



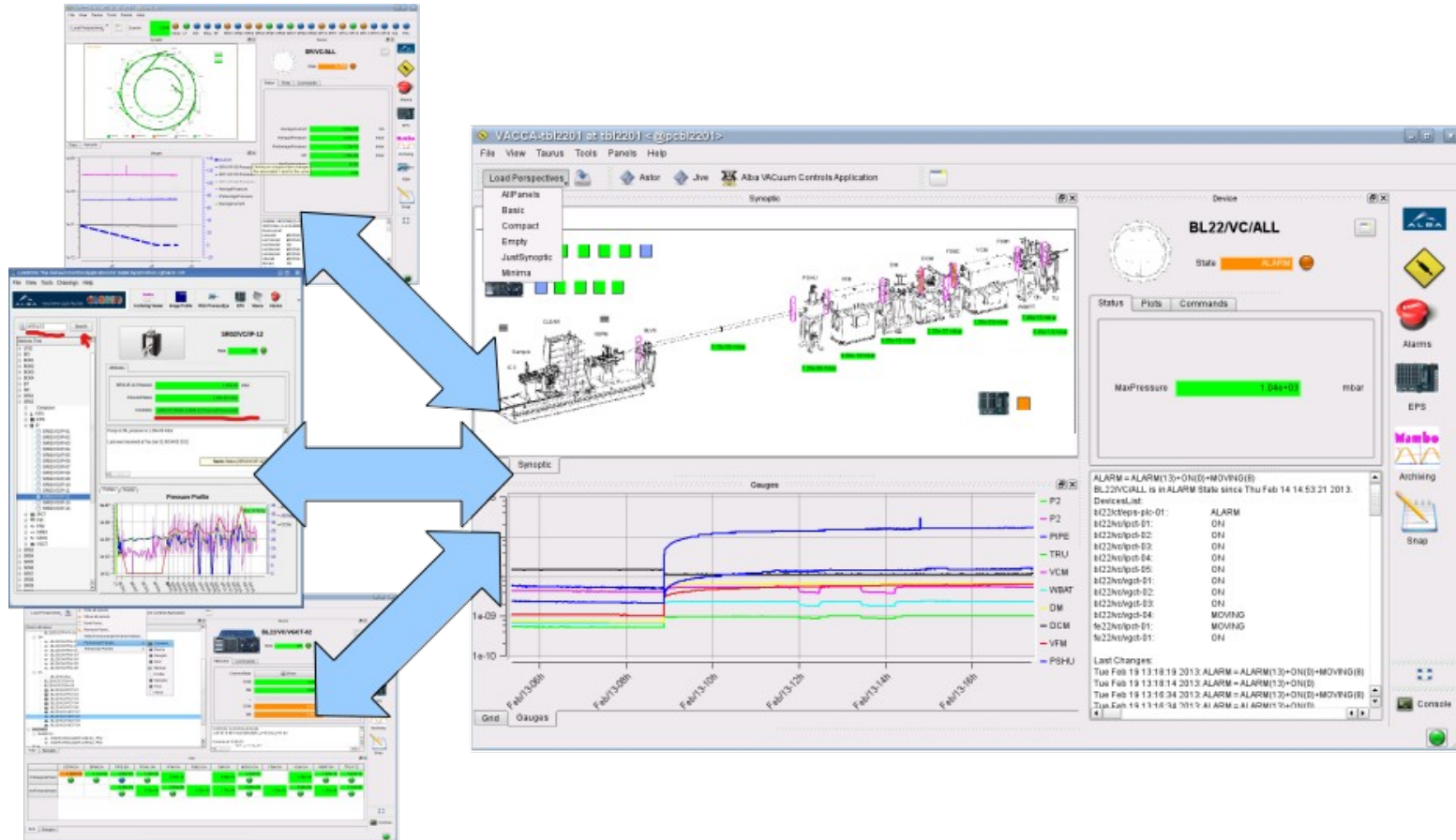
VACC4 (+ PANIC)

**Viewer and Commander Control Application
An SCADA experience for Tango**

Sergi Rubio Manrique, ALBA Synchrotron

Introduction
Usage
Components and API's
Performance
Panic
Future

A framework for building highly customizable applications, build on top of a Taurus GUI.



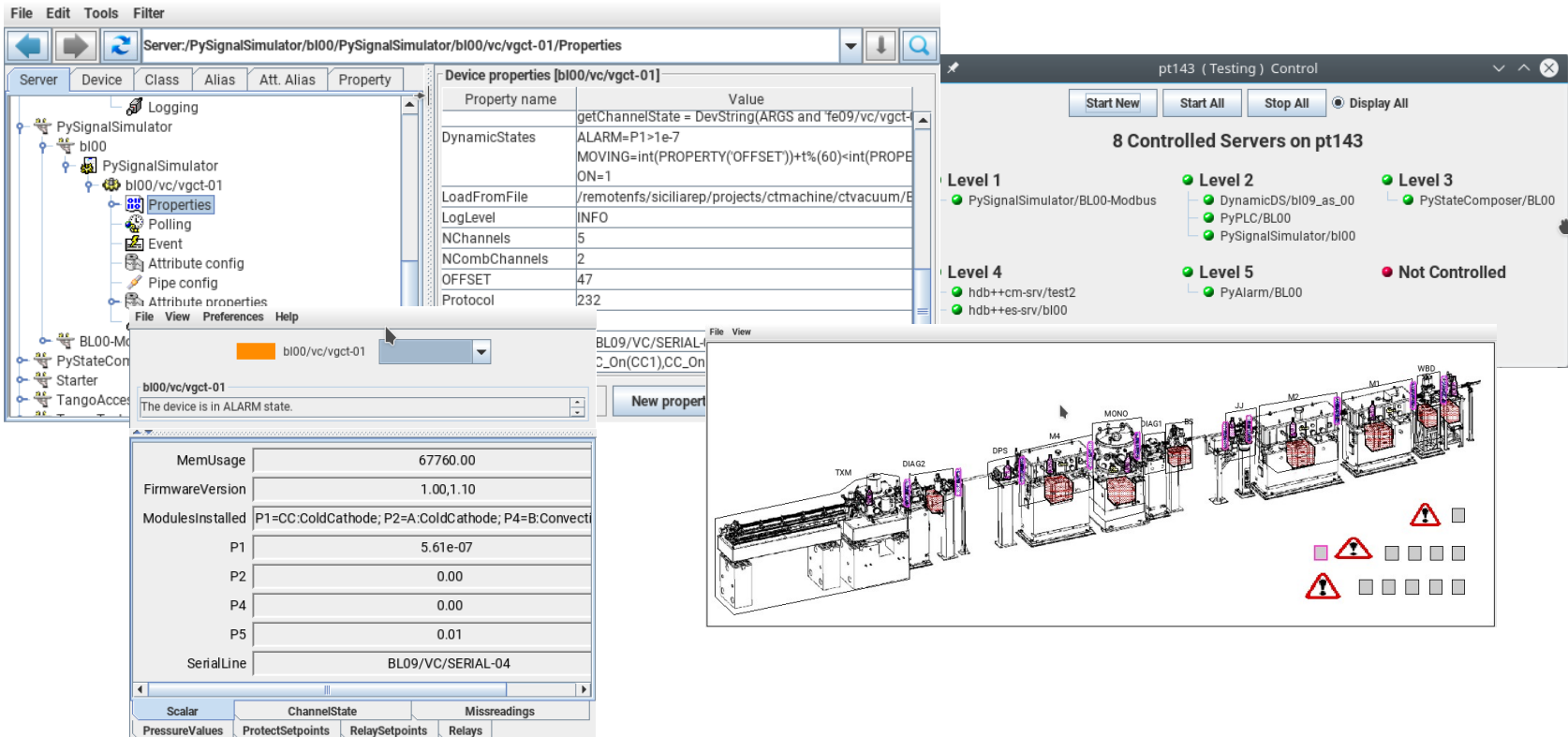
Taurus GUI provides perspectives, user settings and a framework to design new GUI's "online"

Why VACCA?

- An optimized navigator, to browse devices through large control systems.
- Full featured, to provide all tools needed by the **final user** (scientist or non-software engineer).
- Tango and Linux only, but accepting models that do not exist in Taurus 3 as hosts and properties.
- Customizable, to suit user needs and **hide control system complexity**. (Jive, Astor and devices/attributes the user doesn't need).

Tango Control System

Jive, Astor, Mambo, Jdraw, ATKPanel, DeviceTree



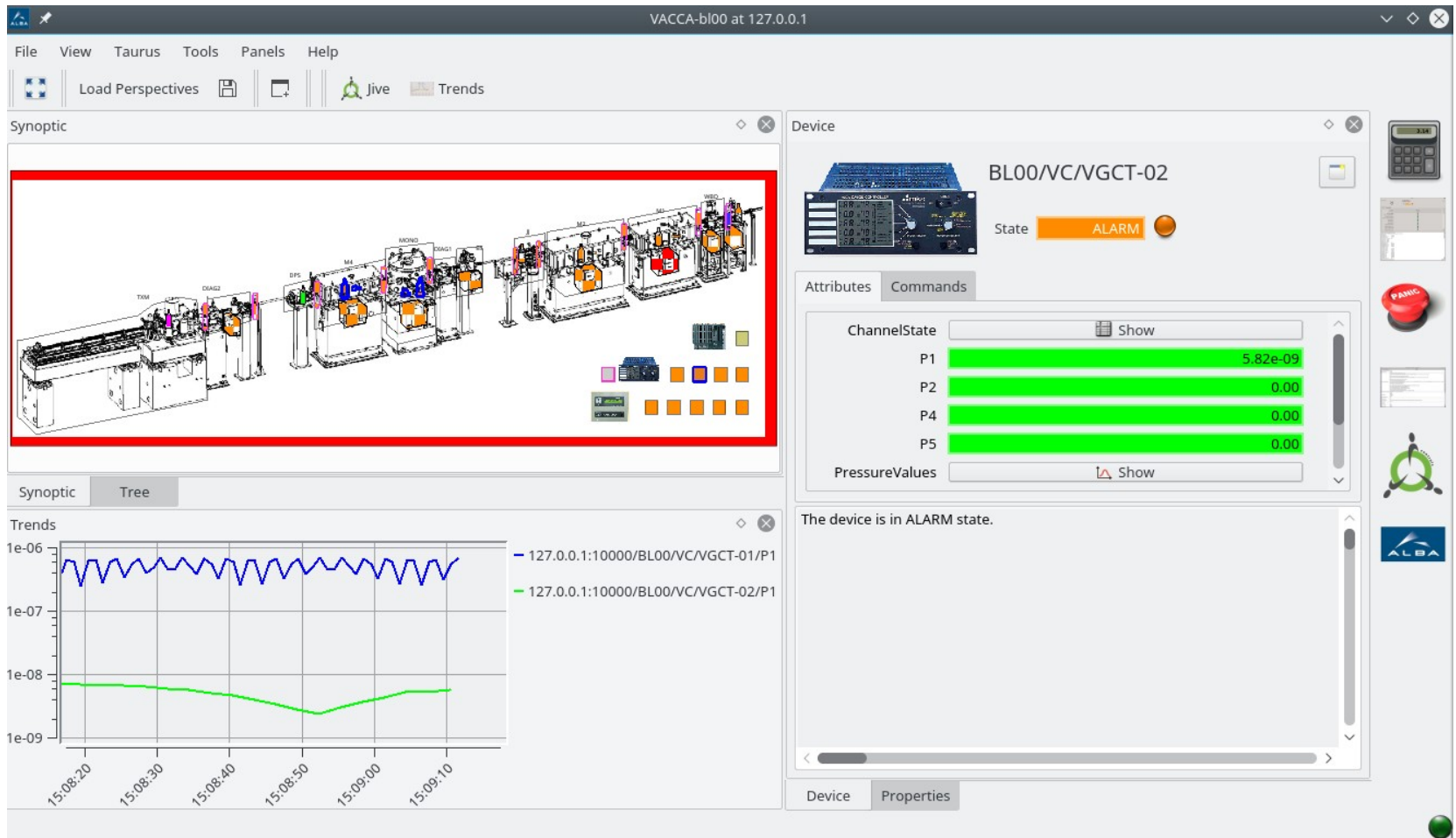
The screenshot displays three overlapping windows from the Tango Control System interface:

- Device properties [bl00/vc/vgct-01]:** A window showing the configuration for a specific device. It includes a tree view on the left with nodes like Logging, PySignalSimulator, and Properties. The main pane shows a table of properties:

Property name	Value
DynamicStates	getChannelState = DevString(ARGS and 'fe09/vc/vgct-01')
LoadFromFile	/remotenfs/siciliarep/projects/ctmachine/ctvacuum/E
LogLevel	INFO
NChannels	5
NCombChannels	2
OFFSET	47
Protocol	232
- pt143 (Testing) Control:** A window showing the status of 8 controlled servers on pt143. It includes buttons for Start New, Start All, Stop All, and Display All. The servers are organized into levels:
 - Level 1: PySignalSimulator/BL00-Modbus
 - Level 2: DynamicDS/bl09_as_00, PyPLC/BL00, PySignalSimulator/bl00
 - Level 3: PyStateComposer/BL00
 - Level 4: hdb++cm-srv/test2, hdb++es-srv/bl00
 - Level 5: PyAlarm/BL00
 - Not Controlled: (indicated by a red dot)
- BL09/VC/SERIAL-4:** A window showing the status of a specific device. It includes a table of properties:

Property name	Value
MemUsage	67760.00
FirmwareVersion	1.00.1.10
ModulesInstalled	P1=CC:ColdCathode; P2=A:ColdCathode; P4=B:Connect
P1	5.61e-07
P2	0.00
P4	0.00
P5	0.01
SerialLine	BL09/VC/SERIAL-04

+ TaurusGUI, TaurusTrend, Panic



Select, Config Properties, Attributes, Plot Archiving, Start/Stop, Show Alarms

VACC4



Connecting things together

Py
Tango



fandango



Package for Alarms
and Notification of Incidences
from Controls

VACCA, Tango and Taurus



VACCA does not "reinvent" the wheel, but extends Taurus with specific widgets for Tango services:

- Database and properties
- Host management (Starters)
- Alarms
- Archiving (Archivers, Managers, Extractors)

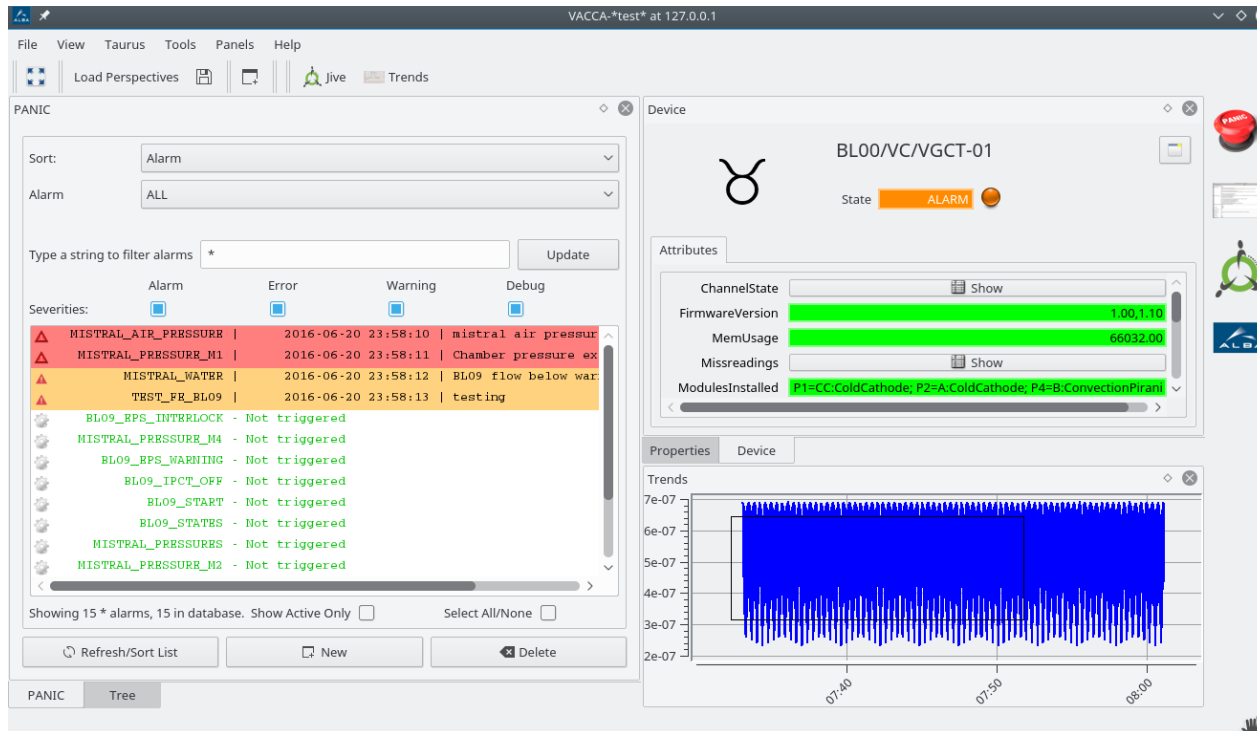
VACCA 1.0 (java, 2007) : (JL. Pons, F.Poncet)

VACCA 2.0 (python, 2009): (A.Wolowicz, R.Suñé, F.Becheri, M.Guijarro, T.Coutinho)

VACCA 3.0 (taurus, 2012) : ESRF / Max IV (C.Falcón , A.Milán, J.Forsberg, C.Pascual)

VACCA 4.0 (library, 2015): TangoDB, Panic, SuSE packages (C.Falcón, D.Roldán, A.Götz)

Developed for the Tango Workshop at ICALEPCS 2015 (WEPGF148), aimed to provide a single tool to manage the whole Tango Control System.



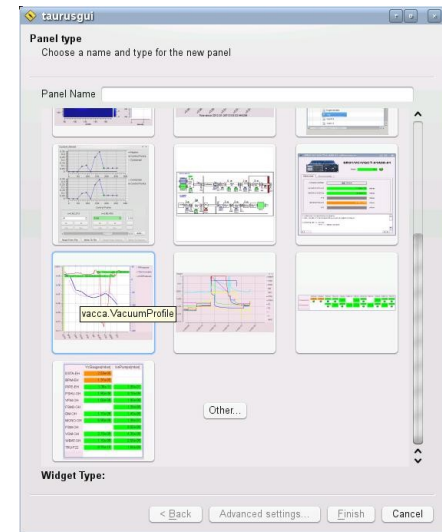
It integrates Tango Database (properties) and Alarms (**PANIC**) with the already existing widgets.

EPICS Control System Studio is an Eclipse-based collection of tools to monitor and operate large Epics Control Systems.

VACCA is both a control application and a Taurus widget library build to supervise and control large (and small) Tango Control Systems.

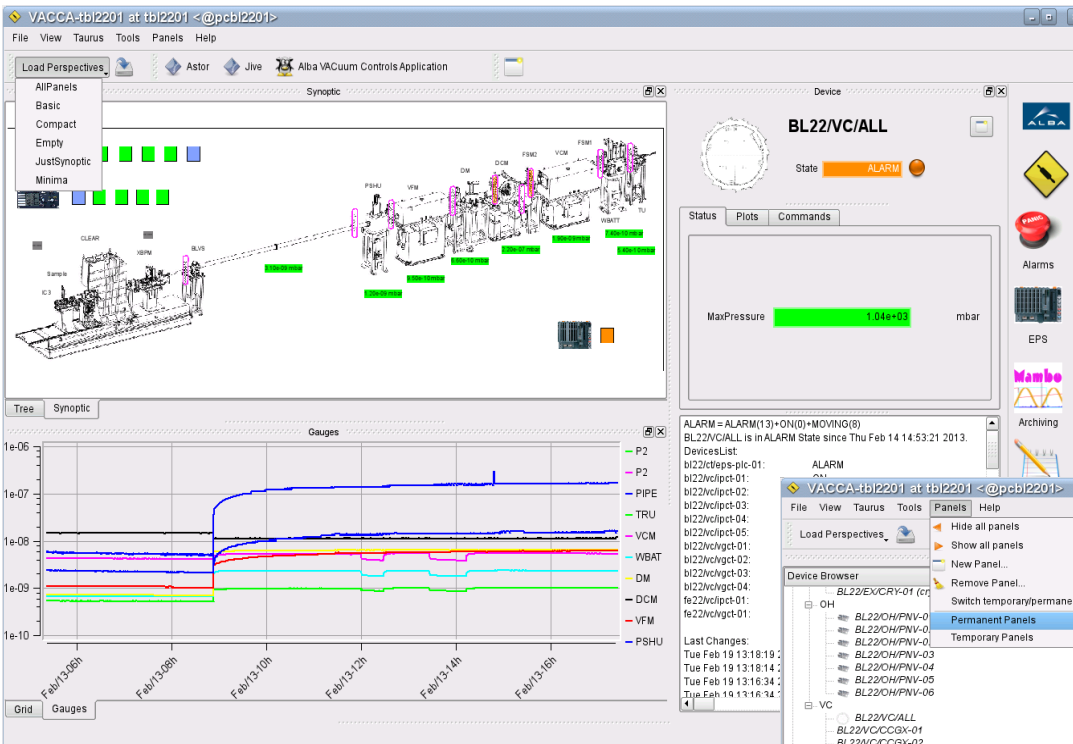
Both provide synoptic-based navigation, data browser/finder tools, archiving access and alarm visualization.

- Vaccagui → TaurusGUI
- VaccaPanel → TaurusDevicePanel
- VaccaTree → TaurusDevTree
- VaccaTrend → TaurusTrend
- VaccaApplication → TaurusAction
- VaccaAlarms → Panic.AlarmGUI
- ArchivingBrowser → PyTangoArchiving
- VaccaSynoptic → maxiv.SVGSynoptic /
TaurusJDrawSynopticsView

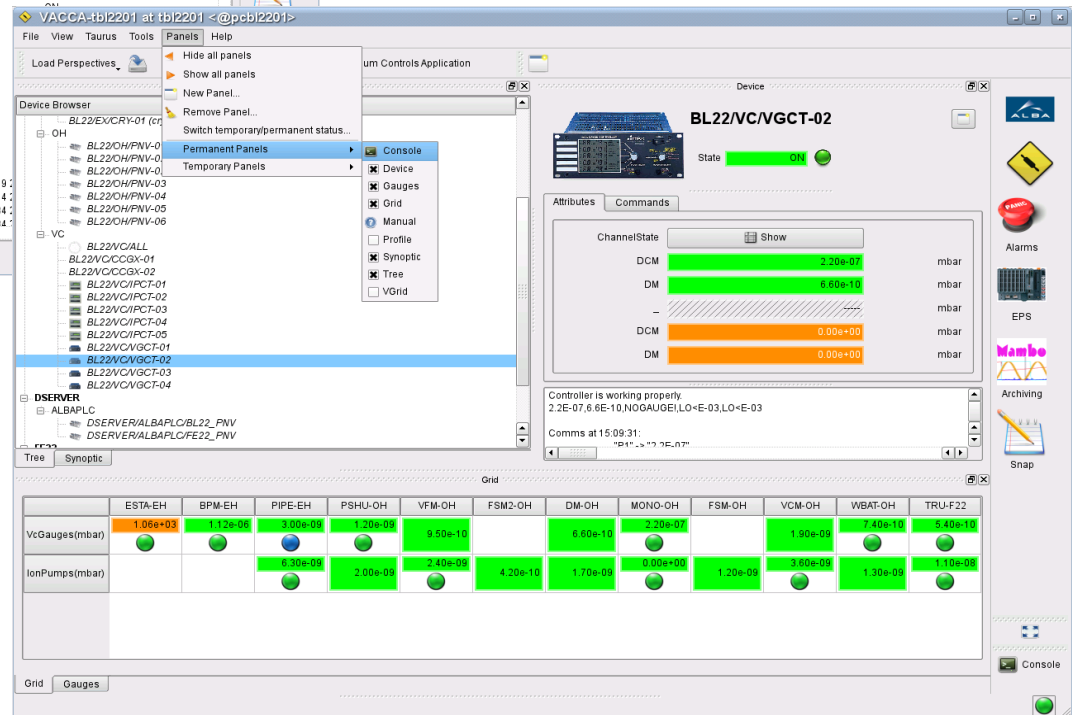


All of them resizable, with enhanced signals, icons for toolbars and drag&drop.

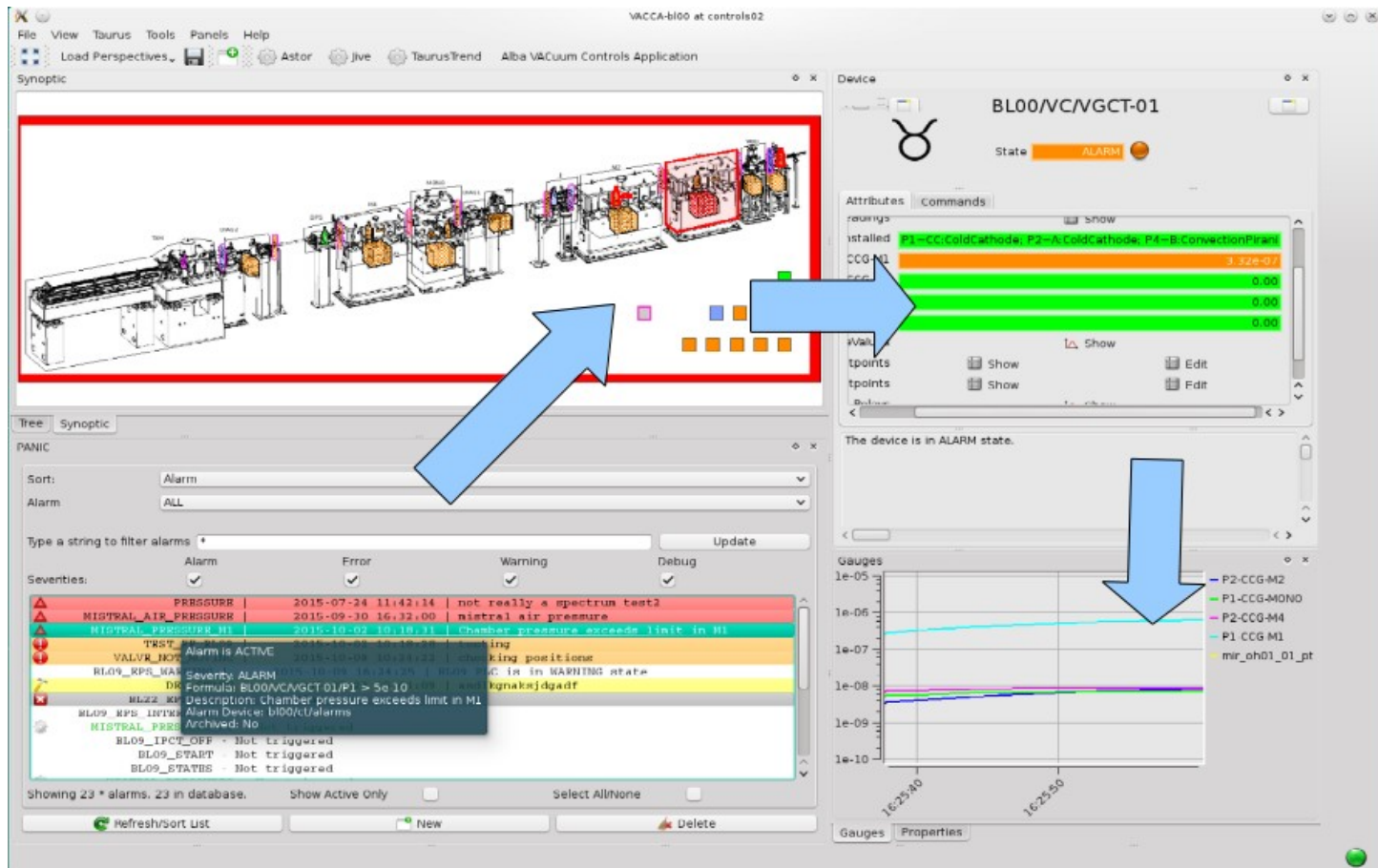
Taurus provides perspectives and menus to customize the application for each end user



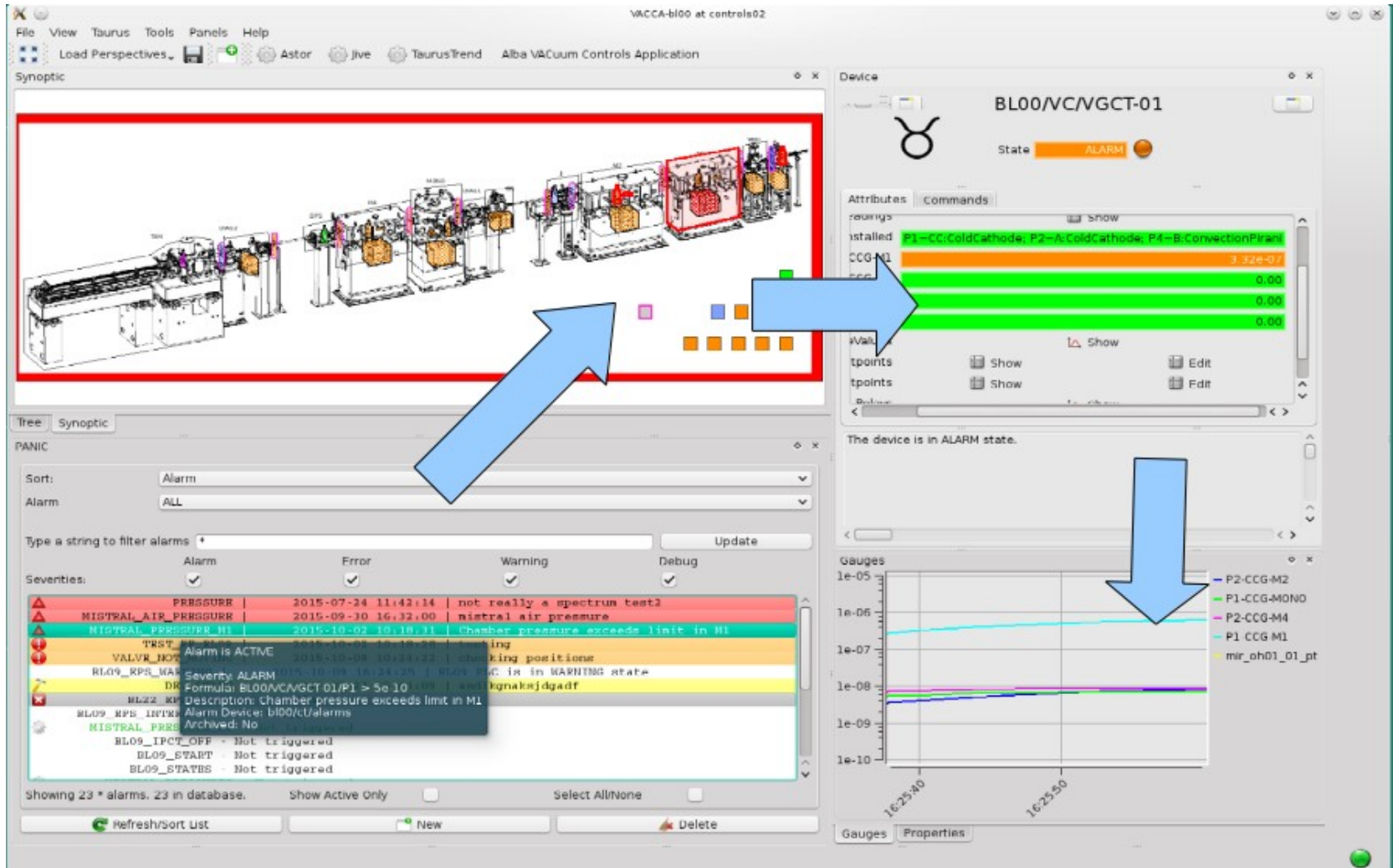
VACCA core is a Taurus GUI with a default set of panel widgets, providing navigation of a control system based on shared signals and drag&drop.

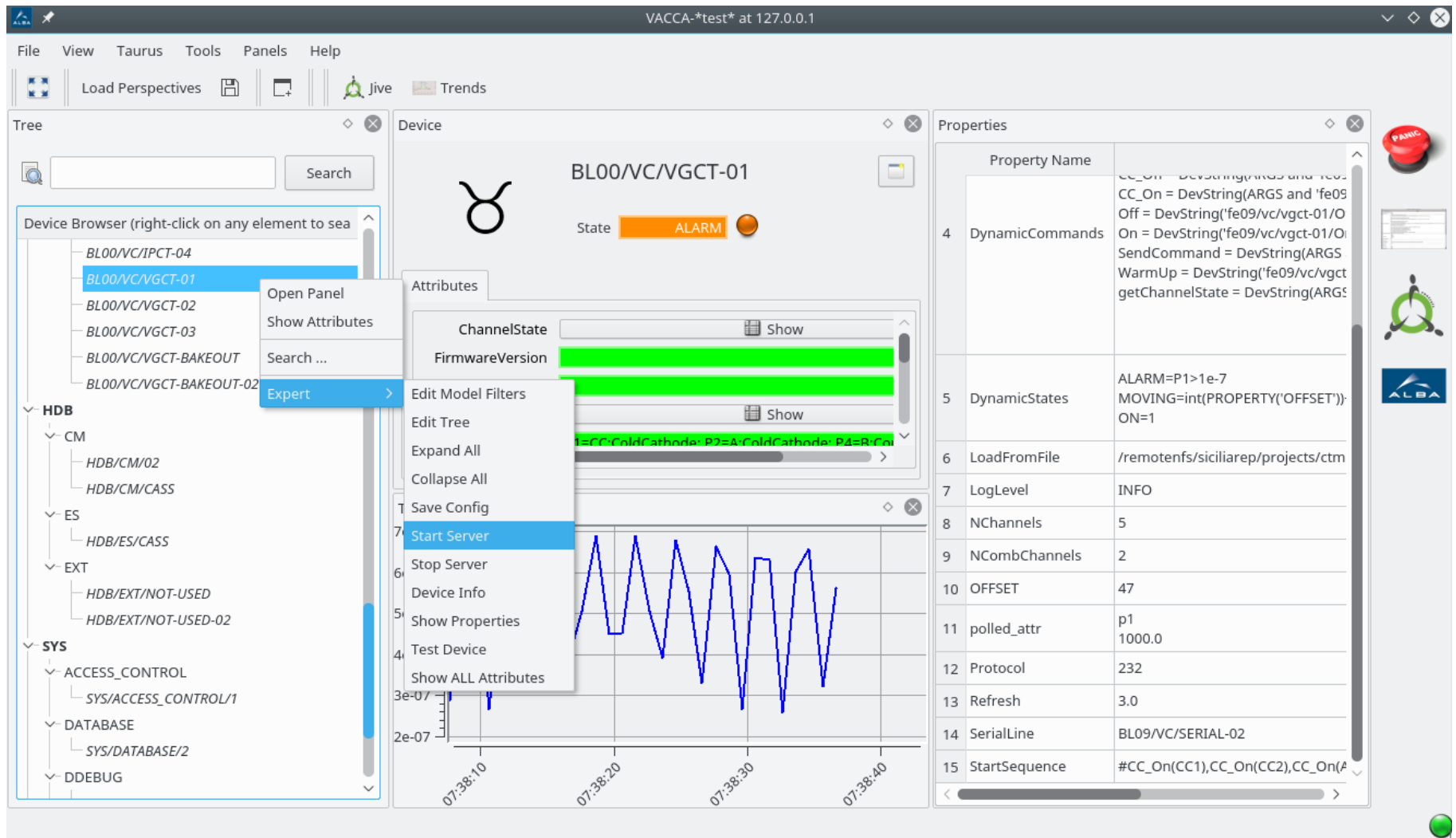


Widgets from the VACCA library are subscribed by default to Taurus Shared Data Manager signals.



All widgets interact between them, by model selection or drag & drop.





The screenshot displays the VaccaTree application window titled "VACCA-*test*" at 127.0.0.1. The interface includes a menu bar (File, View, Taurus, Tools, Panels, Help), a toolbar with icons for Load Perspectives, Save, and Jive/Trends, and a main workspace divided into three panels.

Tree Panel: A hierarchical tree view on the left. The "Device Browser" section is active, showing a list of devices. A right-click context menu is open over "BL00/VC/VGCT-01", with options including "Open Panel", "Show Attributes", "Search...", "Expert", "Edit Model Filters", "Edit Tree", "Expand All", "Collapse All", "Save Config", "Start Server", "Stop Server", "Device Info", "Show Properties", "Test Device", and "Show ALL Attributes".

Device Panel: The central panel displays the selected device "BL00/VC/VGCT-01" with a "State" indicator set to "ALARM". Below this, a "ChannelState" section shows a green bar, and a "FirmwareVersion" section shows a green bar. A graph at the bottom shows a blue waveform over time, with labels for 07:38:10, 07:38:20, 07:38:30, and 07:38:40.

Properties Panel: The right panel shows a table of properties for the selected device.

Property Name	Value
4 DynamicCommands	CC_On = DevString(ARGS and 'fe09 Off = DevString('fe09/vc/vgct-01/O On = DevString('fe09/vc/vgct-01/O SendCommand = DevString(ARGS WarmUp = DevString('fe09/vc/vgct getChannelState = DevString(ARG
5 DynamicStates	ALARM=P1>1e-7 MOVING=int(PROPERTY('OFFSET')) ON=1
6 LoadFromFile	/remotenfs/siciliarep/projects/ctm
7 LogLevel	INFO
8 NChannels	5
9 NCombChannels	2
10 OFFSET	47
11 polled_attr	p1 1000.0
12 Protocol	232
13 Refresh	3.0
14 SerialLine	BL09/VC/SERIAL-02
15 StartSequence	#CC_On(CC1),CC_On(CC2),CC_On(A

A googling tool for Tango Database

It
d

Type a part of device name and a part of attribute name, use "*" or " " as wildcards:

Filters

Options




Device or Alias: bl00 vgct

Attribute: p*

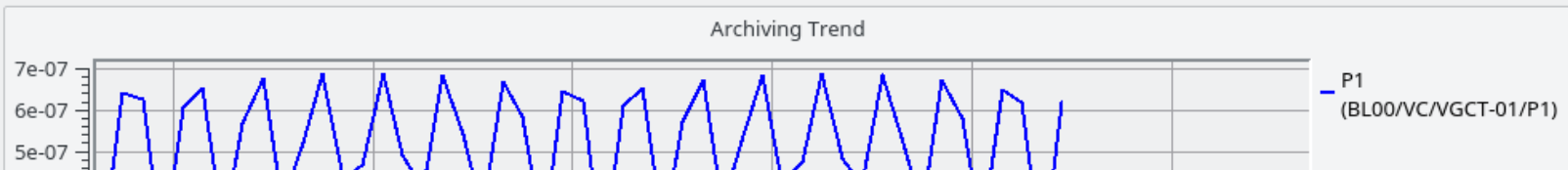
Update

☐ Show archived attributes only

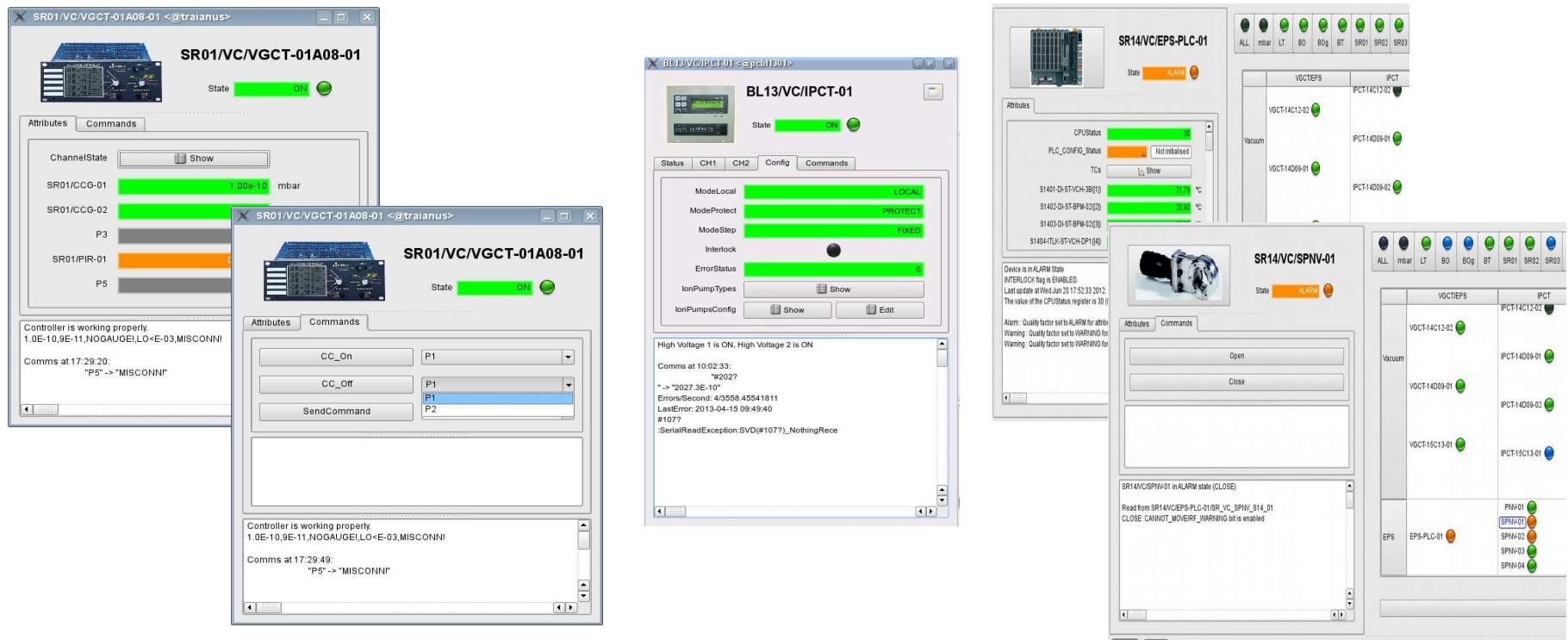
Enter Device and Attribute filters using wildcards (e.g. li/ct/plc[0-9]+ / ^stat*\$ & !status) and push Update

Label/Value	Device	Attribute	Alias	Archiving
P1  6.18e-07	BL00/VC/VGCT-01	P1		HDB
P2  0.00	BL00/VC/VGCT-01	P2		HDB
P4  0.00	BL00/VC/VGCT-01	P4		HDB

Drag any attribute from the first column into the trend or any taurus widget you want:

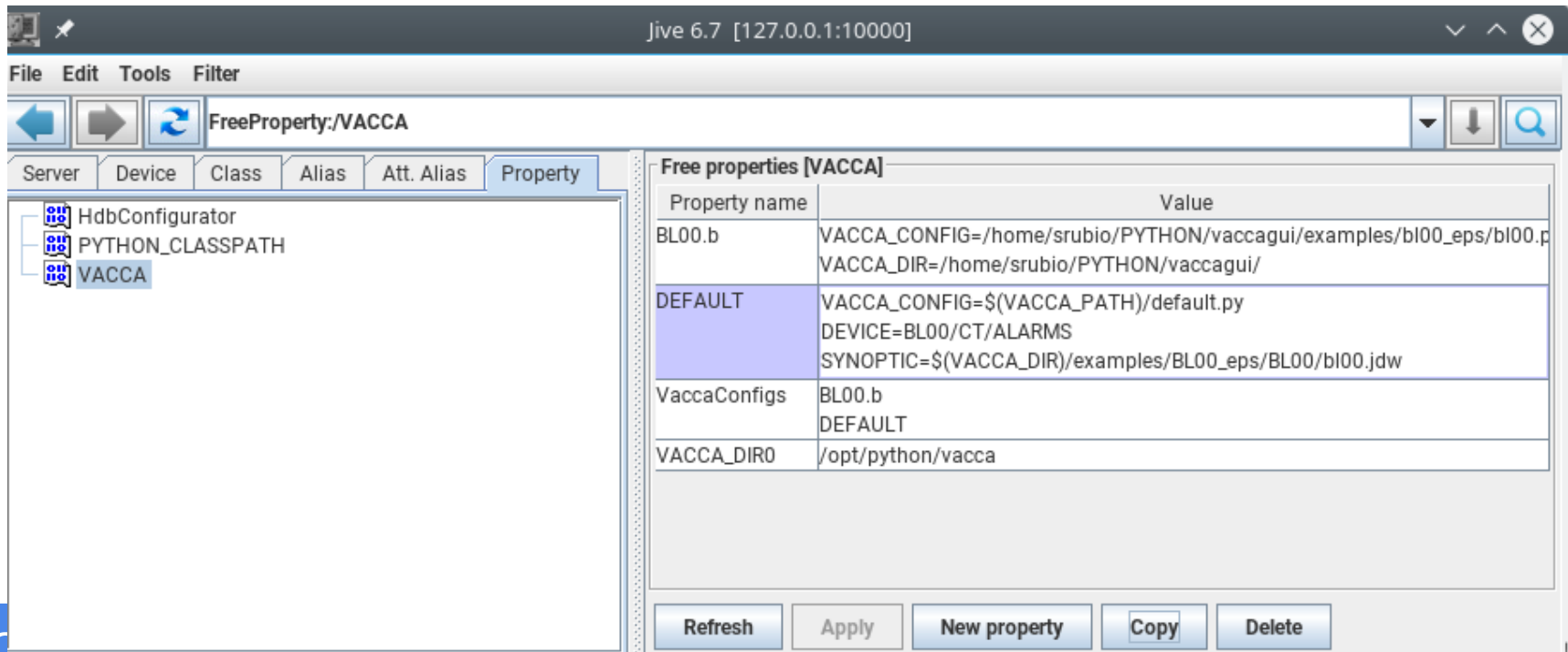


Forms and tabs customizable for each class by defining Command/Attribute/Icon/Property filters.



Since VACCA 3.0; a [config].py file has been used to configure vacca by defining models variables: DEVICE, TREE, SYNOPTIC, TREND, APPS, PANELS, ...

In Vacca 4.2 new definition style has been added using Tango Free Properties, that can be override by environment:



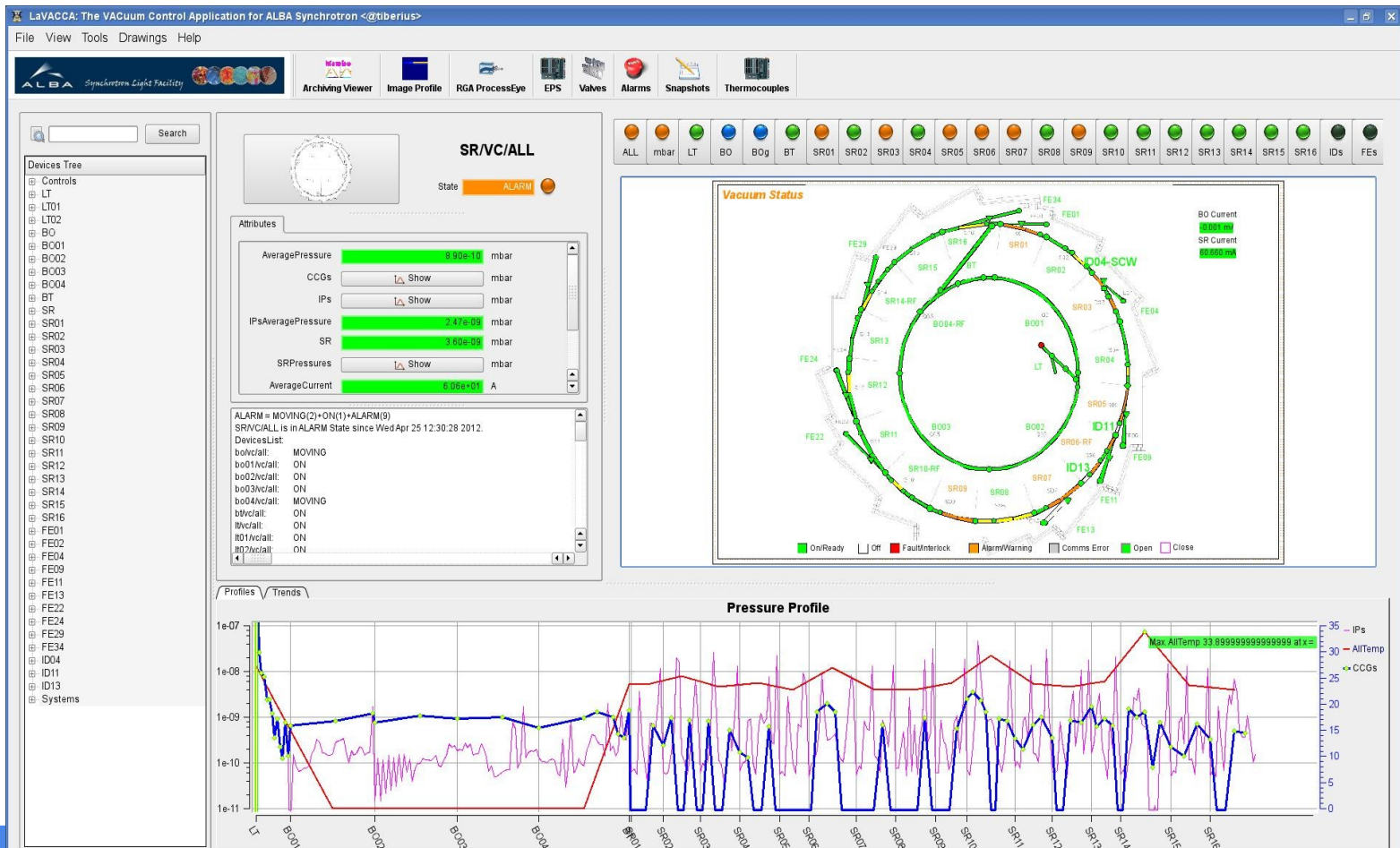
FreeProperty:/VACCA

Property name	Value
BL00.b	VACCA_CONFIG=/home/srubio/PYTHON/vaccagui/examples/bl00_eps/bl00.p VACCA_DIR=/home/srubio/PYTHON/vaccagui/
DEFAULT	VACCA_CONFIG=\$(VACCA_PATH)/default.py DEVICE=BL00/CT/ALARMS SYNOPTIC=\$(VACCA_DIR)/examples/BL00_eps/BL00/bl00.jdw
VaccaConfigs	BL00.b DEFAULT
VACCA_DIR0	/opt/python/vacca

Refresh Apply New property Copy Delete

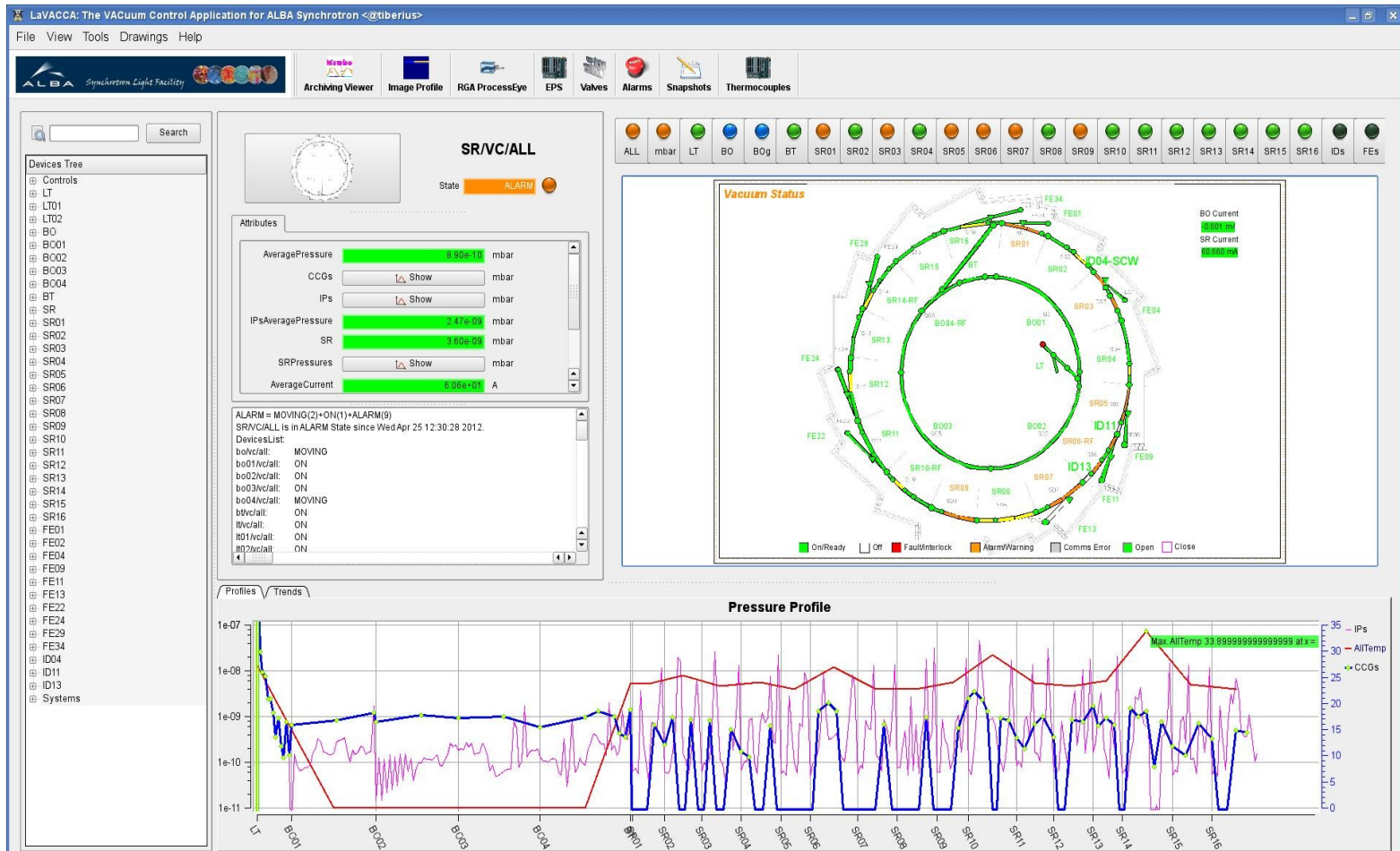
Performance

GUI's opened against large numbers of devices (100-500) have some common issues: high cpu usage, lack of refresh to events, timeouts, huge loading time , ...



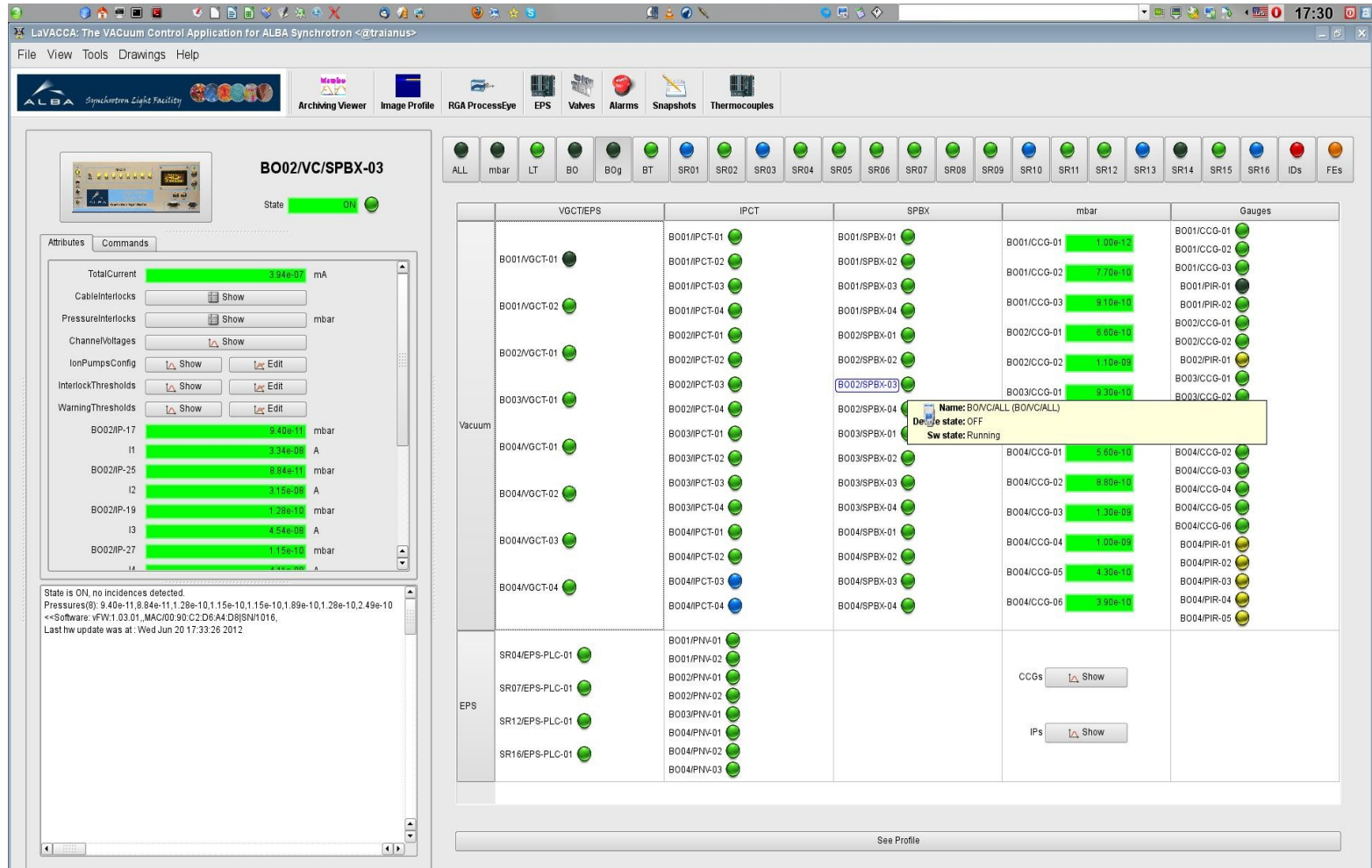
Performance

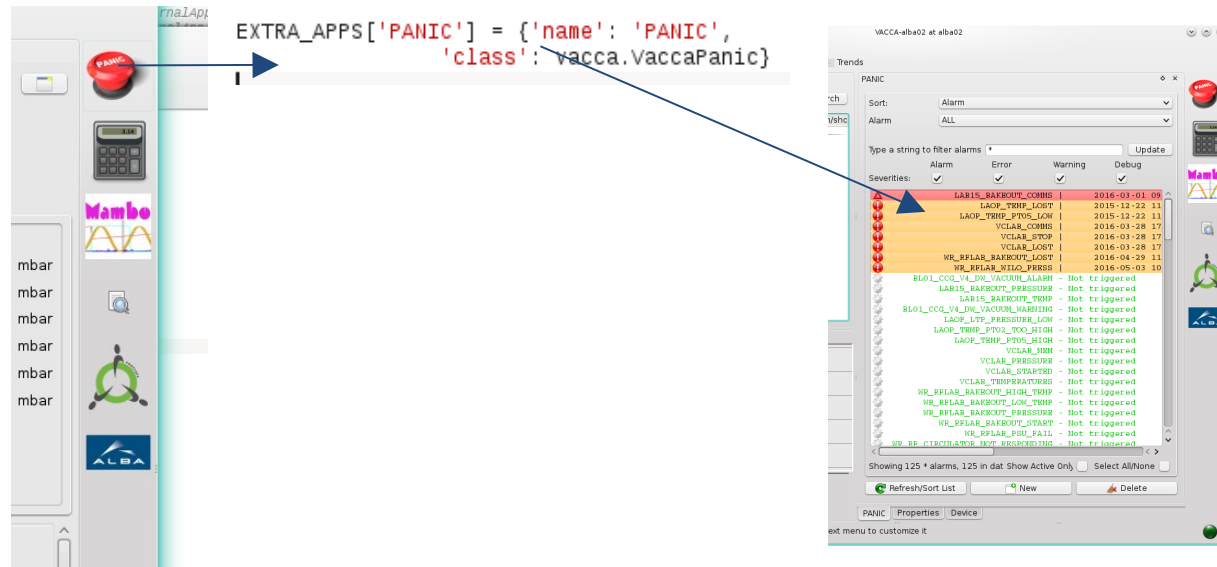
We used PyStateComposer and PyAttributeProcessor to summarize the status and key attributes of the system.



Performance

Just composers are connected at startup, delaying other models until their widget is shown.



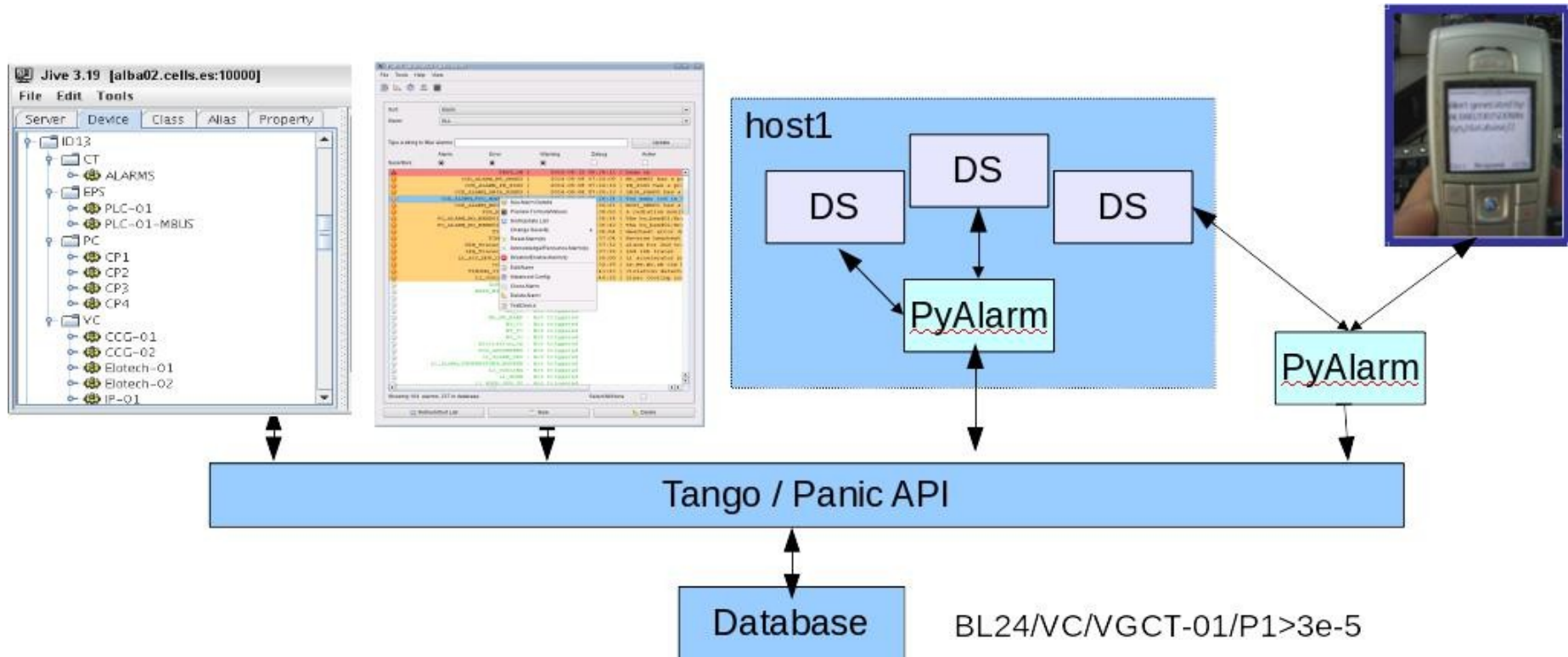


Widgets declared as panels are instantiated at startup thus increasing loading time and overall cpu usage.

Extra applications are instead just added to the right-side bar, they can be opened on-demand and integrated within the gui (drag and drop, saved to perspectives).

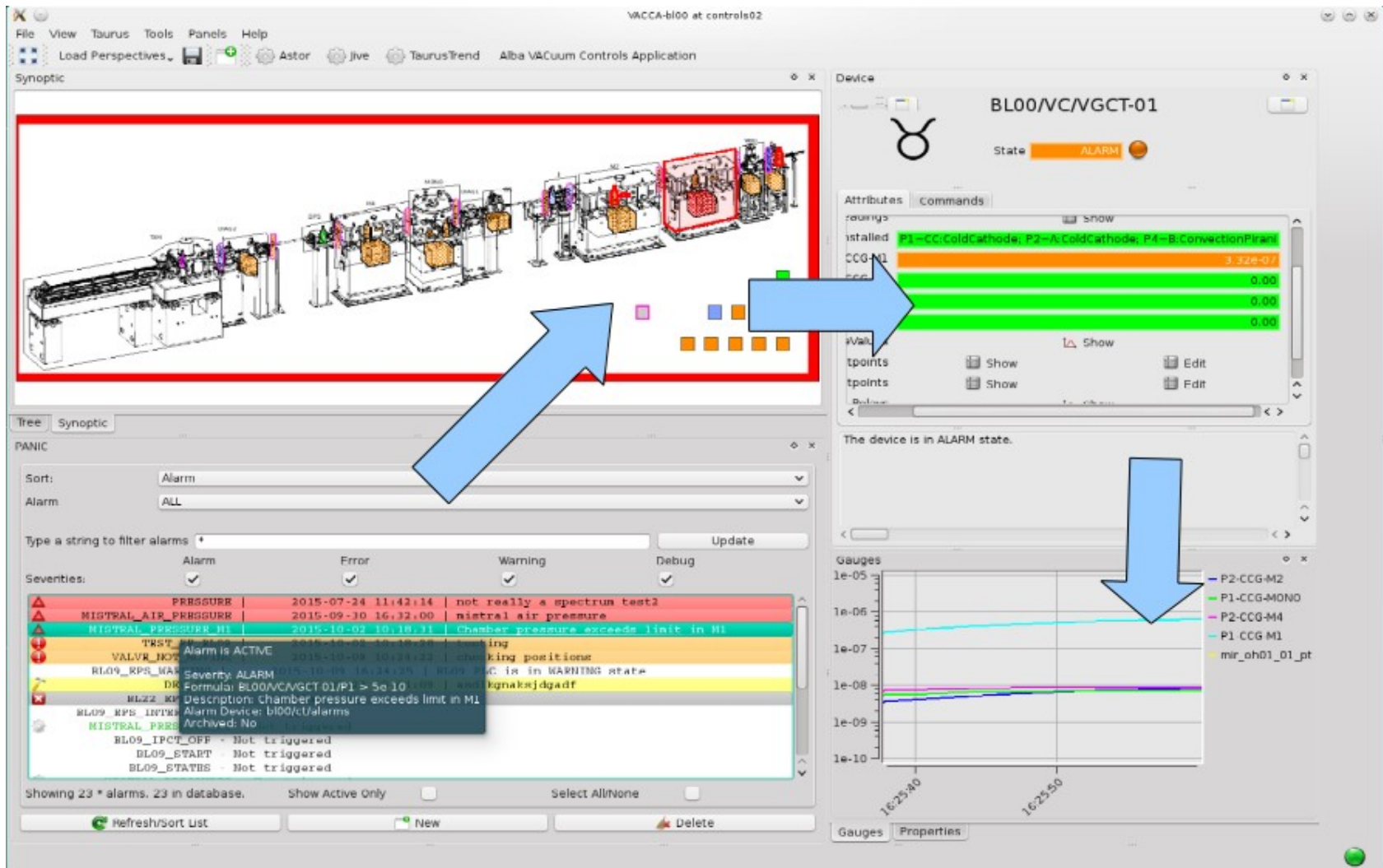
This allows to have this complex (and heavy) tools in the gui without compromising the startup or stability.

In use at ALBA (2010), Max IV, SKA
Last release presented at PCAPAC'14



Summarizing:

- It triggers alarms on evaluation of python formulas.
- Alarms may include **wildcards, values, qualities, delta changes, time of update, ...**
- Provides >20 parameters for tuning evaluation and notification behavior.
- Alarm database centralized (TangoDB/Snap), Alarm evaluation distributed (PyAlarm).
- Alarms trigger notification by **email, SMS, Voice or any Tango command** available.
- Used for automated control (FE) or archiving (SNAP).



PANIC Setup

Name: CIRCE_PRESSURE OK Acknowledge

Device: bl24-circe/ct/alarms

Description: Chamber pressure exceeds limit

Receivers: %VACMV,%CTRLMV,%VIRGINIA,%CTRL2

tb12401:10000/BL24/VC/VGCT-01/P1>3e-5 OR tb12401:10000/BL24/VC/VGCT-01/P2>3e-5

tb12401:10000/BL24/VC/VGCT-01/P1 > 3e-5

OR



tb12401:10000/BL24/VC/VGCT-01/P2 > 3e-5

Add Expression

Raw Edit

Add Relation

Clear

Edit

Save

Cancel

Close

PyAlarm Device Configuration		
bl00/ct/alarms		
	Attribute Name	Attribute Value
1	AlarmThreshold	3
2	AlertOnRecovery	no
3	AutoReset	3600
4	CreateNewContexts	False
5	Enabled	120
6	EvalTimeout	500
7	FlagFile	/tmp/alarm_ds.nagios
8	FromAddress	oncall
9	HtmlFolder	htmlreports
10	IgnoreExceptions	True
11	LogFile	/tmp/alarm_bl09_MISTRAL-CT-Alarms.log
12	LogLevel	INFO
13	MaxMessagesPerAlarm	20
14	PollingPeriod	5
15	Reminder	0
16	RethrowAttribute	False
17	RethrowState	True
18	SMSConfig	controls@cells:cells.cells
19	StartupDelay	0
20	UseProcess	False
Refresh		

PANIC Status

Last major release of Panic GUI on 2014

But, Panic core (API+PyAlarm) in constant evolution since then.

Most relevant changes are optimization on alarm evaluation and introducing multi-host alarms (SKA).

TCS team doing a lot of tests over Panic and debugging the tool intensively (> 1350 attrs/alarm).

On PANIC systems with > 100 alarms three different issues appear:

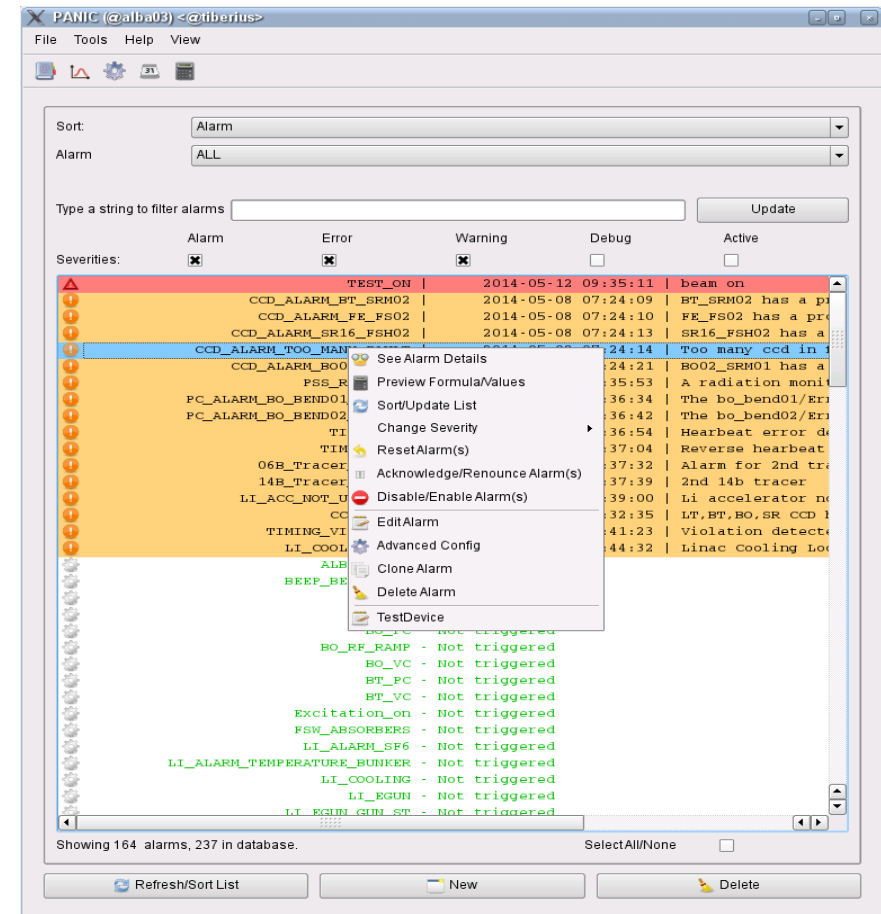
- Performance, slow startup, slow refresh (notifications not affected).
- Hard to discriminate alarms for different interest groups (search bar is not enough)
- Applying multiple configs to many alarms resulting on timeout.

To solve that a new device class will be added to the PANIC packages:

PanicContext.

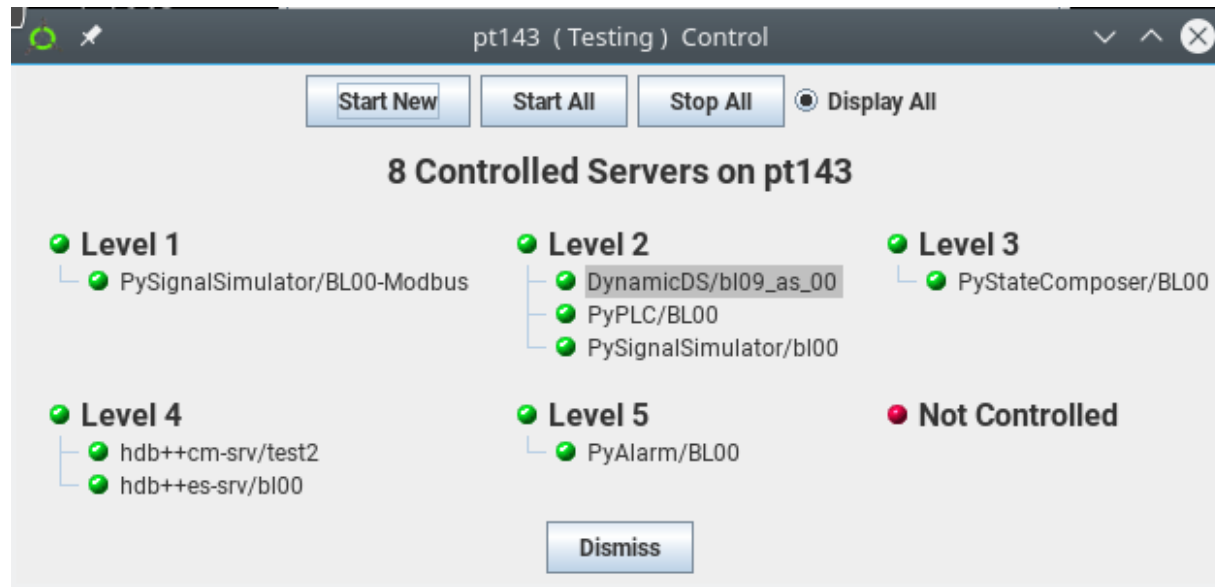
The new DS will group alarms by area-of-interest; caching and ordering Alarm states.

Current PyAlarm behavior won't change and PanicContexts won't be exclusive.



Once available, the GUI could be open against the whole system or just a set of contexts, subscribing only to the contexts attributes instead of the individual alarms.

Fandango, PySignalSimulator, PyStateComposer used to emulate Tango control systems.



Real devices exported to pickle or .json files and then played by simulators.

It allowed to validate VACCA and Panic for new installations.

Both projects have two types of documentation, API (sphinx) and Recipes (tango webpage or internal wikis) that needs to be updated.

VACC4 core suffered many changes in the last months and backwards compatibility may vary, contact me if you have issues updating.

HDB++ integration within Vacca/Taurus is still under development, as arrays are not yet supported nor configuration from GUI.

All PANIC packages are going to be merged in one and moved to GitHub (panic library, PyAlarm DS, Panic-ui (as panic, panic.ds , panic.gui)).

New PanicContextDS will require changes at API level and GUI, a fully functional release should be expected for january 2017.

Integration with Elettra Alarm database and multi-host snaps (SKA) to be evaluated against Kibana/ElasticSearch implementation done by MaxIV.

Thanks for your attention

