Tools to be used with the Panel Editor

The aim of this part of the tutorial is to show examples and suggest solutions to common problems that we have found during the development of the VLT Graphical User Interfaces.

Tools used during GUI development

- **seqWish, this can be attached to the panel editor to design some parts which are not automatically supported by the tool -e.g. graphics**

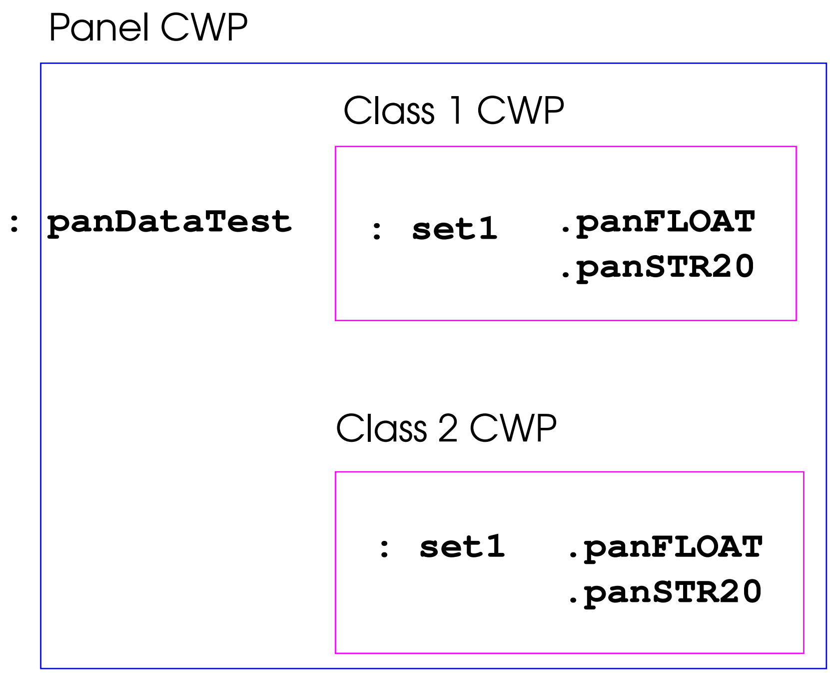
- **tkinspect, at the moment this is the only way to get inside a running GUI application to add 'print' like DEBUG statements ... good old FORTRAN times.**

- **TclPro, debugger for tcl/tk application .. Not Public Domain .**
  It is also able to debug methods associated to incrTk classes .. not yet part of our standard tools

uifClass - Basic Concept

The panel editor support the use of 'objects' allowing the GUI developer to design 'panels' that can be shared by other interfaces, thus minimizing the maintenance and upgrading of a large number of panels.

The aim is to have a widget that can be configured to read the attribute panSTR20 from point set1 or point set2



The panel CWP is set to **`:panDataTest`**, that is, all the DB attributes must be under this database tree.

Accessing Data from Remote Environments

In our applications we found that the remote env. is specified as follows:

- **As hard-coded parameter for the envName widget option**. To be Avoided
- **Derived by and environment variable** -e.g. telescope identifier, m1m3 LCU name, etc ..
- **The name is derived from a parameter in the database.** Telescope GUI interface uses this to get the name of the LCU associated to the current focus.

The aim of this example is to be able to define the name of a remote environment without having to re-edit the panel definition

The steps are :

- **The envName field in the configuration panel is filled with the name of a variable.**

- **The variable is initialized to define both the actual name of the environment and the associated CWP.**

```
proc xxxInitData {} {
   global remoteEnv env

 # Define name of the remote environment and associated CWP

   if {[info exists env(REMOTE_ENV)]} {
      set remoteEnv $env(REMOTE_ENV):panDataTest
   } else {
      set remoteEnv $env(RTAPENV):panDataTest
   }
}
```

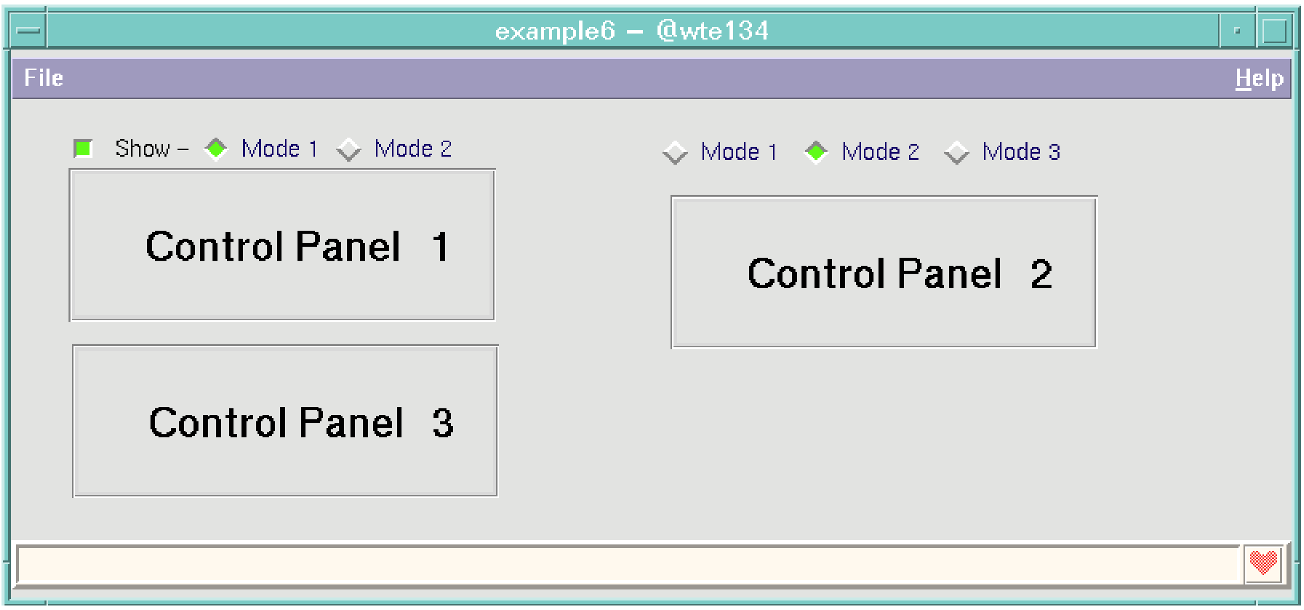
Special Widgets

Widgets classContainer and classFrame allow the user to switch between different uifClass widgets.

The interaction between these two widgets and the menu bar contribute to :

- **Save space in big panels.**

- **Create 'context sensitive' GUIs and hence be able to display what the user 'currently' needs.**



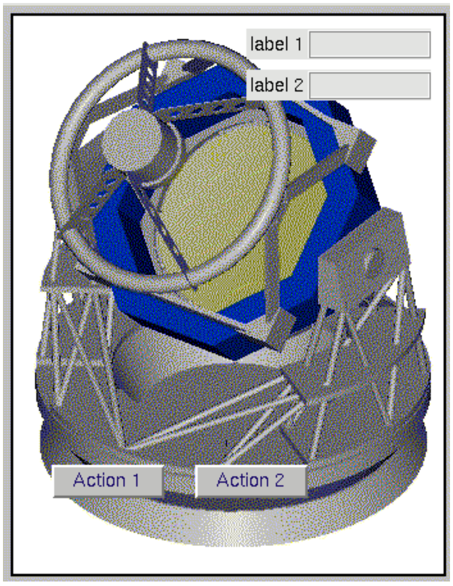
Graphics

The panel editor does not support an interactive creation and editing of graphical elements.

The user has to use tcl/tk statements to add simple graphics such rectangles, lines, circles etc ...

```
$cv create rectangle 200 300 400 350 -width 2 -outline blue
```

Is there any chance to design less boring control panels ?



Would this be enough ??

uifClass widgets can be used as a canvas where to create an image and lay on top some widgets.

Optimization

Panel

- Use polling to update VECTORS or TABLES.

- Use UifOutputArray to update VECTORS and TABLES.

- Use events for SCALAR data changing ‘slowly’ in time.

- Use the “on change” filter (!=) for events associated to SCALARS

Control Programs

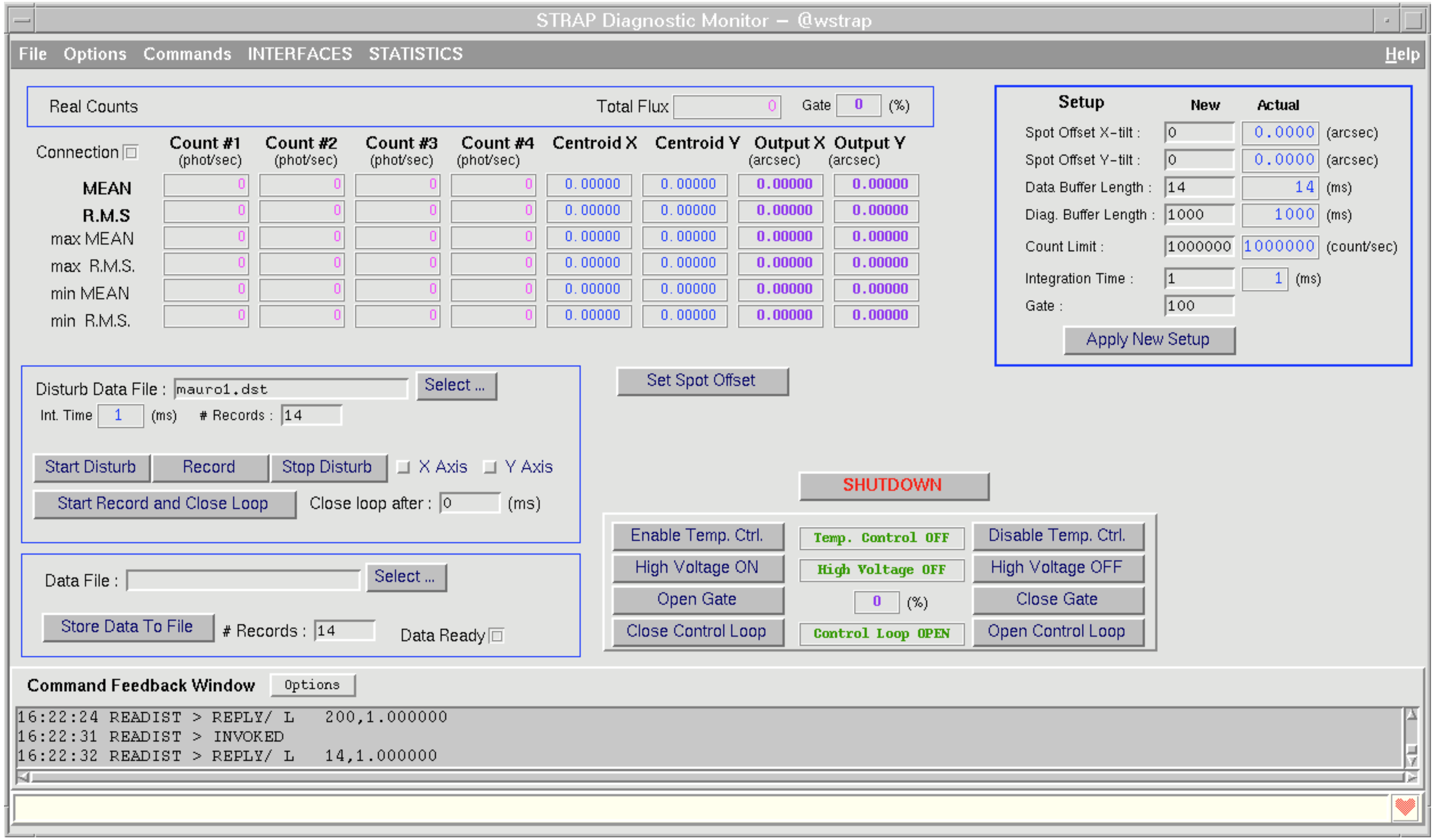
- Do not write VECTORS or TABLES element by element. Either write all data in one go or row/column wise

- Avoid to scan large TABLES

General Remarks

- Do not wait until the GUI is finished to test its functionality.
- Do not add "widgets" by hand, if you have special needs follow example in paragraph 5.3.12 of the manual.
- Use panDbRead to access DB values.
- If you use directly seq_xxx procedures, add also the code to handle error conditions.
- Sometimes it's difficult to import 'public domain' widgets because they require a given setup/configuration at the moment of the creation (rtd widgets) and the panel editor just creates an empty instance of the widget.
- Before implementing your own 'work around' to the problem raise the point, ask people and even better ...

Submit an SPR :-)


STRAP Diagnostic Monitor — @wstrap
File Options Commands INTERFACES STATISTICS
Help
Real Counts
Total Flux
Gate
(%)
Connection
Count #1 (phot/sec)
Count #2 (phot/sec)
Count #3 (phot/sec)
Count #4 (phot/sec)
Centroid X
Centroid Y
Output X (arcsec)
Output Y (arcsec)
MEAN
R.M.S
max MEAN
max R.M.S.
min MEAN
min R.M.S.
Setup
New
Actual
Spot Offset X-tilt :
Spot Offset Y-tilt :
Data Buffer Length :
Diag. Buffer Length :
Count Limit :
Integration Time :
Gate :
(arcsec)
(ms)
(count/sec)
Apply New Setup
Disturb Data File : mauro1.dst
Select ...
Int. Time 1 (ms) # Records : 14
Start Disturb
Record
Stop Disturb
X Axis
Y Axis
Start Record and Close Loop
Close loop after : 0 (ms)
Set Spot Offset
SHUTDOWN
Enable Temp. Ctrl.
Temp. Control OFF
Disable Temp. Ctrl.
High Voltage ON
High Voltage OFF
High Voltage OFF
Open Gate
Close Gate
Close Control Loop
Control Loop OPEN
Open Control Loop
Data File :
Select ...
Store Data To File
Records : 14
Data Ready
Command Feedback Window
Options
16:22:24 READIST > REPLY/ L 200,1.000000
16:22:31 READIST > INVOKED
16:22:32 READIST > REPLY/ L 14,1.000000