XAL and Online Modeling

Paul Chu

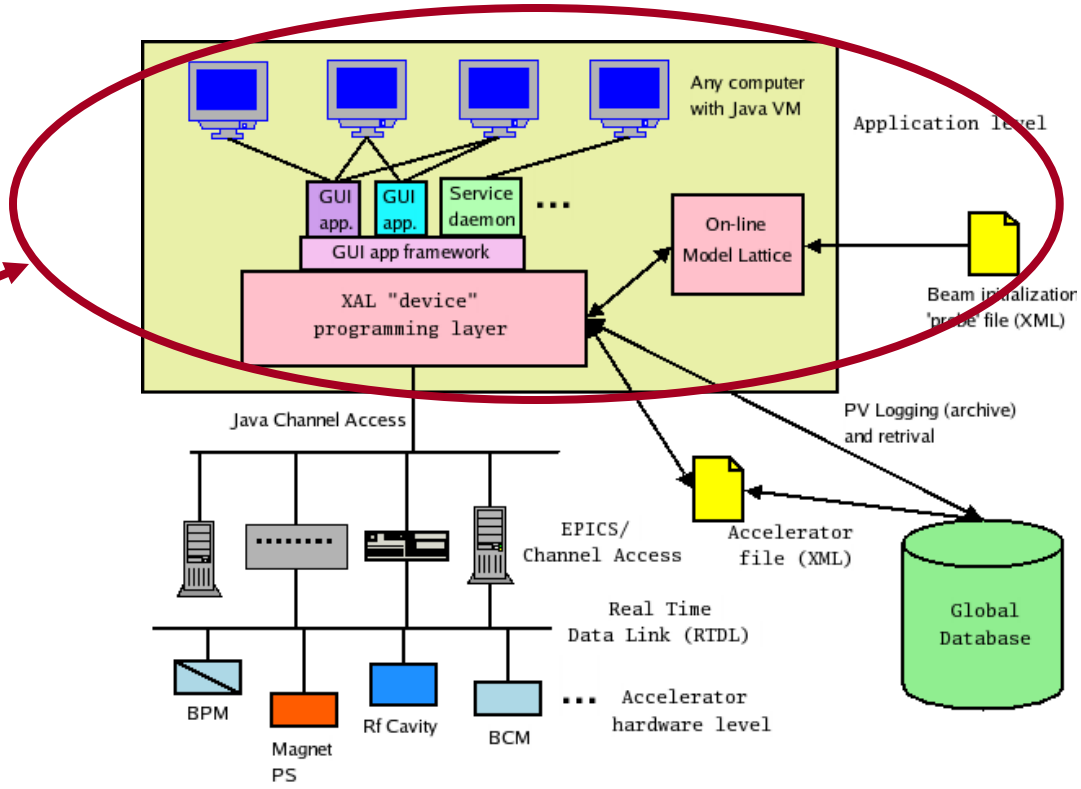
XAL/Online Model

- Introduction
- Online model
- Matlab demo
- XAL app demo
  - Online model app
  - Knob app

XAL Overview

- High level physics application software is for modeling, integrated operation and accelerator physics studies
- Application software environment --

XAL and its apps



XAL Accelerator Hierarchy

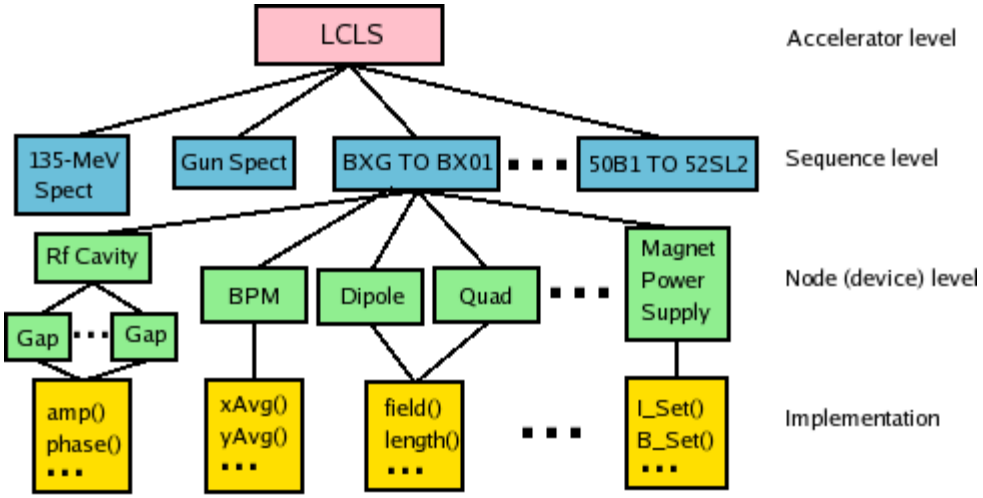
- XAL is Java based
- Includes a class hierarchy describing the accelerator structure
- Methods exist to directly work with accelerator devices

Java.lang.Object

AcceleratorNode (base class)

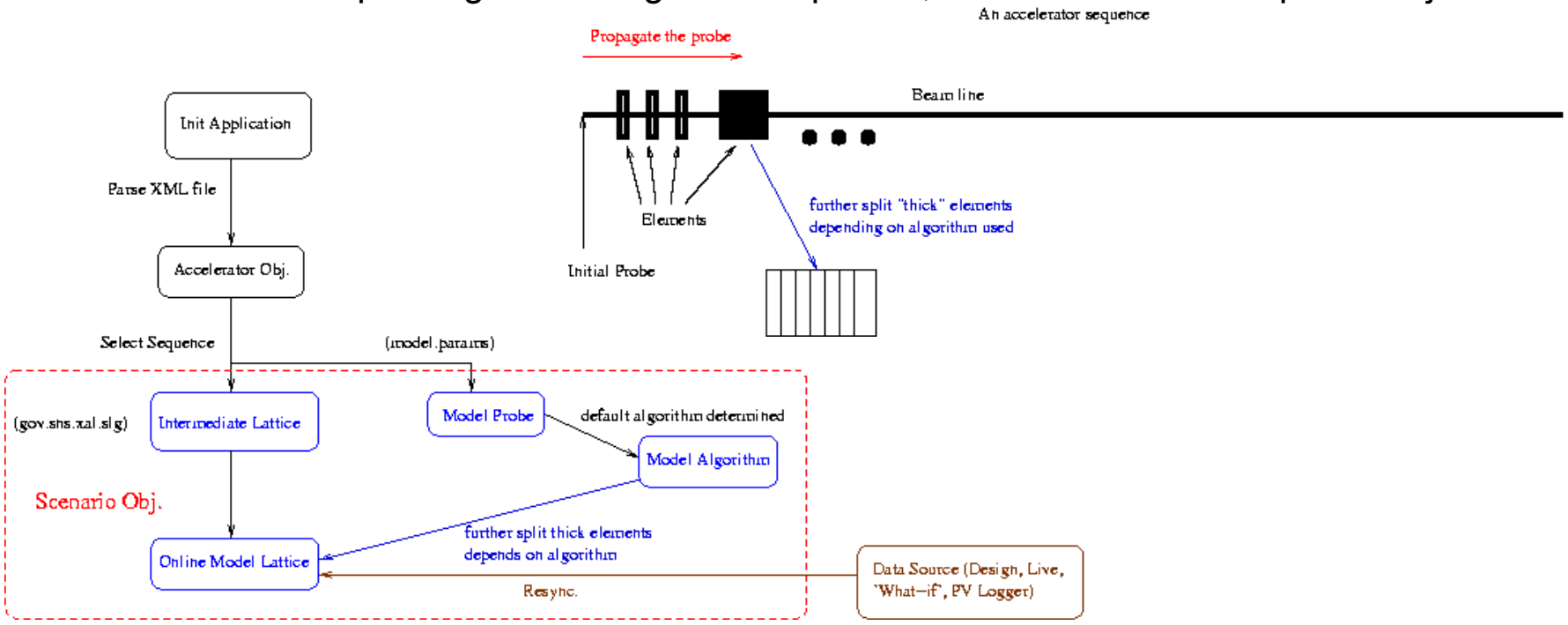
AcceleratorSeq

Accelerator



XAL – How does it work?

- XAL online model based application flow diagram
  - Prepare a Scenario object
  - Run model
    - Propagate the probe through the sequence
    - When encounter an element, apply the corresponding “action” (algorithm)
    - After the probe gone through the sequence, all calc info is in the probe object



XAL in Matlab

- Prepare Java class paths to ***xal.jar*** and ***ext.jar***.
- ***Import*** statements.
- Read in XAL XML file
  - *theAccelerator = XMLDataManager.acceleratorWithPath('/usr/local/lcls/physics/xal4lcls/xal_xmls/main.xal');*
- Select section(s) to work with
  - Define 1st section: *seq1 = theAccelerator.getSequence('BXG TO BX01');*
  - Define 2nd section: *seq2 = theAccelerator.getSequence('BX01 TO TD11');*
  - Put two sections together:
    - *seqList = ArrayList();*
    - *seqList.add(seq1);*
    - *seqList.add(seq2);*
    - *comboSeq = AcceleratorSeqCombo('2Sections', seqList);*

XAL in Matlab (cont.)

- Get device collection with qualifier
  - BPMs: *bpms = seq.getNodesOfType('bpm');*
  - Good BENDs + QUADs:
    - *Import gov.sns.xal.smf.impl.qualify.*;*
    - Define 'qualifiers':
      - *orTypeQual = OrTypeQualifier();*
      - *andTypeQual = AndTypeQualifier();*
    - For BENDs: *orTypeQual.or('dh');*
    - For Quads: *orTypeQual.or('quad');*
    - Exclude bad ones:
      - *andTypeQual.and(QualifierFactory.getStatusQualifier(true));*
    - Put together: *andTypeQual.and(orTypeQual);*
    - Result: *goodMags = seq.getAllNodesWithQualifier(typeQual);*

How to Run Online Model

- Prepare a "scenario" first
  - *scenario = Scenario.newAndImprovedScenarioFor(seq);*
- Set *probe* (from a probe file or model.param): probe also determines which algorithm to use
  - From model.param: *probe = ProbeFactory.getEnvelopeProbe(seq);*
  - From file: *probe = ProbeXmlParser.parse(file.getPath());*
  - *scenario.setProbe(probe);*
- Lattice synchronization (design, live)
  - *scenario.setSynchronizationMode(Scenario.SYNC_MODE_XXX);*
    - XXX can be DESIGN, LIVE, or RF_DESIGN.
    - PVLogger is also supported via 'what-if' method.
  - *scenario.resync();*
- Run model
  - *scenario.run();*
- Get results
  - Trajectory: *traj = probe.getTrajectory();*
  - Probe State: *states = traj.statesForElement(elementID);*
  - Physics Parameters: *twiss = states[0].getTwiss();*
  - R-Matrix: *R_mat = traj.getTransferMatrix('BPMS:IN20:425', 'BPMS:IN20:511');*
    - *R_mat_11 = R_mat.getElem(0, 0)*

Paul Chu
pchu@slac.stanford.edu

How to Run Online Model (cont.)

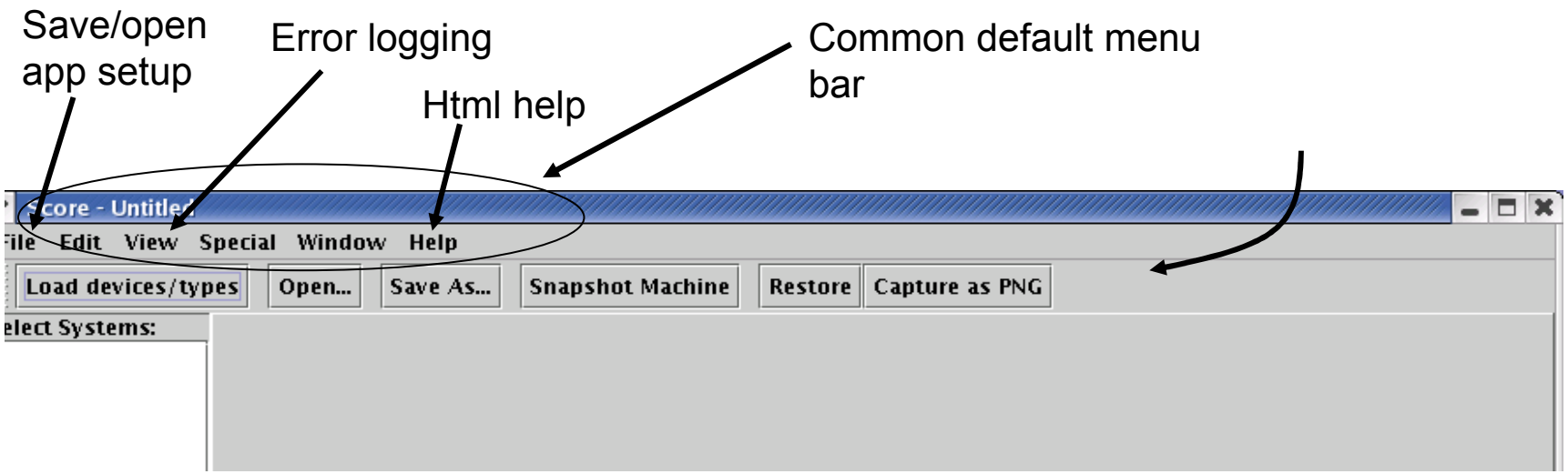
- Use ‘what-if’
  - Find which magnet (RF) you want to change:
    - *mag = seq.getNodeWithId(‘QUAD:IN20:781’);*
  - ***import*** *gov.sns.xal.smf.proxy.*;*
  - Set test value: *scenario.setModelInput(mag, ElectromagnetPropertyAccessor.PROPERTY_FIELD, testValue);*
  - Re-sync and re-run the model.

Channel Access

- Define path to libjca.so in librarypath.txt if using JCA/JNI (default on the CA network).
- Connect to a BPM PV:
  - *bpm = seq.getNodeWithId('BPMS:IN20:771');*
  - Get this BPM's horizontal reading: *bpm.getXAvg()*
- Connect to any PV:
  - *Import gov.sns.ca.*;*
  - Create a 'Channel Factory': *cf = ChannelFactory.defaultFactory();*
  - Create a Channel: *ch = cf.getChannel('BPMS:IN20:771:X1H');*
  - Get the channel value: *ch.getValDbl*

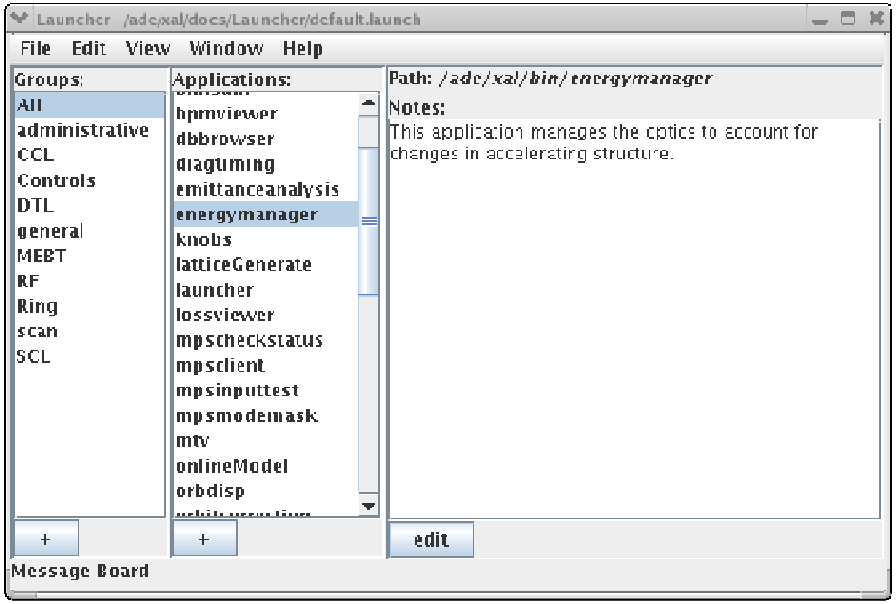
XAL Application Basics

- Common look-and-feel for all apps.
- GUI framework provides many functionalities for free.
  - Cut/paste editing feature
  - Last 10 documents opened are shown in the “File” pull-down menu.
  - Screen capture as PNG file, memory usage, etc.
- Document based: can run multiple “documents” at the same time, can save settings to documents.



XAL Application Basics (cont.)

- Standard HTML-based help.
- Each app can build its own jar file (should not have any dependency among apps).
- A “launcher” application for novice users.
  - The launcher content is a saved document.



XAL Application Basics (cont.)

- Application launching script
  - Example:

    > java –DuseDefaultAccelerator="true" –Xmx120m $XAL_APPS/virtualaccelerator.jar

  This script will load the default accelerator and save a click, and allocate more memory for the JVM.

Documentation

- XAL Java Doc
  - http://www.slac.stanford.edu/grp/lcls/controls/docs/physics/xal/javadoc/
- Matlab script location
  - /home/softegr/pchu/matlab/
- Many more to come…