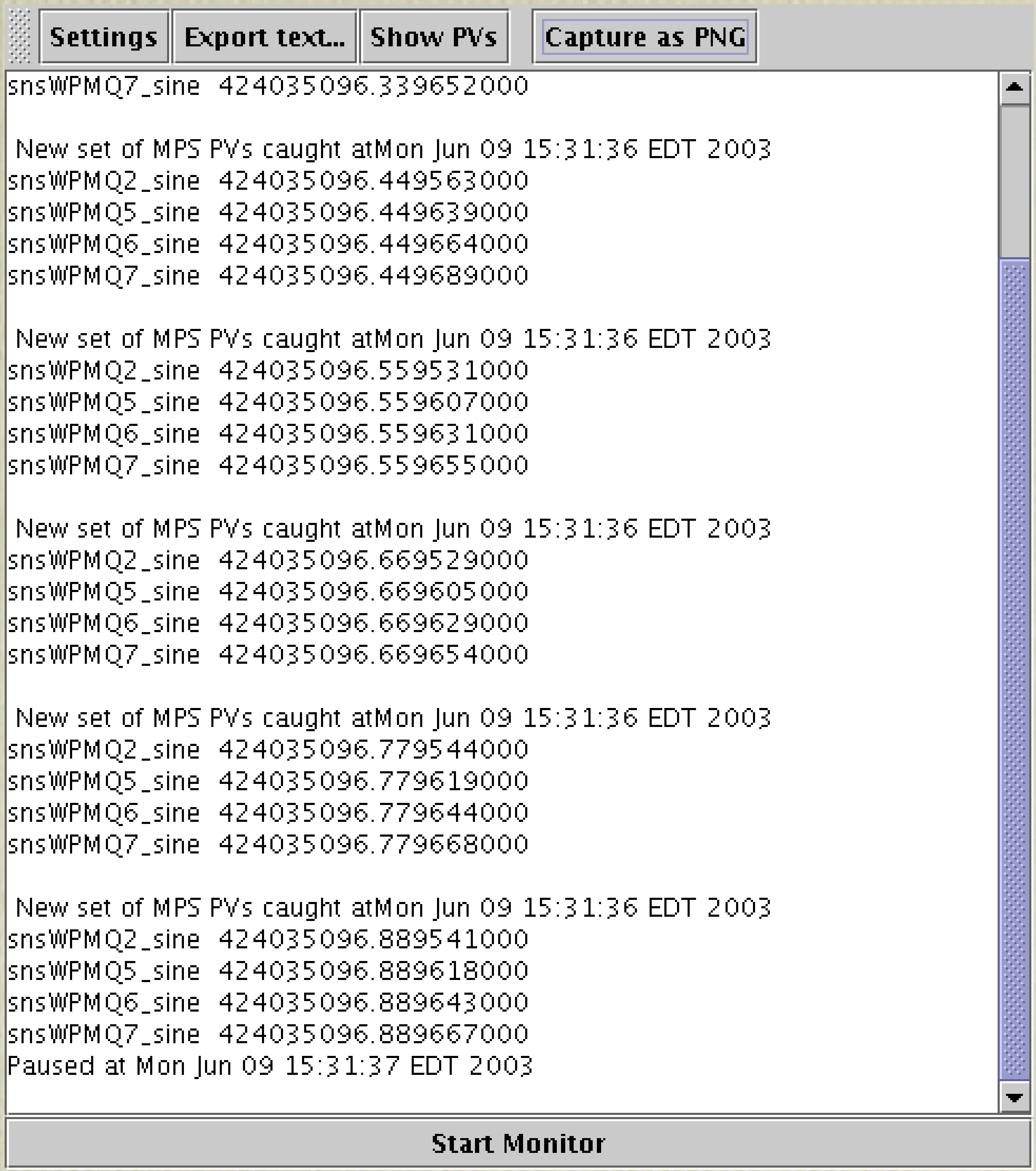
- Ported from a Matlab version to XAL



MPS Post Mortem

J. Galambos

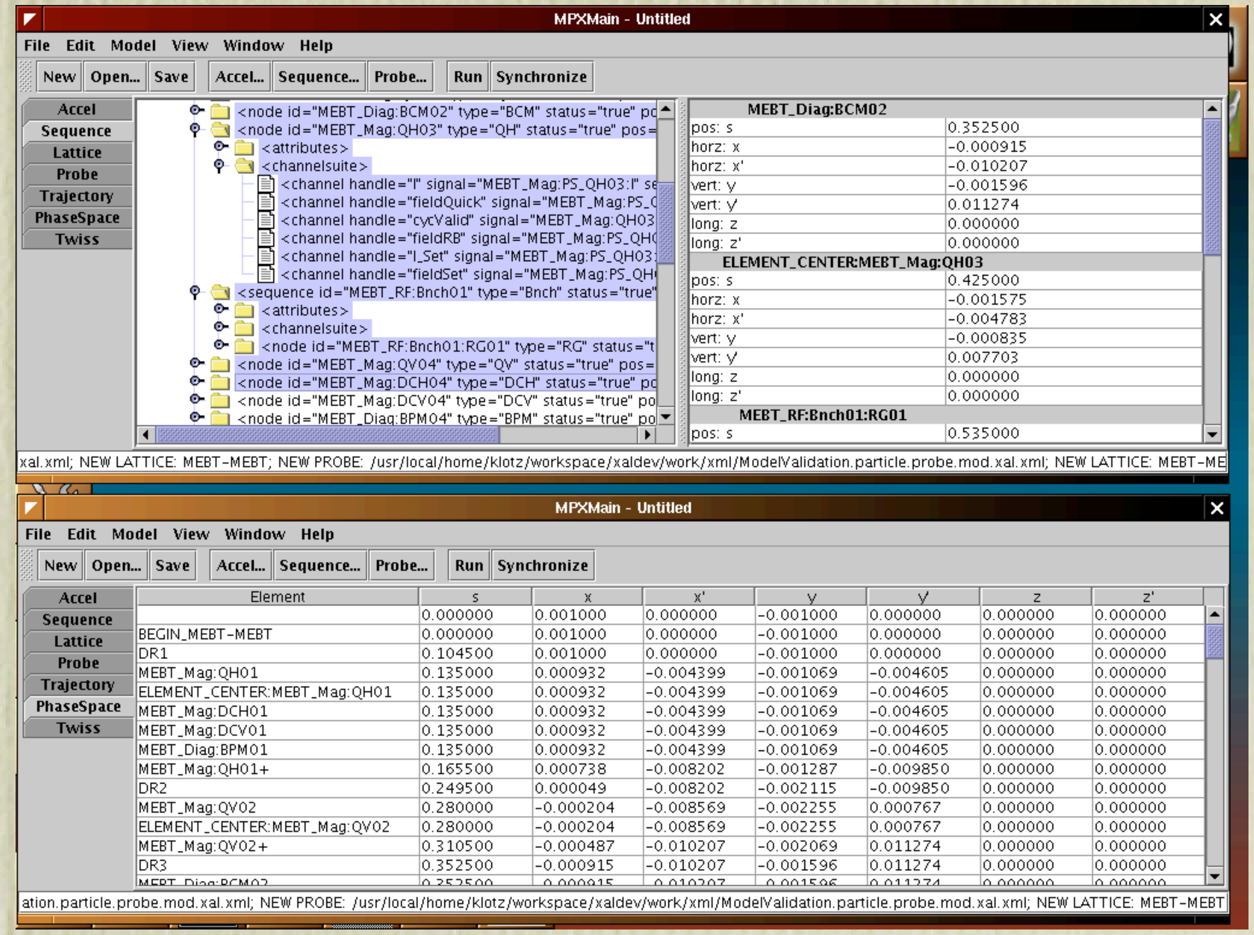
- Monitors for machine protection trips
- Groups events by macro pulse
- Sorts events within a macro pulse by time stamp



MPX Main

W.D. Klotz

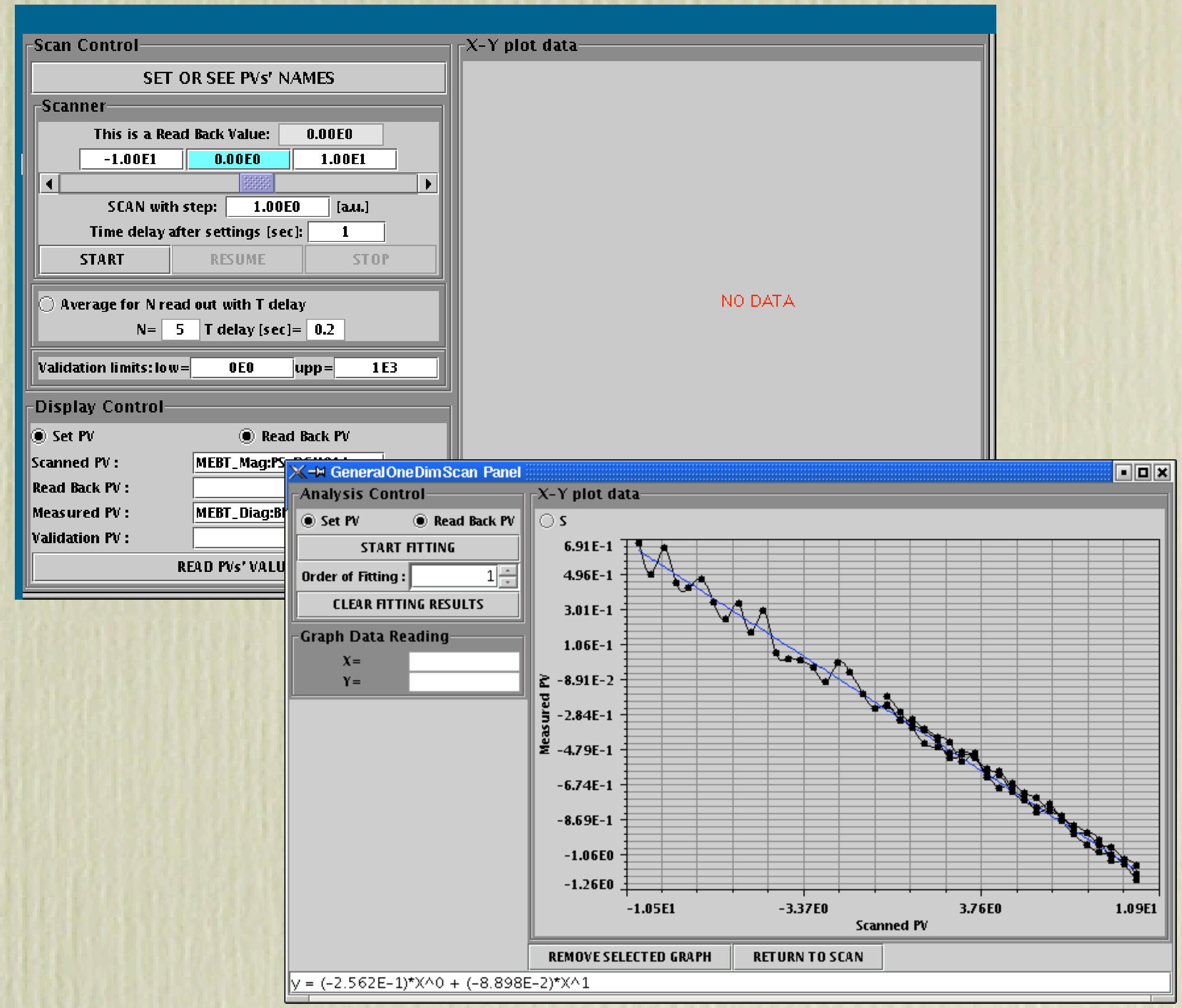
- User interface to the online model
- Runs the model through a selected section of the accelerator
- Displays twiss parameters and phase space information at each element
- Synchronizes model with real machine



One-D Scan

A. Shishlo

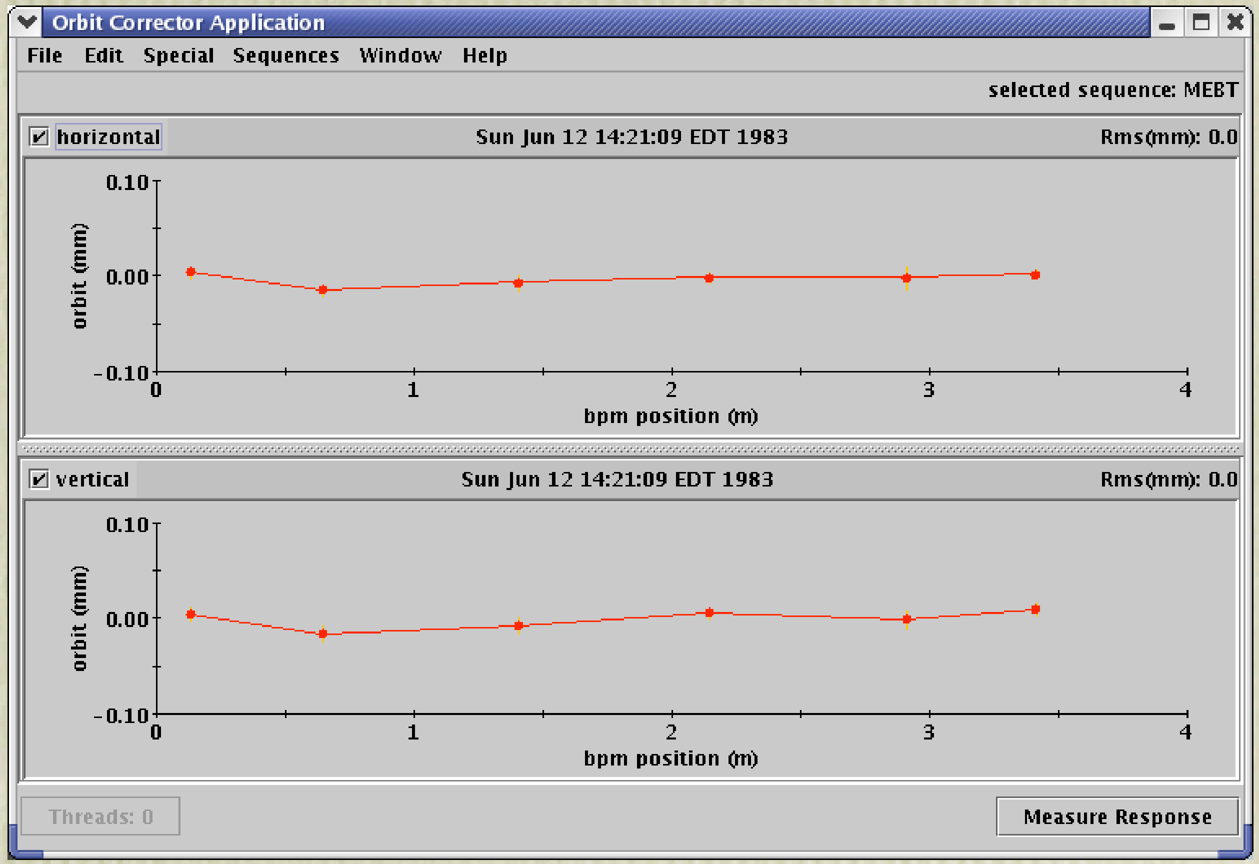
- Scans a PV and monitors another PV during the scan
- User specifies details of the scan
- Provides for analysis of scanned data



Orbit Correction

T. Pelaia

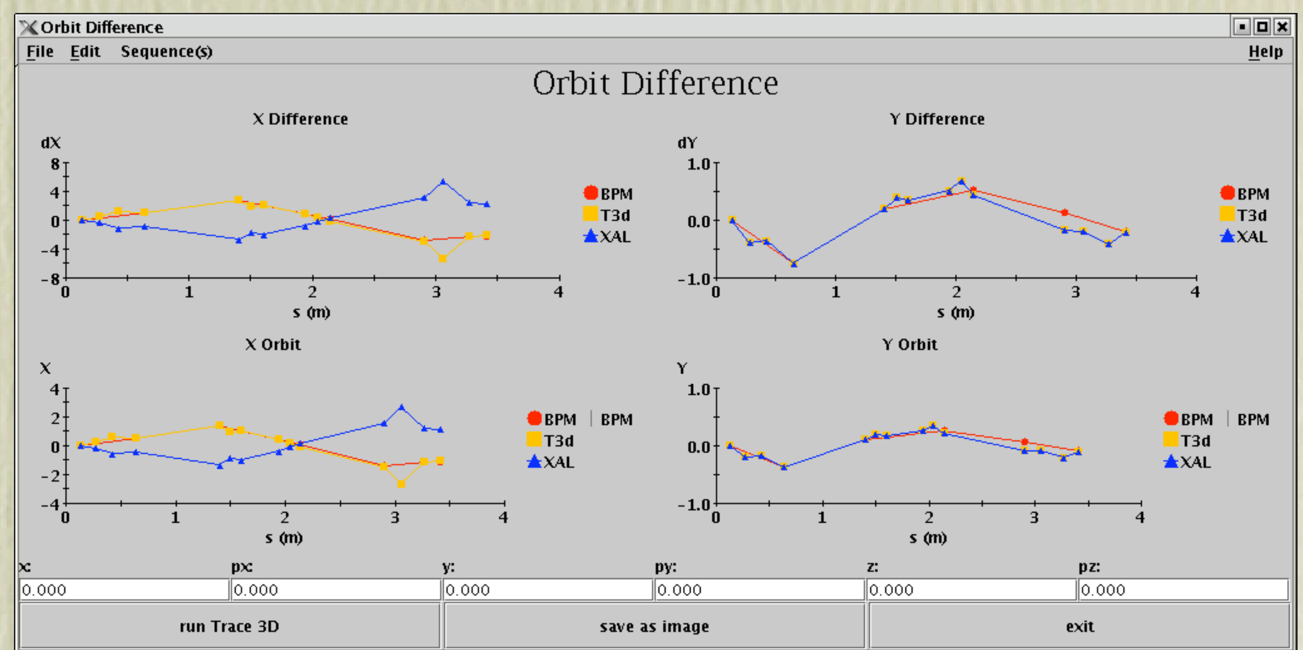
- Monitors and displays the live orbit
- Corrects the orbit in the background as necessary
- Measures the BPM-Corrector response matrix
- Optimization specified via satisfaction curves



Orbit Difference

C.P. Chu

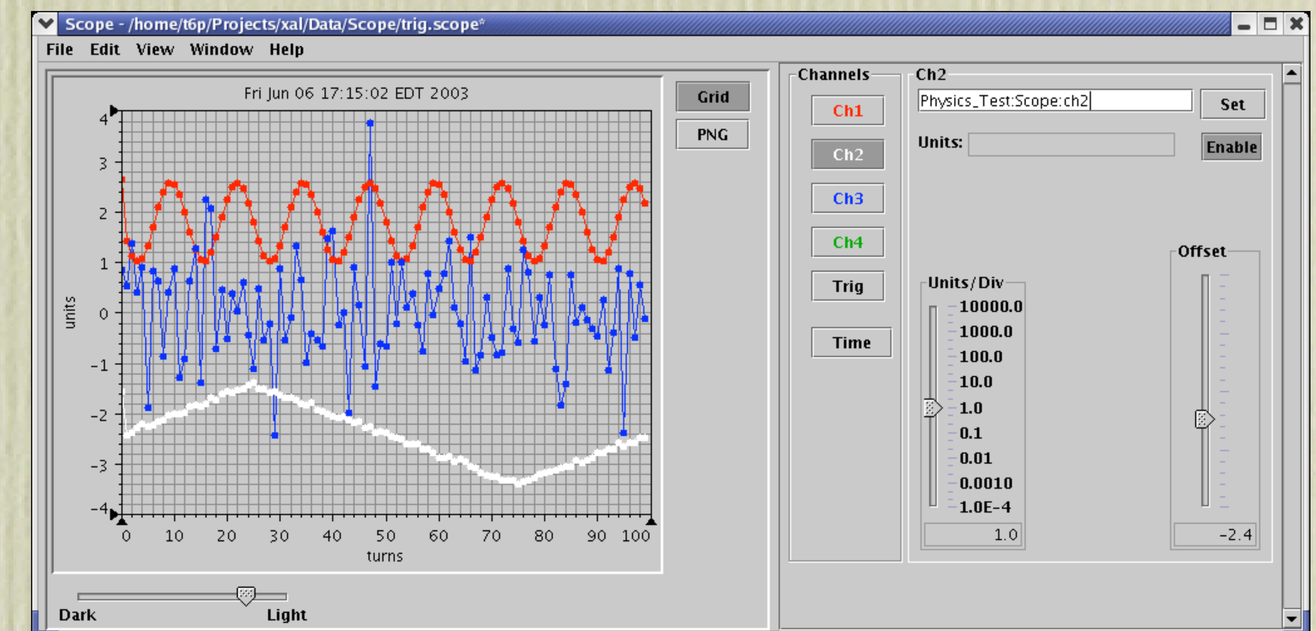
- Tool for verifying the machine against the online and Trace 3D models
- Useful to confirm magnet wiring
- Varies magnet strengths and displays the resulting change in orbit



Scope

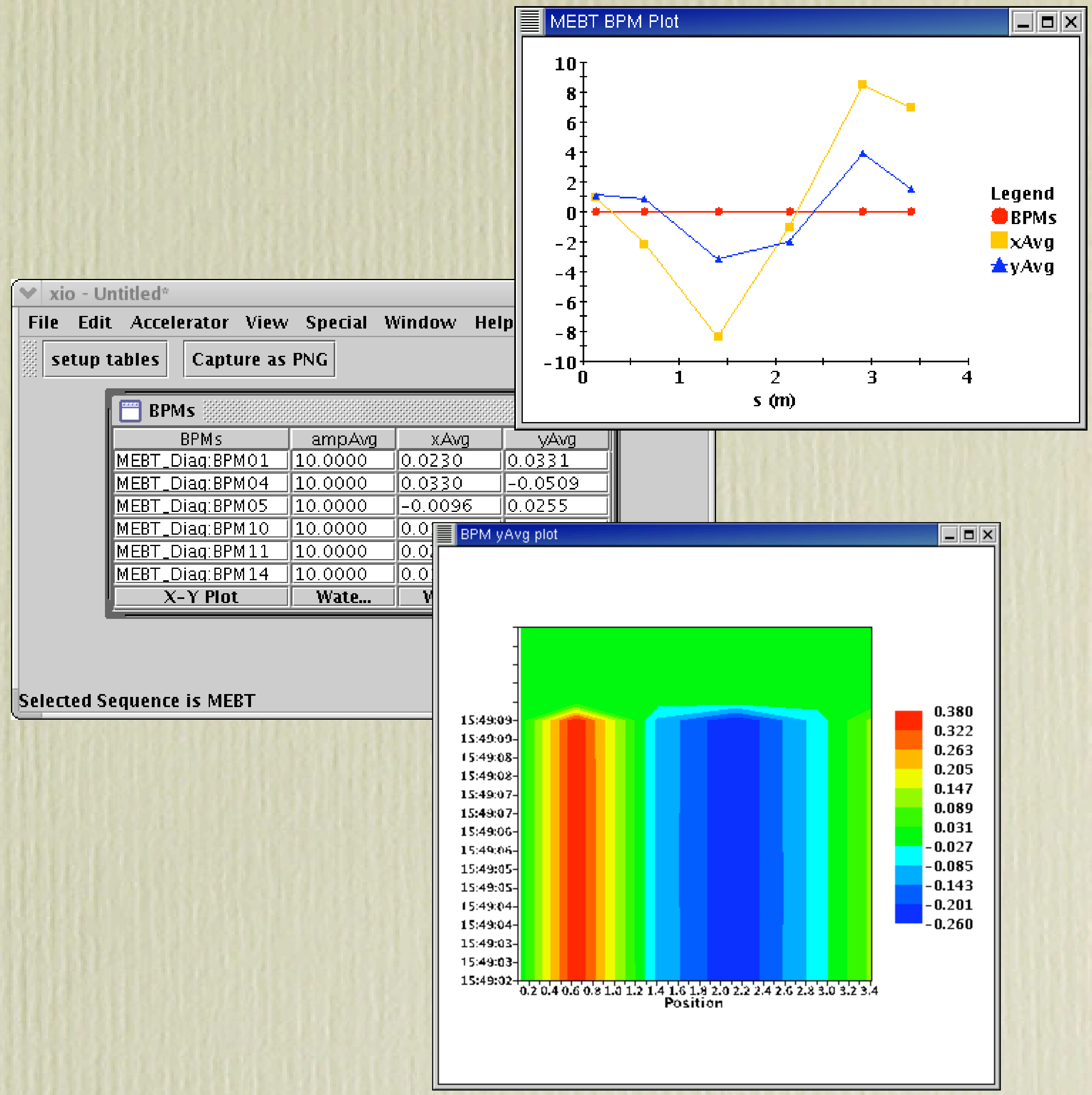
T. Pelaia

- Virtual Scope
- Monitors and displays concurrent waveforms
- Waveforms aligned against a common time reference
- Each channel requires a waveform PV, time delay PV and sample period PV



N. Pattengale, C.P. Chu
J. Galambos, D. Ottavio

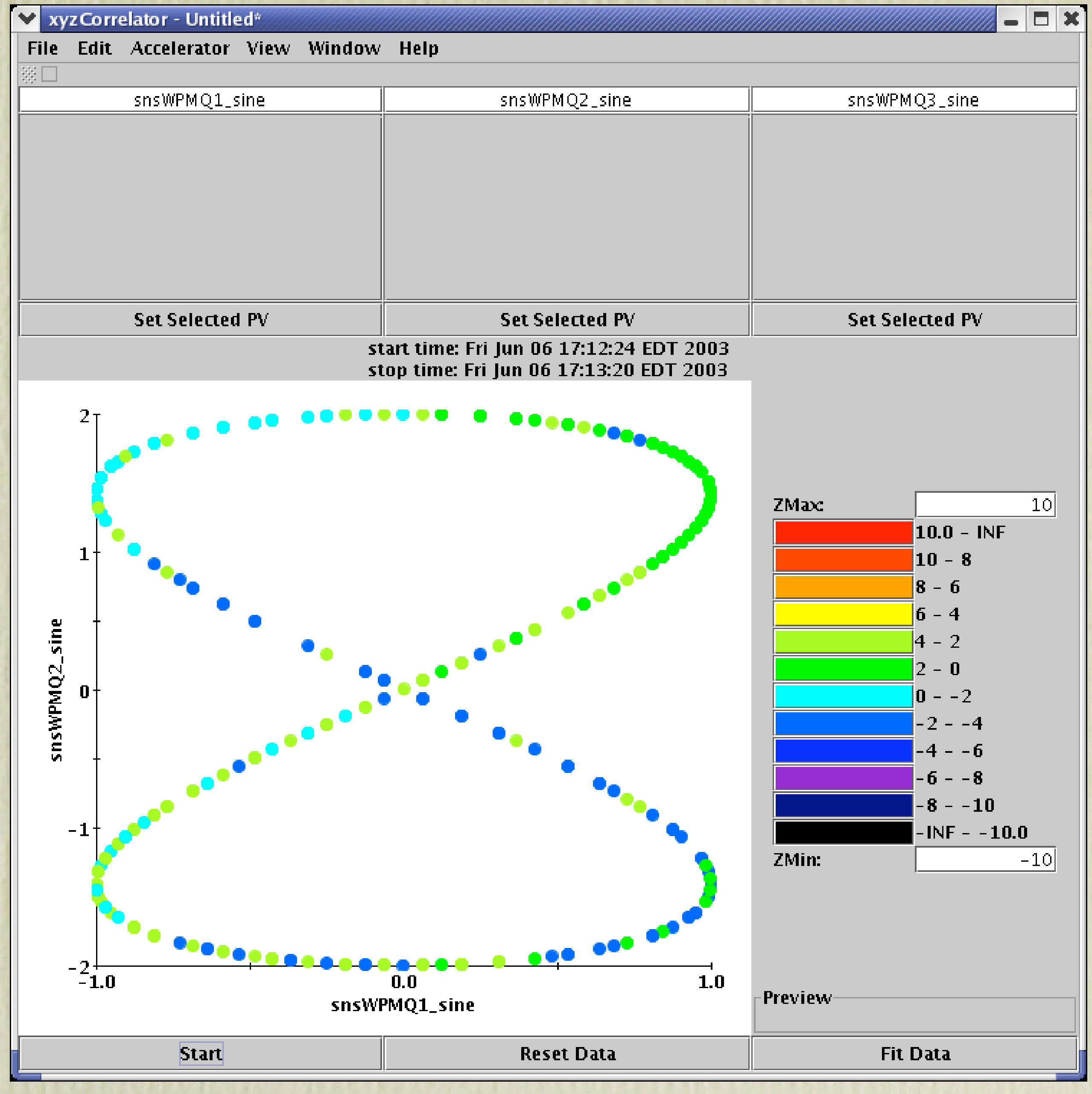
- Live monitoring of PVs grouped by type
- Tabular and graphical views
- Convenient drill down PV selection tool organized by accelerator sequence and element type



XYZ Correlator

C.P. Chu, J. Galambos

- Plots correlated PV scalars relative to each other
- Can correlate two or three PVs
- Provides data fitting
- Allows exporting of data



Part II

XAL Correlator

Correlation Defined

Two or more events are correlated if they occur within a specified time window relative to each other. More specifically, at SNS, we usually pick a time window narrow enough to guarantee that two or more events are from the same macro pulse.

Other Correlators

- EPICS Correlator
  - Developed by Peregrine McGehee
  - Written in C++
  - Specific to Epics
  - Set the standard for the XAL correlator

Motivation for an XAL Correlator

- Want a pure Java correlator for XAL
- Want an extensible, component based package
- Need for handling nested correlations (e.g. orbit capture)
- Desire to have support for efficient, custom event filtering
- Want support for a variety of correlation configurations

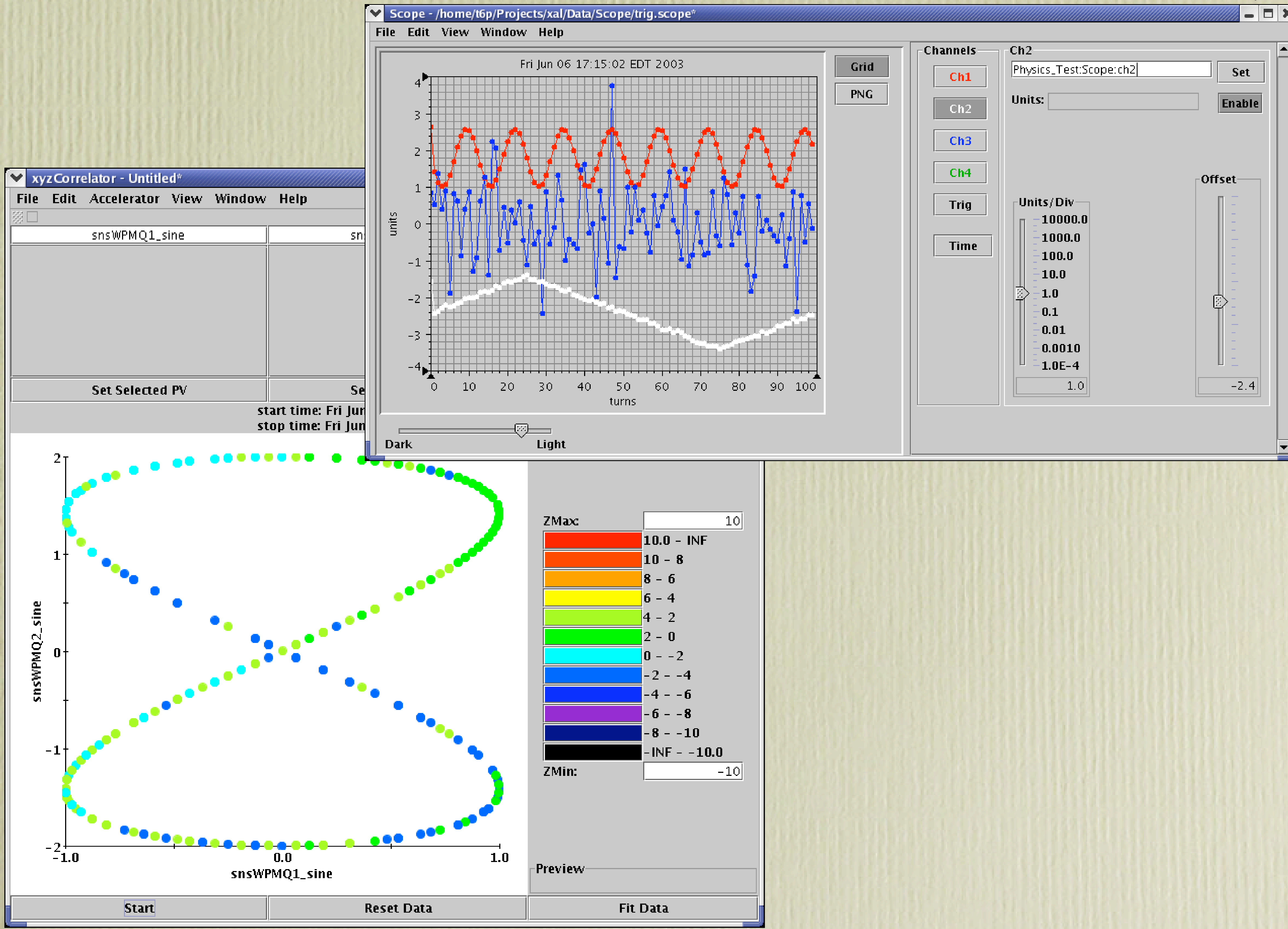
What is the "XAL Correlator"?

- XAL package for correlating events
- Written in pure Java
- Two fundamental packages
  - Abstract correlator foundation to correlate events from sources
  - Channel specific implementation to correlate XAL channel monitor events (EPICS PV monitor events)

Sample Applications

Application	Sources	Filter
Orbit Correlator	BPM Correlators	BPM count
BPM Correlator	BPM xAvg, yAvg, ampAvg	Amplitude threshold, count
Scope	Generic array channels	count, trigger
MPS Post Mortem	MPS Signals	state, count
XYZ PV Correlator	generic channels	none

Sample Application Screenshots



Design guidelines

- Create an extensible correlation framework in which the Channel correlator is a specific kind

- Allow for custom filtering of the source events and the correlation set

- Provide a filter factory for common filters

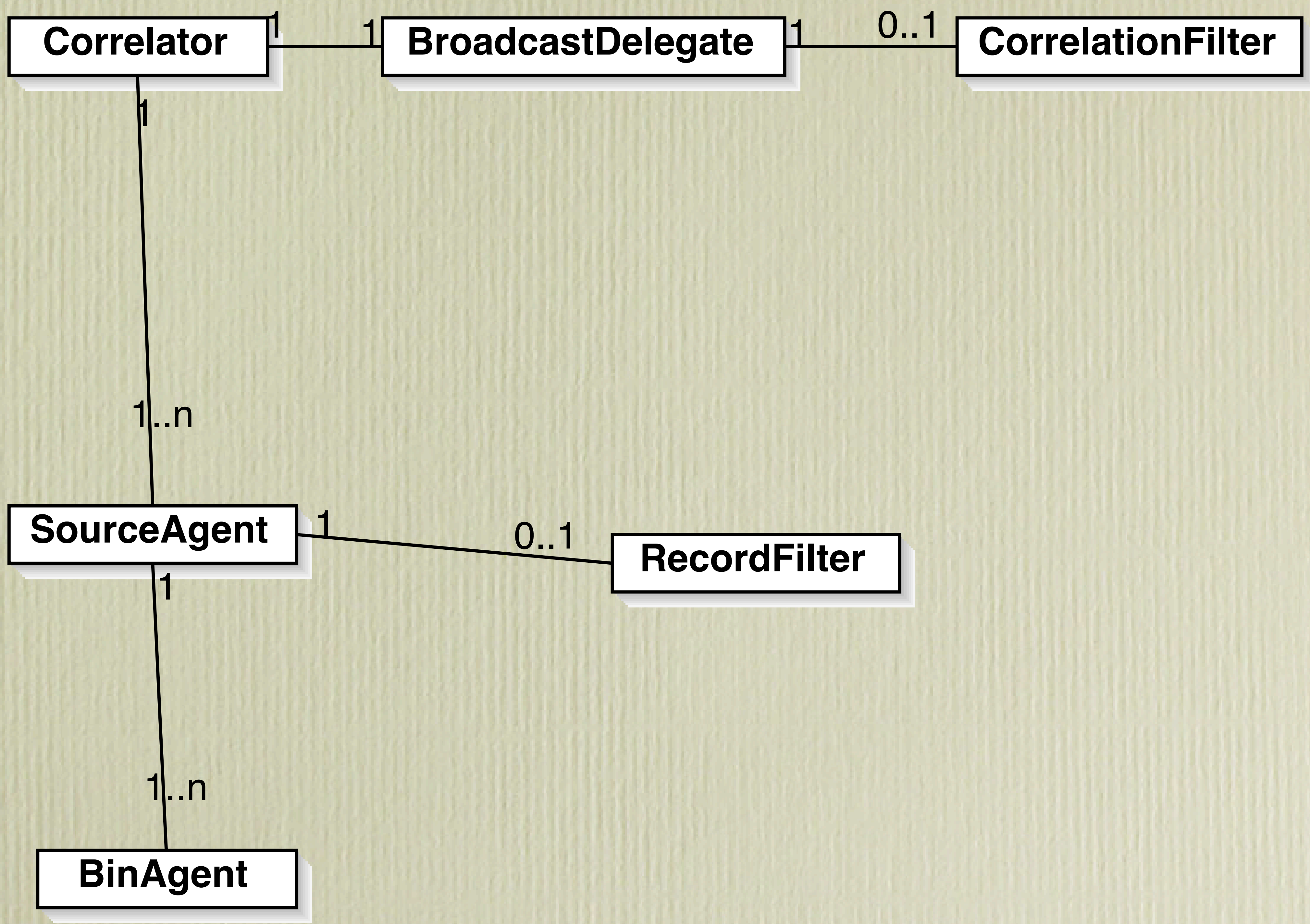
- Make the correlation process efficient and deterministic

- Offer several correlation run modes to support a variety of applications (single shot, recent best periodic, live flood)

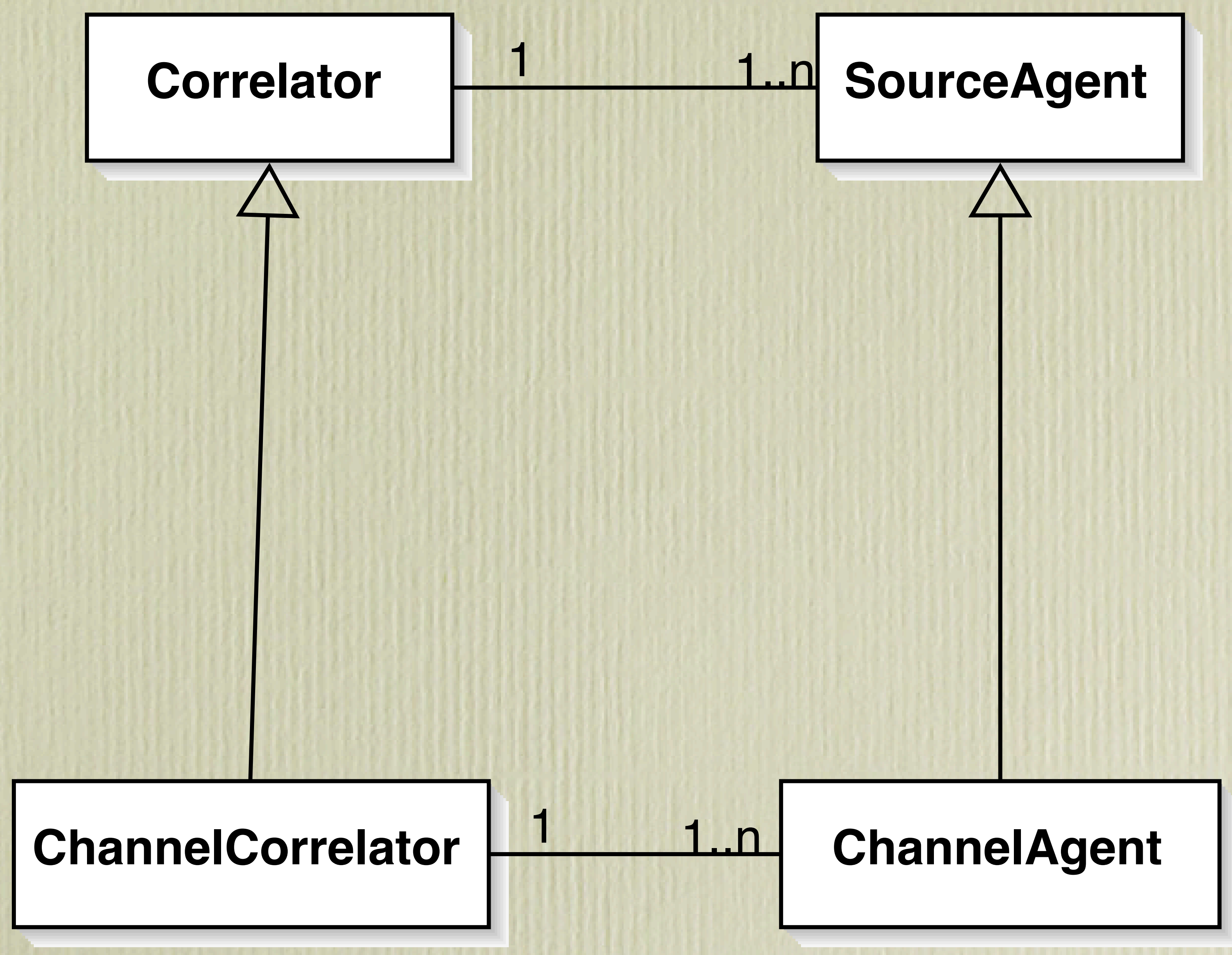
Core Classes

Class	Purpose
Correlator	Public class for specifying the correlation parameters
BroadcastDelegate	Handles broadcasting of correlations to the client on behalf of the Correlator
SourceAgent	Manages the connection to and monitoring of a single source (e.g. PV)
BinAgent	Collects new events that correlate with a single event
RecordFilter	Interface of custom filters to be applied to new events
CorrelationFilter	Interface of custom filters to be applied to new correlations
Correlation	Container of the correlated records
CorrelationNotice	Interface of correlation listeners

Correlator Class Hierarchy



Channel Correlator Inheritance



Correlation Process

1. A source event is posted and filtered by the source agent (e.g. PV monitor event)
2. The source agent recycles the oldest bin agent and sets its time stamp to the event's
3. Existing bin agents check for correlations with the new event
4. New bin correlations are posted internally to the broadcast delegate.
5. The broadcast delegate posts filtered correlations to the registered clients

BinAgent Details

- Each source agent maintains a circular buffer of bin agents
- A bin agent listens for new events from all sources
- Each bin agent maintains the earliest and latest time stamps among the accepted events
- It is okay for a new event to have an earlier time stamp
- Time is measured in seconds since the EPICS epoch as a double value

Nested Correlators

- Our design allows correlators to be nested
- A source agent simply wraps a correlator
- Allows for simpler and more transparent filtering
- Real example is the Orbit Correlator
  - BPM correlates its position and amplitude signals and filters on amplitude
  - Each BPM is a source for an orbit

Correlation Modes

- Post all correlations immediately (works only if the correlation is complete)
- Single shot correlation
- Periodic lossy posting (ideal for GUI applications)
- Periodic buffered posting (to be implemented)

Part III

XAL Framework

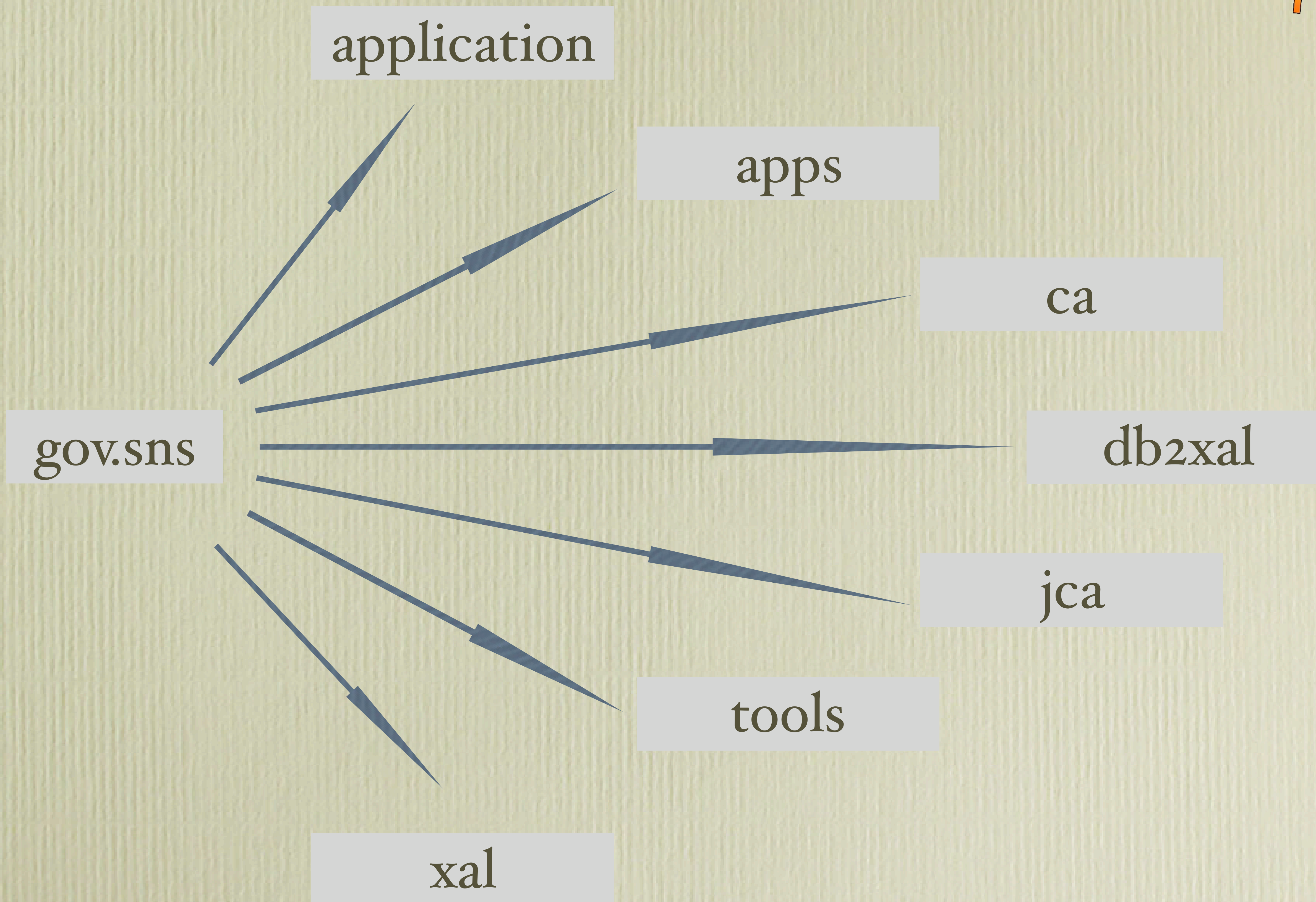
What is XAL?

- XAL is a Java framework for rapidly developing accelerator based applications.
- Founded upon UAL but is now independent
- Provides common tools for the entire accelerator
- Primarily built to accommodate SNS but is quite general

Goals

- Allow rapid development of robust applications
- Develop components once and share them across applications
- Have a common look and feel for all applications
- Develop intuitive, rich human interfaces
- Maintain good performance
- Support online modeling

Top Level Packages



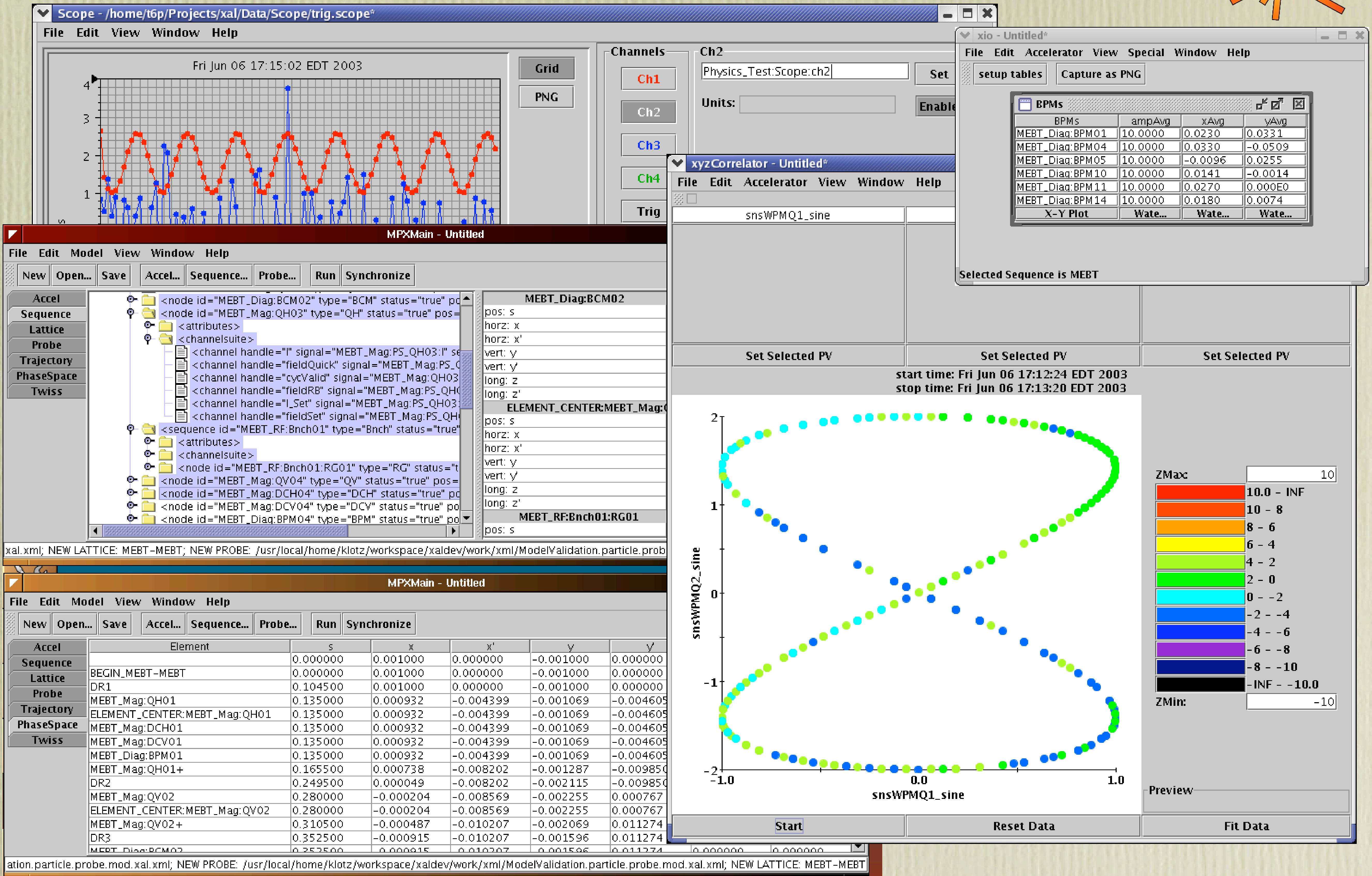
Application Framework

- Provides a common look and feel to all of our applications
- Generic framework plus one accelerator theme
- Conforms to familiar user interface guidelines
- Document based
- Facilitates rapid application development
- Minimal boundaries on the developer
- Easily extensible

Application Classes to Subclass

- ApplicationAdaptor
  - Hooks to handle application events
  - Advertises application specific attributes
- XalDocument
  - Handle document events and behaviors
  - Save and restore a setup
- XalWindow
  - Main window for a document

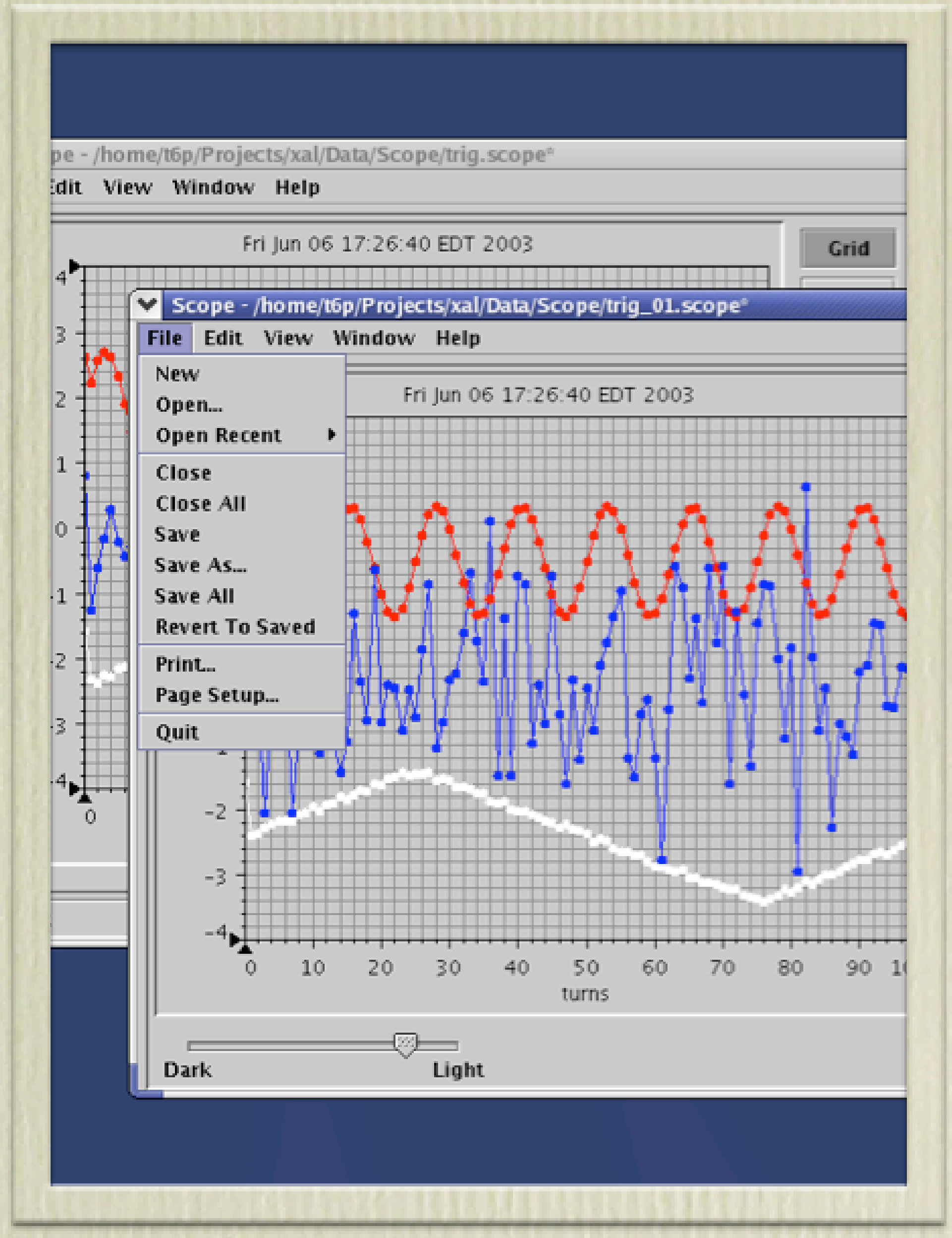
Some Application Screenshots



Familiar Look and Feel for Free

- Familiar Menus and commands
- Multi-document paradigm
- Standard output and error console
- Conforms to human interface guidelines
- Inherit several features with zero overhead

Scope Application



Tools

- Facilitate rapid development
- Some provide a common look and feel
- A few examples among dozens of current tools:
  - GUI components and utilities
  - data management
  - charting
  - math
  - correlator

Channel Access

- Two packages: gov.sns.ca and gov.sns.jca
- gov.sns.ca provides a convenient, abstract layer for generic channel access
    - Our applications and the XAL framework use this package alone for channel access
- gov.sns.jca is an adaptor that bridges gov.sns.ca with a slightly modified version of jca

Accelerator Input

- Oracle database holds the accelerator definition which consists of accelerator sequences, devices and PVs
- The accelerator definition is copied into an XML file
- XAL reads the XML file and generates an object graph view of the accelerator
  - Accommodates overriding and extension
- Online model generates a lattice view of the accelerator with sequential elements

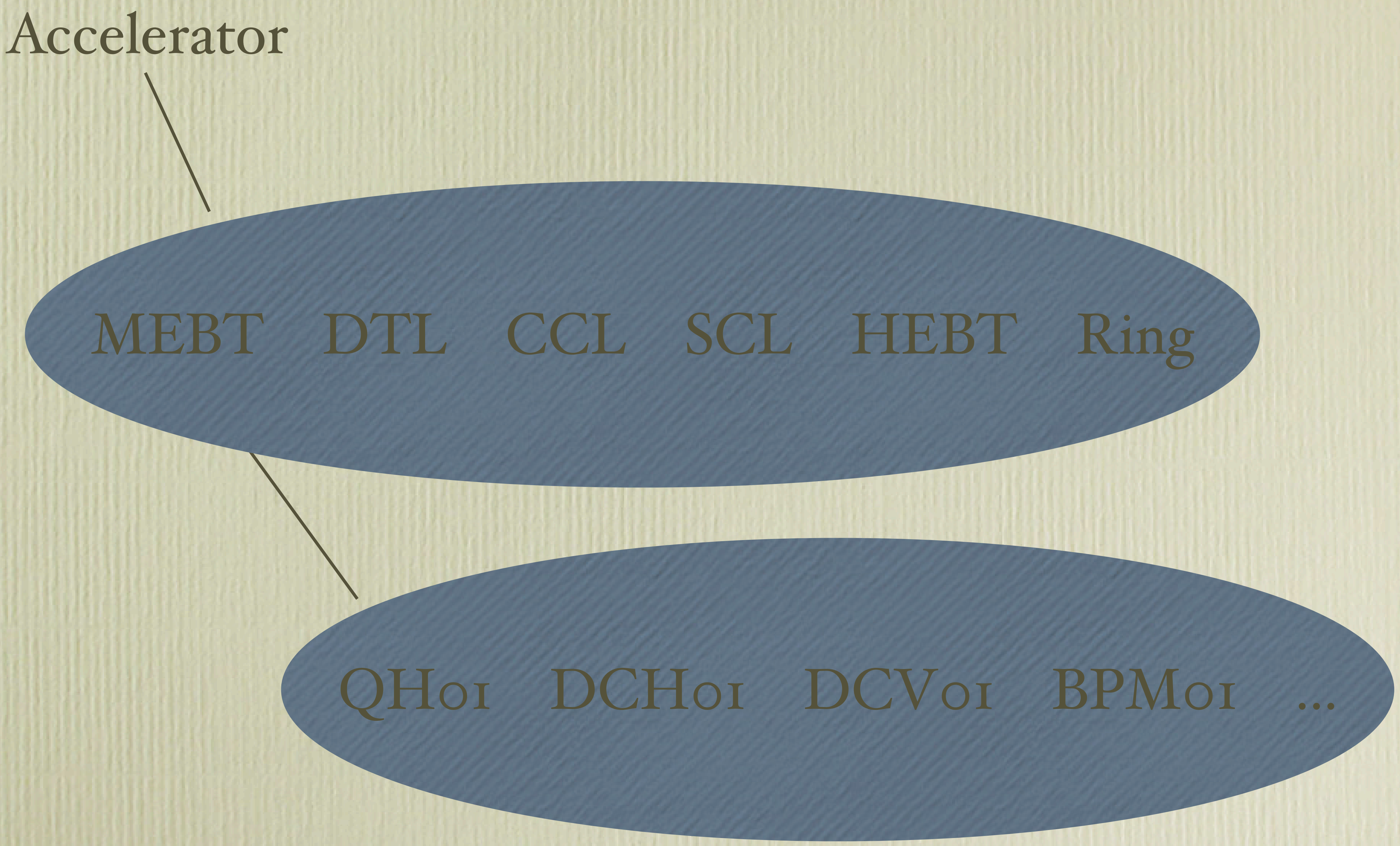
Device View vs. Lattice View

	Device View	Lattice View
Overlap	Allowed	No
Physical Mapping	One to One	One to many
Drifts	No	Yes

SMF Object Graph

- Accelerator contains accelerator sequences
- Accelerator sequence
  - corresponds to a physical section of the machine
  - contains accelerator nodes and sequences
- Accelerator node
  - One accelerator node per physical device
  - No drifts

Sample Accelerator Hierarchy



Common Accelerator Nodes

- One class per device type
- Magnets
  - Bends, Correctors, Quadrupoles
- Diagnostics
  - BPM, BCM
- RF devices
- Generic
  - Completely data driven (no specific class)

Online Model

- Based on UAL Element/Algorithm/Probe architecture
- Calculates twiss parameters and transfer matrices
- Data synchronization for live analysis
- Lattice view is generated from SMF view but has drifts and element slices which are appropriate for model calculations

Present Activities

- Move to JCA 2.0 when it is released
- Collaborate with other JCA stakeholders
- Complete the online model development
- Begin design and development of an agent based architecture
- Write several applications

Conclusion

- XAL has provided a rapid development environment for developing accelerator physics applications in Java
- XAL has proven to be flexible enough to meet new challenges and demands
- XAL applications were successfully used during MEBT commissioning and user feedback has been positive
- For more information and resources please visit: **http://www.sns.gov/APGroup/appProg/xal/xal.htm**