MPS GUI

Setup

Environment

- Check out the CVS module physics/mps/mpsgui into your workspace
    - From now on, the root of project is referred to as $MPSGUI_ROOT
- You can use eco to checkout the package. The name of the module is mpsgui.
- Run ant to build the mpsgui and mpscud jar files.
- The mpsgui and mpscud scripts should use the jar files built in the jar folder.

Dependencies

For library paths, see the build.xml file. Note that the dependent jar files are copied into the lib folder and are part of CVS. To update the jar files, run ant update_dependencies. This should copy the files from $PHYSICS_TOP on dev onto the lib folder. You can then commit these updated dependent jar files into CVS.

- ezJCA (version R0-0-10)
- javainterfaces (current version)
- MPS Config(current version)
    - Oracle JDBC driver
    - Sqlite JDBC driver
- SwingUtil (current version)
    - jgoodies.jar can be added at runtime for the Matlab look & feel
- xal4lcls
    - ext.jar (current version)
    - xal.jar (current version)

Test

- Test using the mpsgui and mpscud scripts from the command line. This sets the Oracle Wallet environment variable and other EPICS variables.

Development

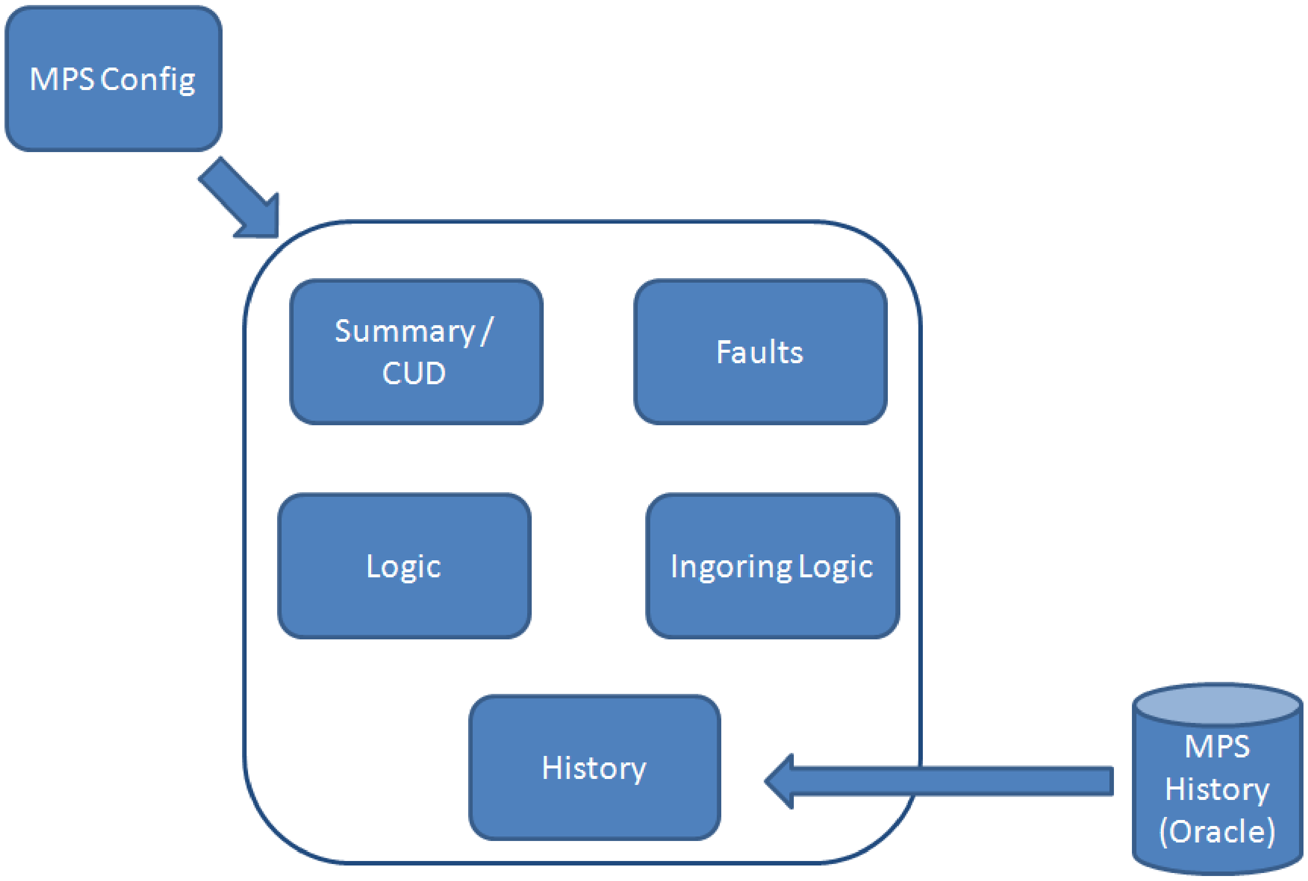
Major Recurring Tasks

- Previously, one had to check the live history in the MPS GUI HistoryViewer (and if empty, restart the MPS History Server). However, now there is a CRON job that monitors this and will trigger alarms for the operators

Tips

- Find the Java field that contains the UI component that prompted your development (change the back-/foreground color, if not sure); then use Eclipse References (right-click menu) to find where the field is being used

Overview



Launch Scripts

mpsgui

- Located under $MPSGUI_ROOT, must be executable (chmod +x)
- Sets up the ORACLE Wallet and the log file
- Launches MPSGui using Plastic3DLookAndFeel from jgoodies
- byenti is a legacy system property that tells MPSGui to interpret bypass end times as absolute (or, if false, relative) times

mpscud

- Located under $MPSGUI_ROOT, must be executable (chmod +x)
- Sets up the Oracle Wallet and the log file
- Launches MPSGui in CUD mode using Plastic3DLookAndFeel from jgoodies
  - Operators and Mike Z know details about the CUD mode
- byenti is a legacy system property that tells MPSGui to interpret bypass end times as absolute (or, if false, relative) times

Classes and Functions

User Interface

- The main user interface consists of 5 tabs: Summary, Faults, Logic, Ignore Logic, and History
- Each tab has a corresponding package under edu.stanford.slac.mpsgui.ui
  - Classes in those packages are complex Swing components (mostly, JPanels and JTables)
- Each XYZPanel class under edu.stanford.slac.mpsgui.ui is a subclass of edu.stanford.slac.mpsgui.ui.AbstractPanel
  - (Almost) every widget of the panel is stored in a field in the corresponding XYZPanel class, where XYZ is a descriptive string that explains what the panel shows
  - The fields are public to allow registration of the listeners (however, it doesn't make sense to set them)

edu.stanford.slac.mpsgui.ui.AbstractPanel

- The root of all MpsGui XYZPanel classes

- Defines how widgets are created/added to its subclass

edu.stanford.slac.mpsgui.ui.MacroDetailsPanel

- Shows the details of a Macro
- Part of IngoreLogicPanel and LogicPanel

edu.stanford.slac.mpsgui.ui.MpsGuiPanel

- The main panel that is added to the JFrame
- Contain a JTabbedPane as the immediate child

edu.stanford.slac.mpsgui.ui.MpsGuiTable

- The root of all MpsGui XYZTable classes
- Features a TableSorter (see SwingUtil) and alternating background colors for rows

edu.stanford.slac.mpsgui.ui.SliderCachePanel

- Allows user to select a cached state

edu.stanford.slac.mpsgui.ui.edm.EDMPanel

- Not used anymore

edu.stanford.slac.mpsgui.ui.fault.AbstractFaultPanel

- The root of all XYZFaultPanel classes
- Shows detailed information common to all types of faults

edu.stanford.slac.mpsgui.ui.fault.CurrentFaultsPanel

- Not used anymore

edu.stanford.slac.mpsgui.ui.fault.EPICS/LinkNodeChannel/LinkNode/LinkProcessorFaultPanel

- Shows information specific to each fault type

edu.stanford.slac.mpsgui.ui.fault.MainFaultPanel

- Features a tree-like browser for faults and a placeholder for the detailed information

edu.stanford.slac.mpsgui.ui.logic.StatesCorePanel

- Contains the text area for HTML that represents the MacroStates (obtainable via MPS Config)

Controllers

edu.stanford.slac.mpsgui.controller.edm.EDMViewer

- Not used any more

edu.stanford.slac.mpsgui.controller.fault.CurrentFaultsViewer

- Not used any more

edu.stanford.slac.mpsgui.controller.fault.MPSFaultsViewer

- Controller for the "Faults" tab
- A tree-like browser for faults hierarchies
- Shows the current information about a selected fault

edu.stanford.slac.mpsgui.controller.hist.MPSHistoryViewer

- Controller for the "History" tab
- Shows live MPS history
    - Default is the last 8 hours (see edu.stanford.slac.mpsgui.MpsGuiProperties)
    - Periodically checks for new MPS History messages
    - If the live history panel is empty, please trouble-shoot the MPS History Server
- Features a panel for an interactive search of the MPS history
    - Filter applies to fault names only
    - Launches new search on "key typed", tries to cancel the previous search (may break in the future)

edu.stanford.slac.mpsgui.controller.ignorelogic.IgnoreLogicViewer

- Controller for the "Ignore Logic" tab
- Monitors the state of ignoring macros (using MPS Config)
    - Caches previous states
- Shows detailed information about ignoring and ignored macros
- To test, ignore/unignore macros by writing 0 or 1 to YAGS:IN20:211:IN_LMTSW_BYPV

```
caput YAGS:IN20:211:IN_LMTSW_BYPV 1 to ignore
caput YAGS:IN20:211:IN_LMTSW_BYPV 0 to unignore
```

edu.stanford.slac.mpsgui.controller.logic.MPSLogicViewer

- Monitors states of all macros
    - Caches previous states
- Shows detailed information about a selected macro
- Allows user to bypass a selected macro to a state

edu.stanford.slac.mpsgui.controller.summary.SummaryViewer

- Controller for the "Summary" tab
- Initializes the logic "viewer" for the integrated logic panel
- If applicable, styles the summary UI as a CUD (larger font etc.)
- Monitors beam rate PVs (both MPS and actual)
- Monitors bypassed faults and end times; warns (using proper colors) when bypasses are about to expire
- Allows user to re-bypass to a bypassed macro state

Others

edu.stanford.slac.mpsgui.MpsCudController

- Creates the summary UI, tries to setup as a CUD
- Initializes the summary "viewer" (controller)

edu.stanford.slac.mpsgui.MpsGuiController

- Creates the full UI
- If successful, initializes "viewers" (controllers) for each tab

edu.stanford.slac.mpsgui.MpsGuiLauncher

- The main class expects the following arguments (in the order):
    - Config DB URL prefix
    - Logic DB URL prefix
    - History JDBC URL (including username and password)
    - See $MPSGUI_ROOT/mpsgui for an example
- Loads Config and Logic DB via MPS Config
- Depending on the system property slac.launchAsCud, initializes either MpsCudController or MpsGuiController

edu.stanford.slac.mpsgui.MpsGuiProperties

- Contains constants/properties used by MPSGui, such as:
    - Application name, version number, NULL string, date formats, background color for alternate table rows, etc.

edu.stanford.slac.mpsgui.jdbc.hist.HistoryDB

- Client for the MPS History data (currently in Oracle, used to be in sqlite)
- SQL queries are located in the file oracle.properties in the package edu.stanford.slac.mpsgui.jdbc.hist
- Has an auxilliary API to insert/delete test messages

Release to production

- Update the application version in $MPSGUI_ROOT/src/edu.stanford.slac.mpsgui.MpsGuiProperties
- Add a note in $MPSGUI_ROOT/RELEASE_NOTES
    - Increment the tag version accordingly
- Commit to CVS
- Tag with mpsgui-R###
- Check out the tagged version on development using eco, and build with ant:

```
$ cd ~/workspace
$ eco
Enter name of module/package to checkout: mpsgui
Enter name of tag or [RETURN] to use HEAD>mpsgui-R0-1-23
Using mpsgui-R0-1-23. The name of the directory will be mpsgui-R0-1-23.
cvs checkout -P -r mpsgui-R0-1-23 -d mpsgui-R0-1-23 mpsgui
cvs checkout: Updating mpsgui-R0-1-23
...
$ cd mpsgui/mpsgui-R0-1-23/
$ ant
....
jar:
      [jar] Building jar: /afs/slac.stanford.edu/u/cd/mshankar/temp/test/docTest/mpsgui/mpsgui-R0-1-23/jar
/mpsgui.jar

BUILD SUCCESSFUL
$ $ ls jar/mpsgui.jar
jar/mpsgui.jar
$ ./mpsgui
```

- Use cram push and cram upgrade to push and upgrade to this release

  ```
  $ cram ls
  Current version on facility: LCLS  => mpsgui-R0-1-22
  Current version on facility: FACET  => None
  Current version on facility: TestFac  => None
  Current version on facility: Dev  => mpsgui-R0-1-22
  $ cram push
  ....
  $ cram upgrade -f Dev mpsgui-R0-1-23
  ....
  ```

- Launch from lclshome
  - MPS Global => MPS GUI...
  - MPS Global => MPS CUD...
- Note that the mpsgui script redirects all logs to files elsewhere. During development and testing, it is often fruitful to have the logs on the console.