

# An Overview of XAL

XAL - A High-Level Control Application Framework

Christopher K. Allen  
Los Alamos National Laboratory



KEK - February 18, 2005

# XAL Developers and Collaborators

- ORNL
  - Tom Pelaia
  - Paul Chu
  - Andre Shishlo
  - John Galambos
  - Sarah Cousineau
- ESRF
  - Wolf-Dieter Klotz
- LANL
  - Craig McChesney
  - Bob Dalesio
  - Chris Allen
- Cosylab
  - Igor Kriznar
  - Ales Pucelj
  - Mark Plesko

Special thanks to John Galambos for providing much of this presentation!

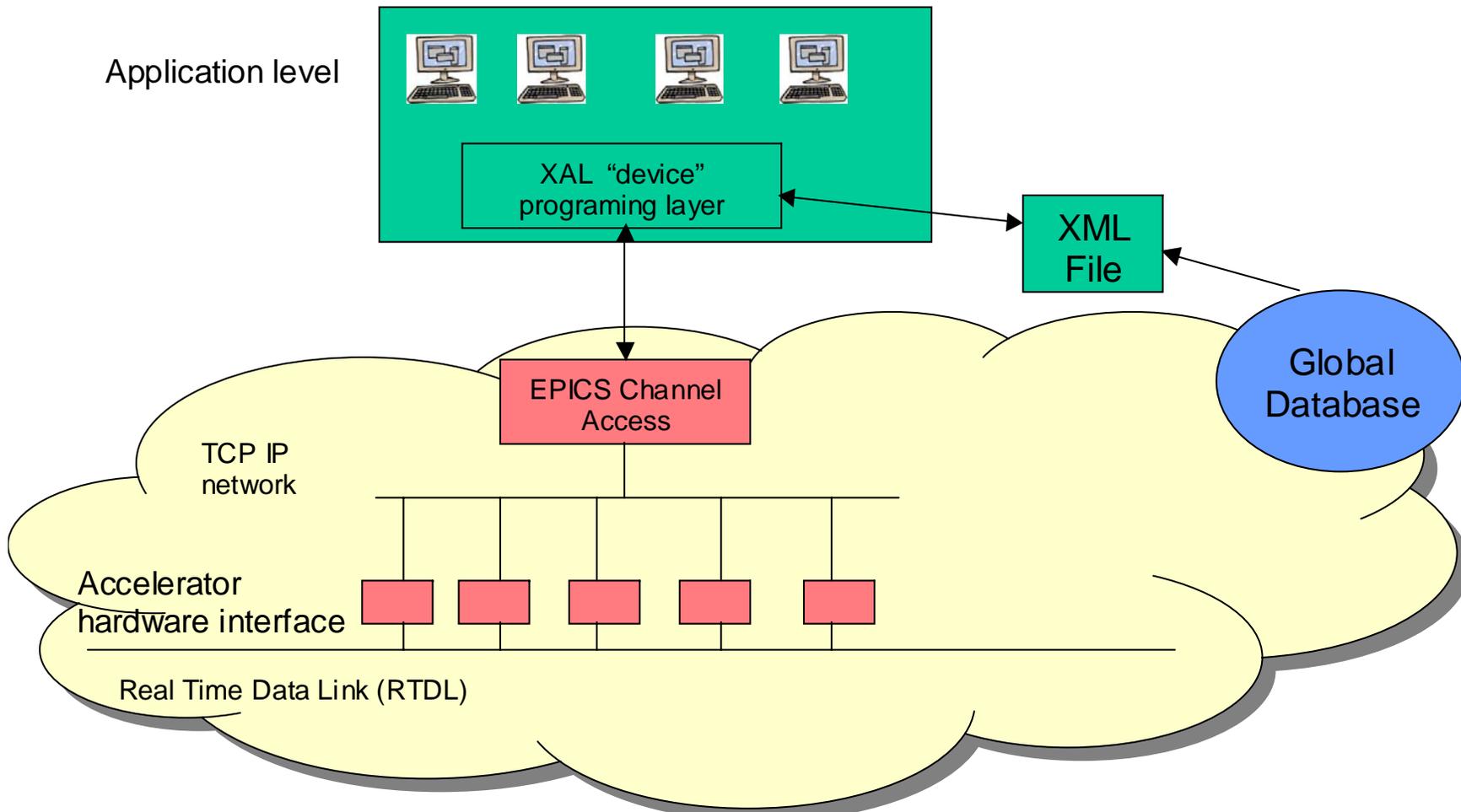
# Outline

see <https://www.sns.gov/APGroup/appProg/xal/xal.htm>

1. XAL Overview
2. Architecture
3. Simulation Subsystem
4. Application Framework
  - Sample Applications
5. Summary

# 1. Overview - Conceptual Diagram

## 1. Overview

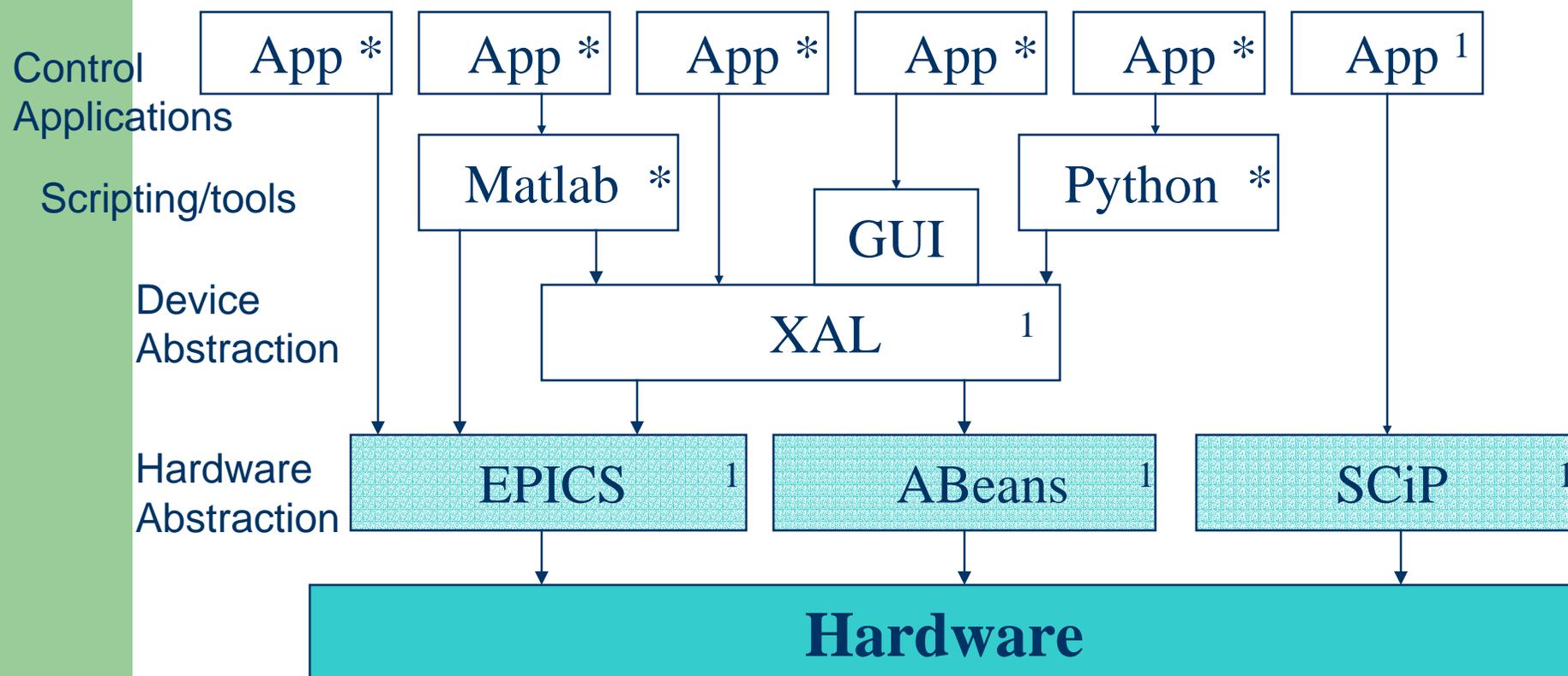


- XAL is a software infrastructure providing a hierarchical, device-oriented view for application programmers
- Java based with scripting

# XAL “Lingo”

- **Framework or Infrastructure** (gov.sns.\*)
  - General code sharable among applications (in xal.jar)
- **Application Framework**
  - See gov.sns.application and gov.sns.xal.application
  - A “template” for building gui applications
- **Accelerator hierarchy, or SMF** (gov.sns.xal.smf )
  - Class structure for representing accelerator hardware
- **Online model** (gov.sns.xal.model)
  - A physics model for doing quick linear transport modeling

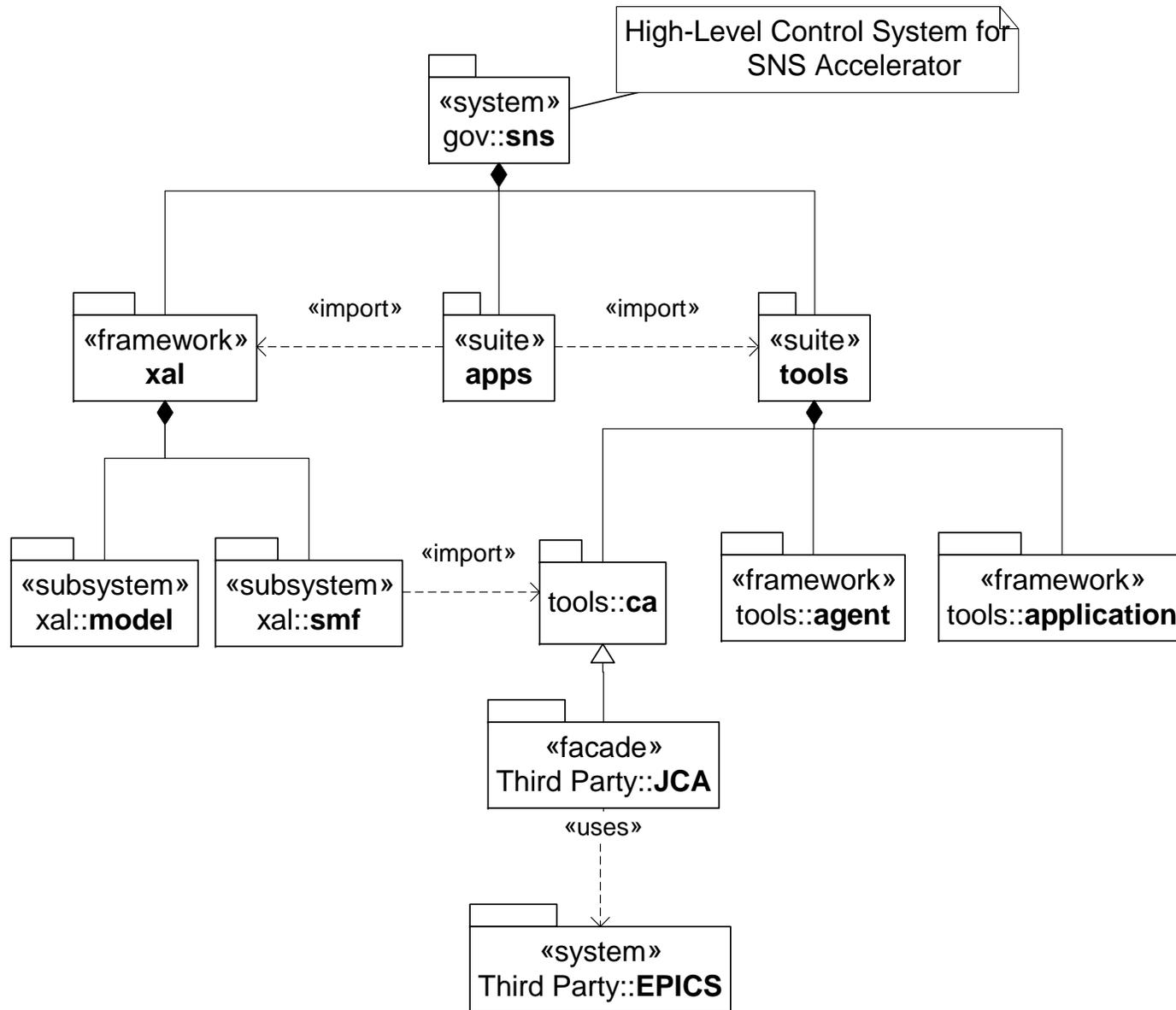
# XAL in the Control System Hierarchy



# 2. XAL Architecture

## UML Subsystem Diagram

# 2. XAL Architecture



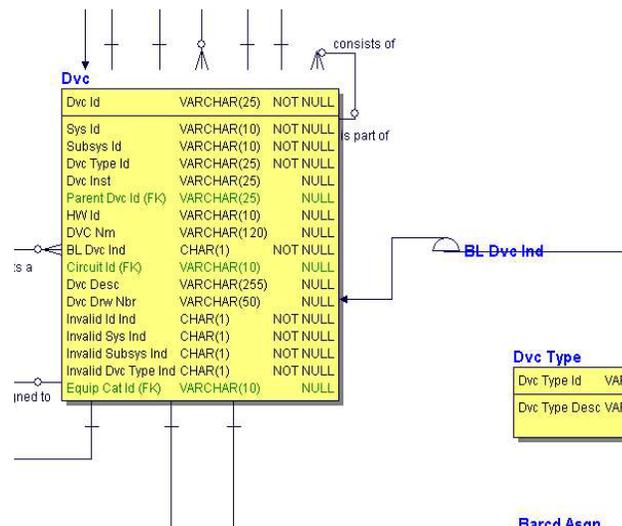
# Global Database Generates XML “Configuration” File

- Serves as a common configuration file for all applications
- Creates the accelerator hierarchy
  - Provides the “map” from the flat list of EPICS signals to the accelerator hierarchy

Database



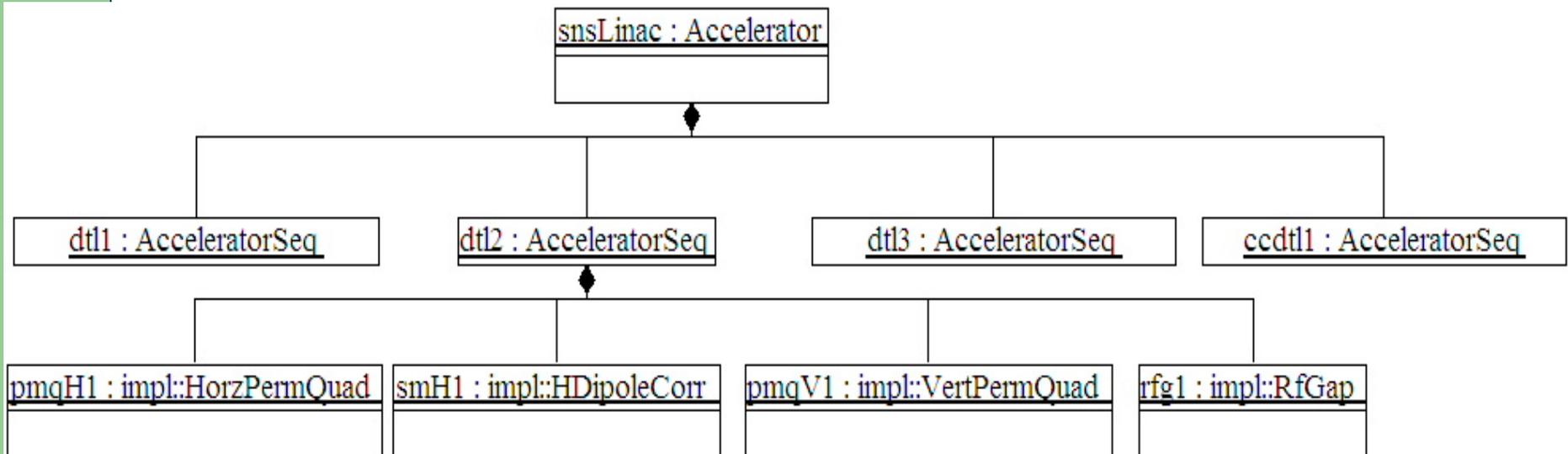
XML configuration file



```
<node type="QH" id="MEBT_Mag:QH01" pos="0.135" len="0.061">
  <attributes>
    <align x="10000" y="2000.046" z="19546.440844" pitch="0" yaw="0"
      roll="0" type="bucket" />
  </attributes>
  <channelsuite name="magnetsuite">
    <channel handle="fieldSet" signal="MEBT_Mag:QH01:fieldSet"
      settable="true" />
  </channelsuite>
</node>
```

# XAL Datagraph

- XAL “Datagraph”
  - Class structure provides hierarchical “device view” of the accelerator
  - Sequences (MEBT, DTL, CCL, ...)
  - Individual Node type classes (Quads, RF, BPMs, etc.)
- Sequences can be “pasted together” as ComboSequences .



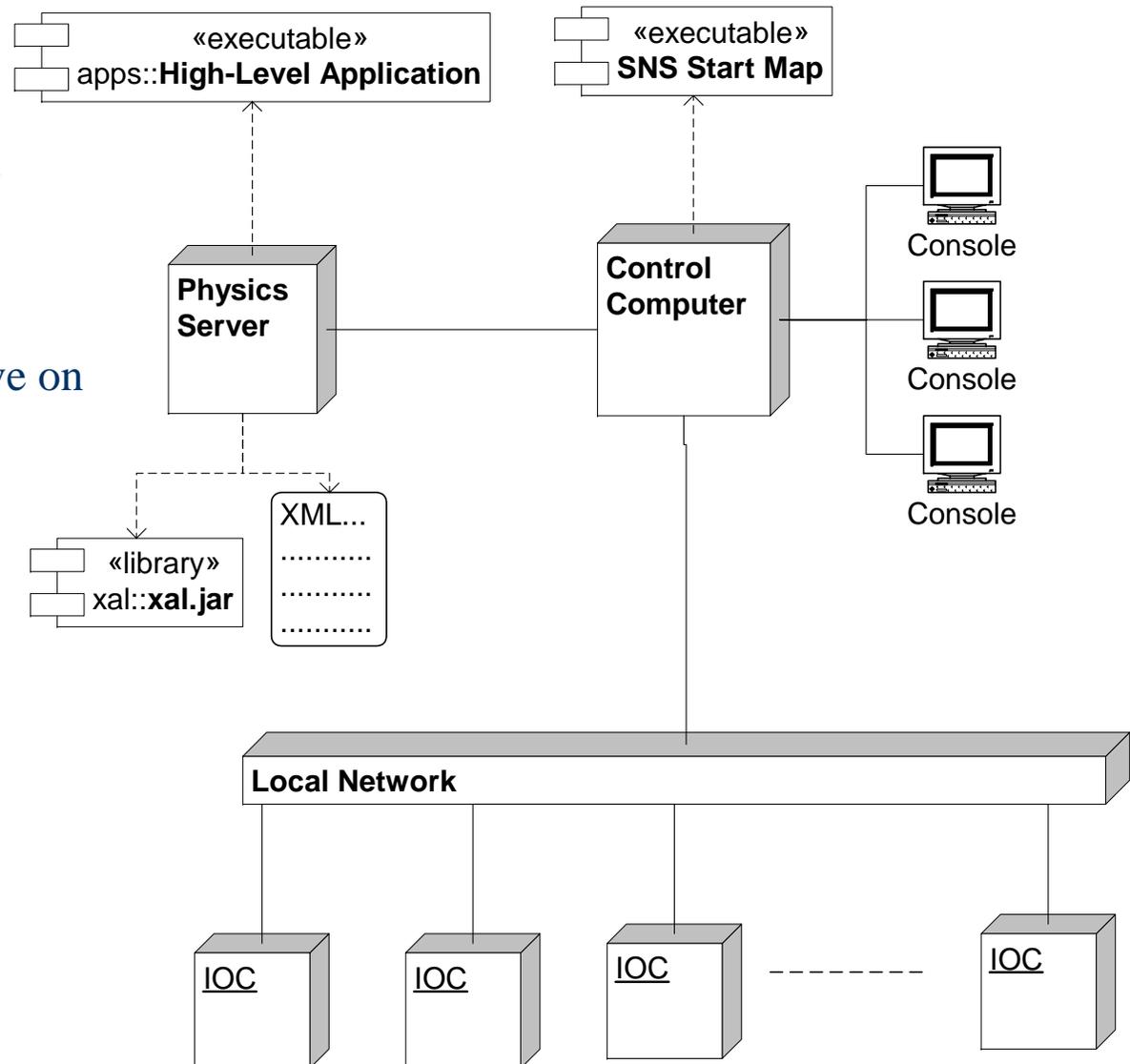
# Tools (gov.sns.tools)

- Correlator (gov.sns.ca.correlator)
  - Used to collect PVs with the same timestamp
- Services
  - Used for internal communication between XAL processes (clients – services )
- Database
  - Simple database communication tools (pv-logger uses this)
- XML
  - Easy interface to write / read structures, used for open/save capability for application documents
- ...

# SNS HL Control Deployment

### Physics Server

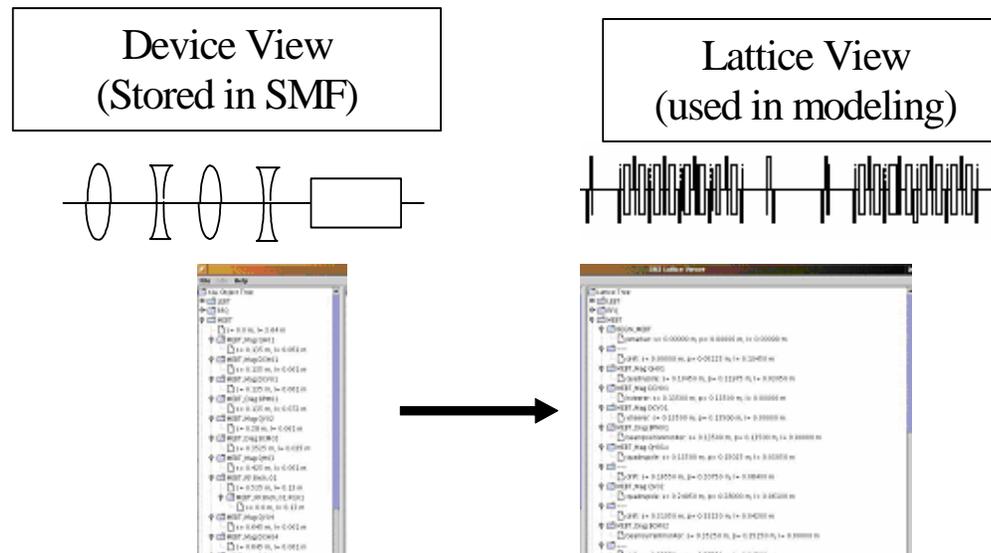
- XAL applications deployed on separate node
- XAL services and monitoring agents live on physics server



# 3. Simulation Subsystem

## 3. Online Simulation

- Lattice Generator (gov.sns.xal.slg)
  - A model of the machine is constructed from the “device” representation
  - Drifts are added, elements are split
- Simulation **Scenario** (gov.sns.xal.model.scenario)
  - Lattice element values can be updated from the machine, user defined values or design values
  - Mostly use an envelope model for linac tracking at present



- Single entry per element

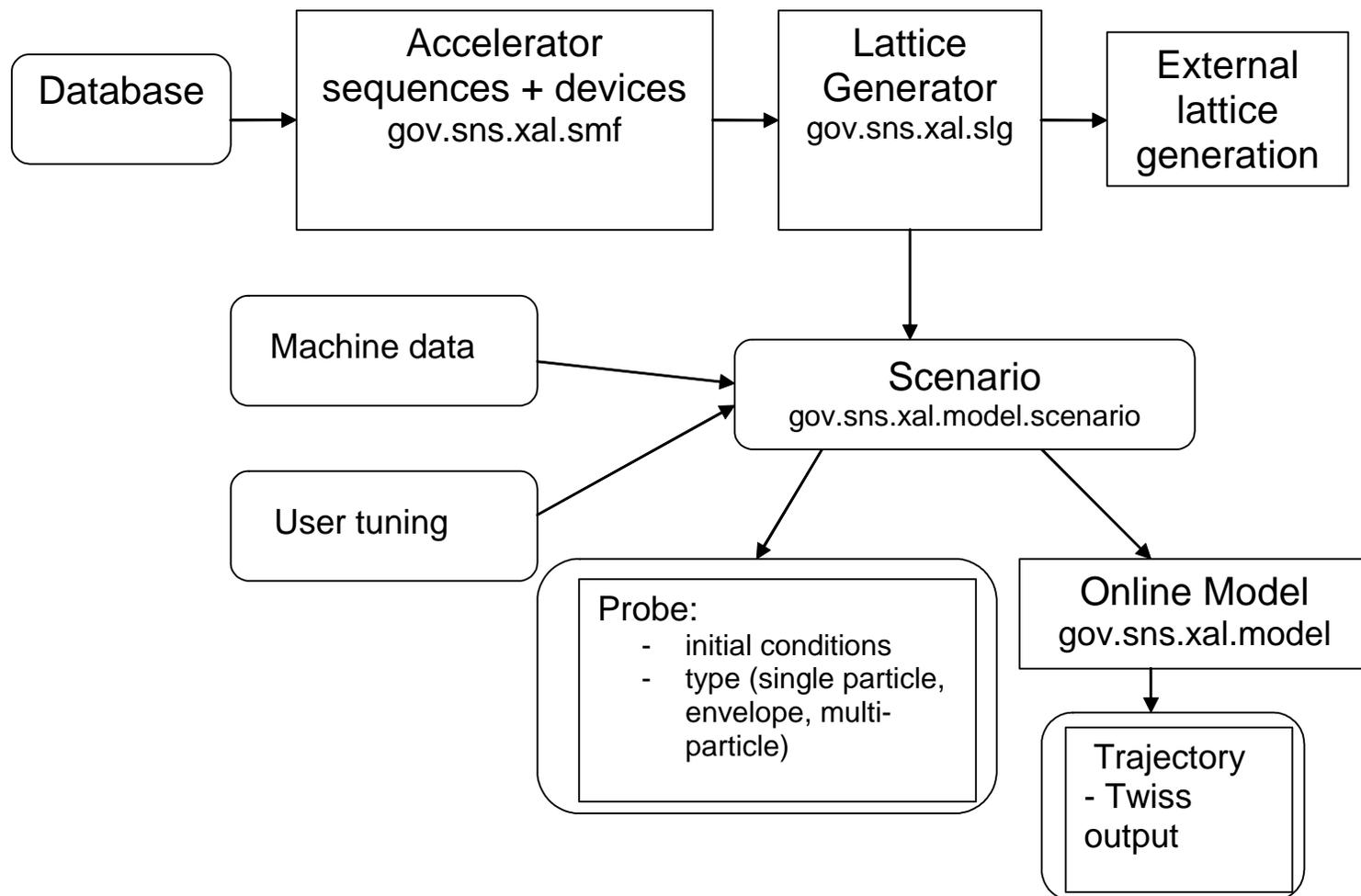
- Only physical devices

- Elements may be split

- Includes drifts

# Simulation Subsystem Architecture

(Allen, Klotz, Galambos, Pelaia, Chu, McChesney)

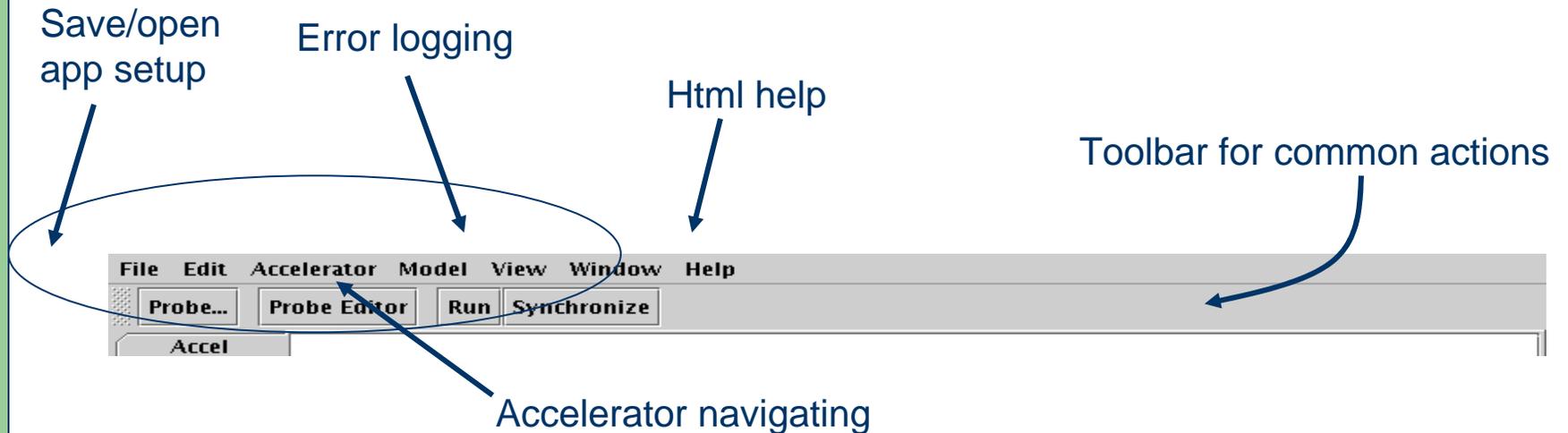


# Online Model (gov.sns.xal.model)

- Online beam simulation (i.e., no “decks”)
  - See LA-UR-02-4979 for theory
- Uses N. Malitsky’s **Element/Algorithm/Probe** architecture:
  - **Element**: Machine representation
    - basic objects that affects the beam (e.g. quad, thin lens, drift)
  - **Probe**: Represents the beam (or aspects of the beam)
    - Envelope, Single-particle, Multi-particle, etc.
  - **Algorithm**: how to propagate the beam
    - Binds **Probe** to **Element** and provides dynamics
    - Linear dynamics, higher-order dynamics, etc.

# 4. XAL Application Framework

(T. Pelaia)



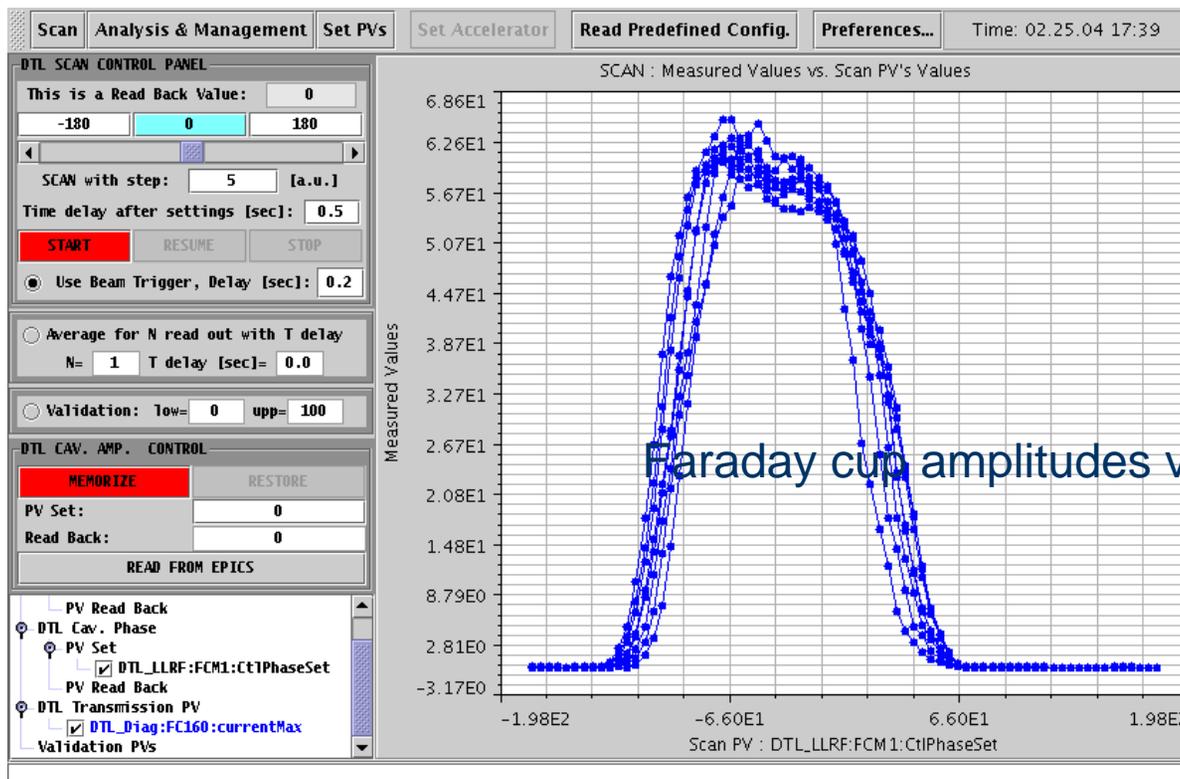
Application Framework used as a common starting point

- Provides a common *look and feel* for all applications
- Quick jump-start for application development
- Easy retro-fixes across many apps
- Incorporates a document/view/control architecture (Software Engineering!)

# 1-D Scan Application

(A. Shishlo)

- Provides an easy way to scan one quantity and monitor others
- Can average over pulses, scan multiple times, pause
- Analysis includes fitting, intersection finding, min/max, etc.
- Easy way to do a quick unanticipated experiment
- Predefined scans with specialized analysis are possible
  - DTL and MEBT phase + amplitude setting applications



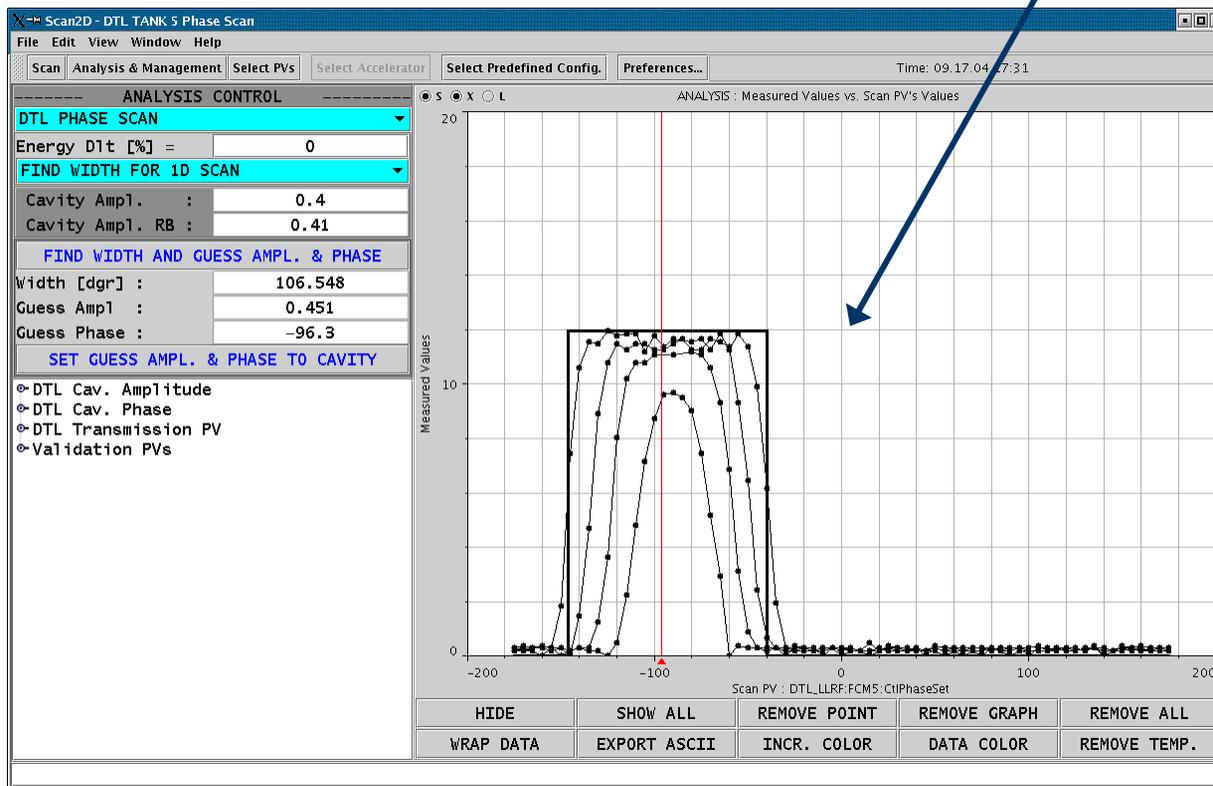
Faraday cup amplitudes vs. DTL RF settings

# 2-D Scan Application

(A. Shishlo)

## 4. Appl Framework

- Added 2-D scan capability (parametric scans)
- Define preset configurations
  - Optional analysis capability
- Used for DTL acceptance scans for phase/amplitude setting
  - Scan DTL phase + amplitude, monitor Faraday Cup signal

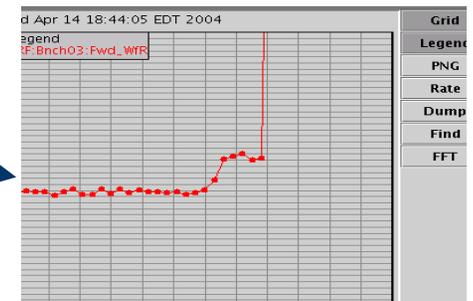
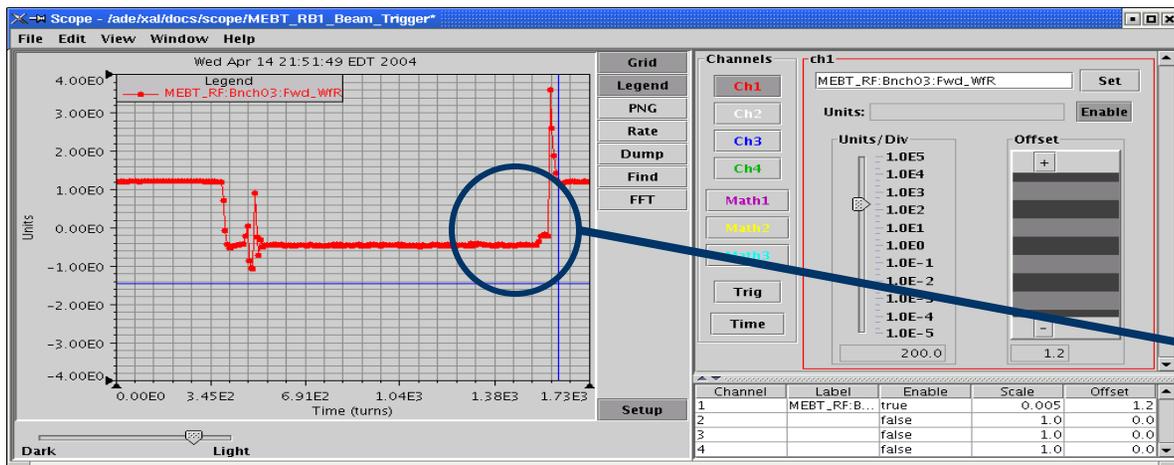


# Scope Application Triggered Acquisition

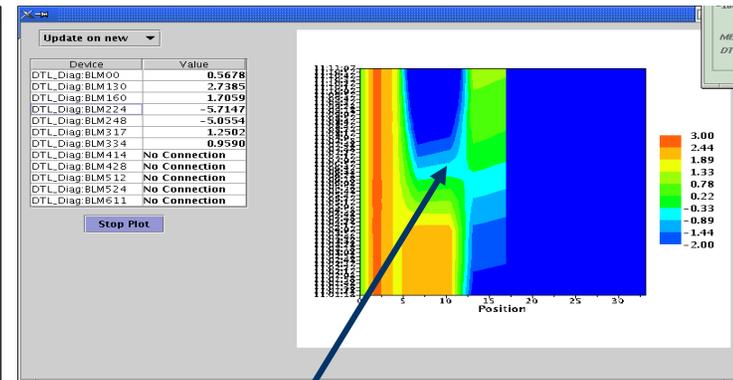
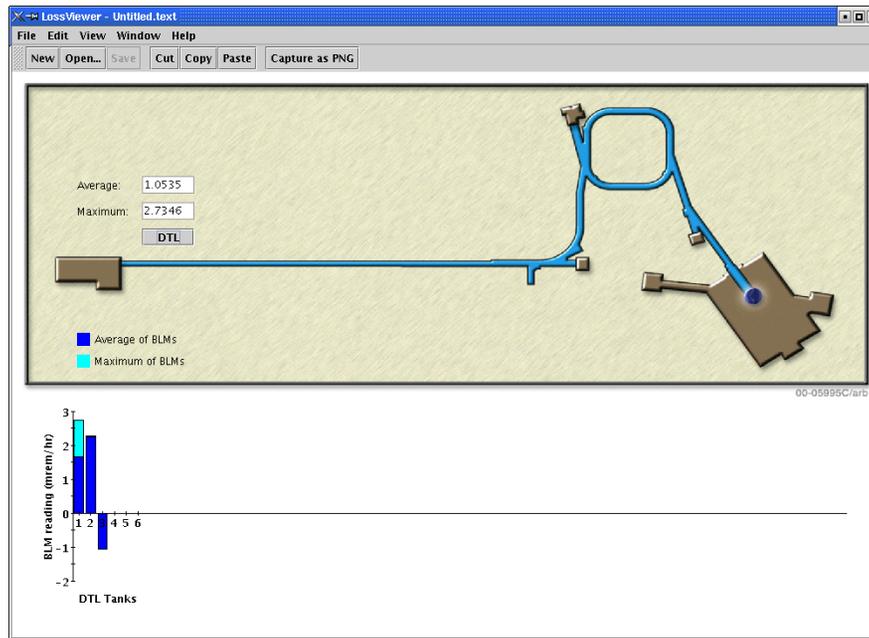
(T. Pelaia)

## 4. Appl Framework

- The Digital Oscilloscope – with a similar user interface as analog scopes
- MEBT rebuncher (RF) forward power trace with beam loading:
  - RF = 1 msec @ 20 Hz, beam = 50 msec @ 1 Hz
  - Use the correlator to filter only RF signals with beam pulses
- Potential for future applications
- Requires vigilance on good signal time stamps and proper time waveform packaging



# Loss Viewer Application (S.Cousineau)

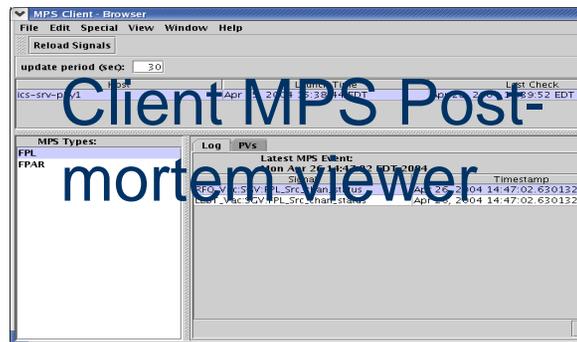


- View a summary of beam loss by machine section
  - “Zoomable” to specific BLMs
  - Viewable as fraction of permissible loss
- Waterfall display of a specific beamline portion
  - Faraday cup inserted here

# Service Applications

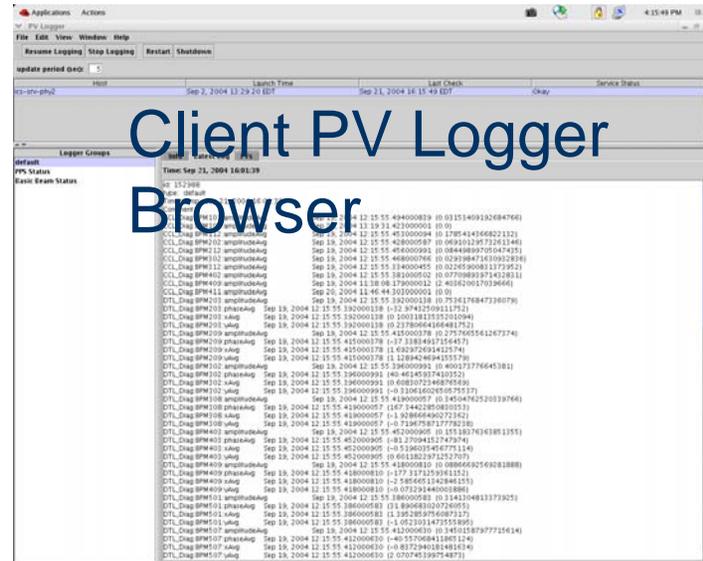
(T. Pelaia)

- MPS post-mortem (tracks first fault incidents)
  - posts statistics to elog daily
  - history views

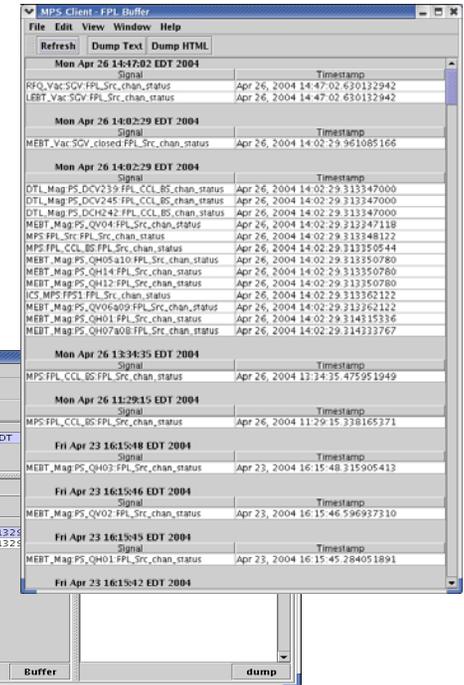


- PV Logger – posts sets of data to the database

- periodically or pushbutton
- Generalized to allow for custom PV sets



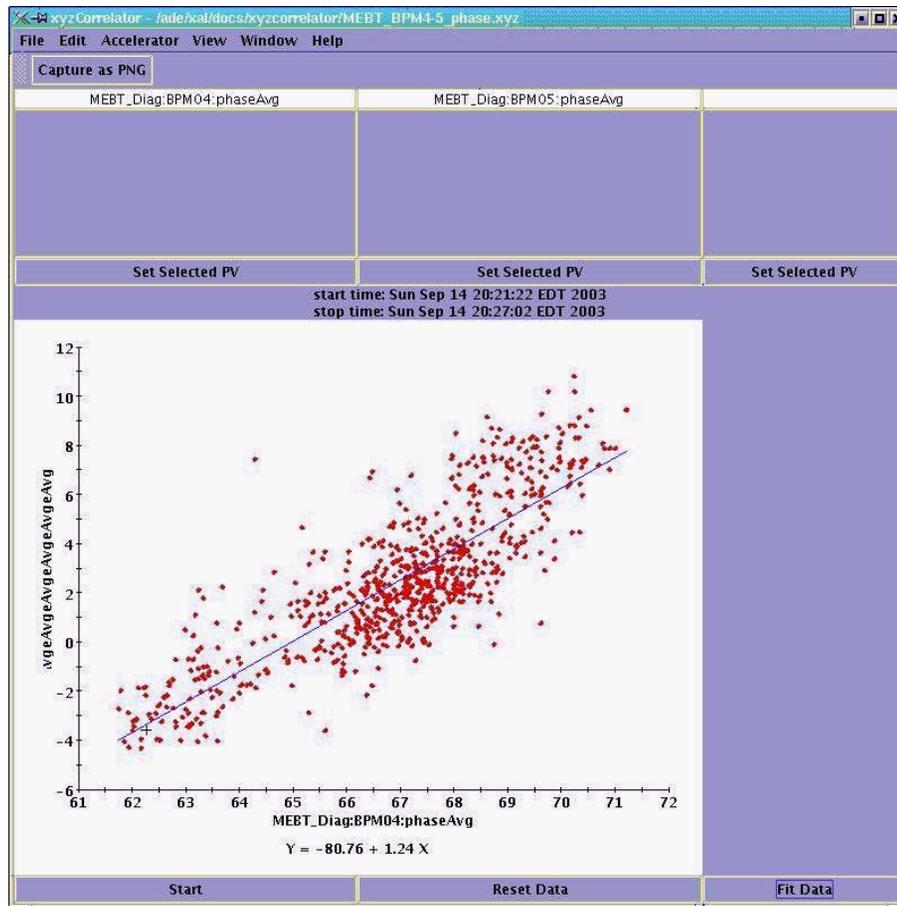
## 4. Appl Framework



# XYZ Correlator Application

(P. Chu)

- Pick 2 (or three) signals and monitor them together
- Can use the time correlator to ensure signals are from the same pulse
- Added customization features + added to the framework
- Can export or fit the acquired data



Correlation between phase measurements of 2 BPMs

# Save-Compare-Restore (Score) Application

(J. Galambos)

- Provides a means to capture machine setup, compare live values to a saved set, and to restore values to a saved set
  - Grabs settable + readback signals
- Can sort by system and device type

- This is the primary means to snap-shot the machine state

Load devices/types		Open...	Save As...	Snapshot Machine	Restore	Capture as PNG	
Select Systems		RFQ	MEBT	DTL	Timing	FE	DPlate
Type	Setpoint name	SP Save Val	SP live Val	Readback Name	RB Save Val	RB live Val	
RF	RFQ:RF:Gain	0.3500	0.3500				
	RFQ:RF:Gain_Rot	116.9083	116.9083				
	RFQ:RF:Int_scale	7000.0000	7000.0000				
	RFQ:RF:Loop	1.0000	1.0000				
	RFQ:RF:cavAmpSet	0.5512	0.5512	RFQ:RF:cavAmpAvg	0.5488	0.5493	
	RFQ:RF:cavPhaseSet	24.3920	24.3920	RFQ:RF:cavPhaseAvg	24.3090	24.2417	
	RFQ_HPRF:Mod1:VCTL_Set	130.0000	130.0000	RFQ_HPRF:Mod1:V_Mon	100.3780	100.3750	
				RFQ:RF:FwdPower	686.1238	673.1348	
				RFQ:RF:RflPower	9.4841	9.0523	
				RFQ_HPRF:Mod1:L_Mon	48.8328	48.8791	
Temp	RFQ:ChlIr_2:T_Set	24.2000	24.2000	RFQ:ChlIr_2:T	25.2192	25.3264	
				RFQ:ChlIr_2:T_LB	24.1819	24.1849	
Vac				RFQ_VacIG_2:P	2.710E-7	2.694E-7	
				RFQ_VacXV:Sts	1.0000	1.0000	

Machine data saved at Sat Aug 30 19:36:59 EDT 2003

# Xio Application (P. Chu)

- General purpose value displayer (tables, and or plots)
- Added to application framework, works for MEBT + DTL

The image shows three overlapping windows from the Xio application. The top-left window displays the 'Accelerator' menu with options: Load Default Accelerator, Load Accelerator, Sequences (with sub-options LEBT, RFQ, MEBT, DTL1, and MEBT-DTL1), and Combo Sequence. A blue arrow points from this menu to the top-right window. The top-right window shows the 'TypeAndHandleSelector' dialog with 'Device Types' (QH, DCH, DCV, BPM) and 'BPM' sub-options (phaseAvg, xAvg, yAvg). A blue arrow points from this dialog to the bottom-right window. The bottom-right window shows a table of BPM data with columns for BPMs, xAvg, and yAvg. A blue arrow points from this table to the bottom-left window. The bottom-left window is a plot titled 'MEBT-DTL1 BPM Plot' showing data points and averages for x and y directions over a distance of 9 km. A blue arrow points from the plot back to the table in the bottom-right window.

*Pick accelerator*

*Pick device + signal types*

←MEBT + DTL ->

BPMs	xAvg	yAvg
MEBT_Diaq:BPM01	-0.0028	0.0154
MEBT_Diaq:BPM04	0.0126	0.0073
MEBT_Diaq:BPM05	-0.0144	0.0127
MEBT_Diaq:BPM10	0.0193	-0.0006
MEBT_Diaq:BPM11	0.0068	0.0023
MEBT_Diaq:BPM14	0.0066	-0.0005

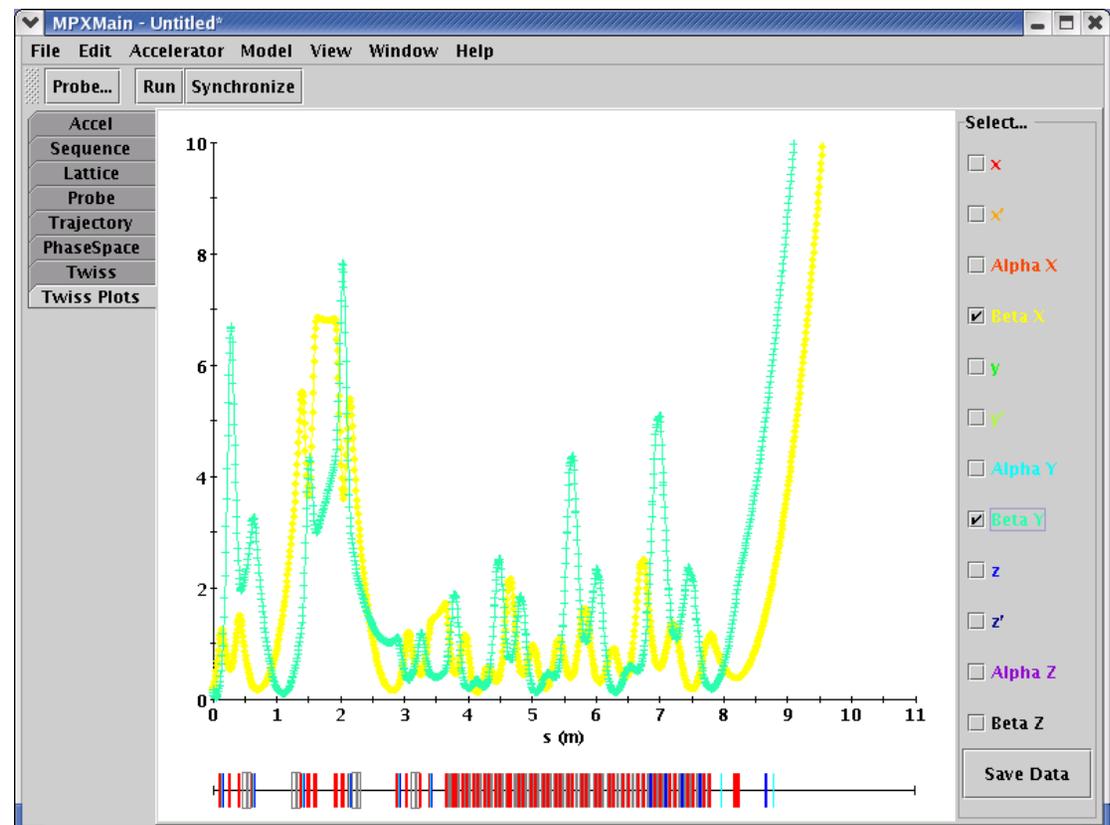
# Online Model Application

(W.-D. Klotz)

## 4. Appl Framework

- An online model is now available within the XAL framework
- Online model can be run for 1) live values, 2) design values and 3) user defined what-if changes
- Can display or dump beam Twiss output

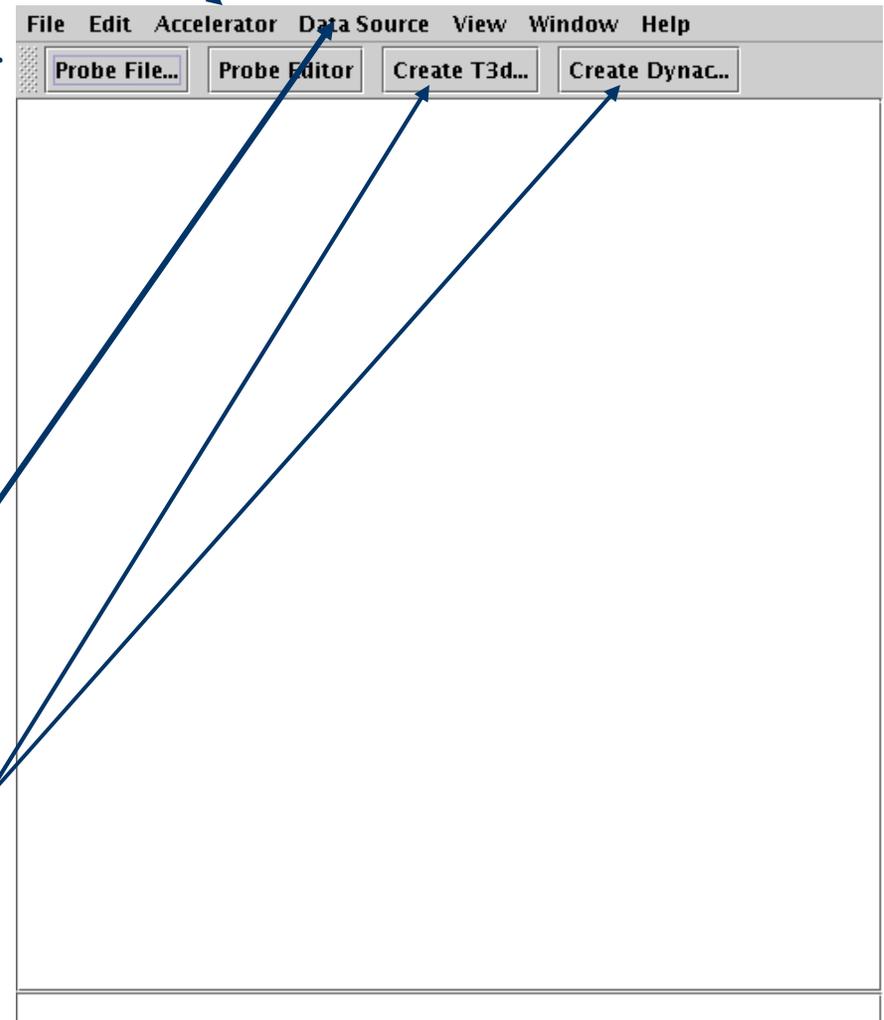
• Vertical and horizontal beta functions through the MEBT, DTL + D-plate for design values



# External Lattice Generator

(P. Chu)

- Generate an external lattice file for selected part of the accelerator
- Uses the online model lattice generation
- Tested though DTL (+ HEFT)
- Can use live or design values
- Trace3D or Dynac input files can be created.
- May be incorporated into the online model application.

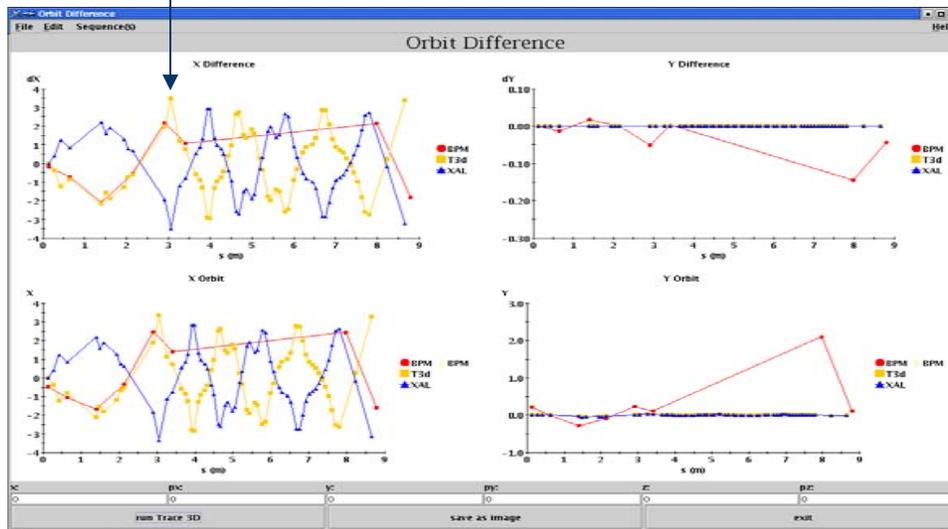


# Orbit-Difference Application

(P. Chu)

- Compares differences in beam orbits, for both BPMs and calculated
- Online model is also used in the Orbit Difference Application, in addition to running Trace 3D (external fortran code)

Kick applied here



- Orbit difference example using the online model
- Used to observe orbit difference in the horizontal direction
- Helped resolve sign issue in BPMs

# Summary

- The XAL application programming infrastructure is in place and working at SNS
  - > 20 applications written – actively used in commissioning activities
  - Online model used more extensively
- SLAC will employ XAL in the LCLS
- Issues
  - Java – “heavier” than typical X-widgets.
  - Need to revisit some applications to make them more efficient.

### “Device View” Environment

- Class structure provides hierarchical “device view” of the accelerator to the application programmers
  - Sequences (MEBT, DTL, CCL, ...)
  - Individual Node type classes (Quads, RF, BPMs, etc.)
- Sequences can be “pasted together” as ComboSequences .

