Dynamic Control Systems

Advantages, Challenges, Strategies

MOBPP05

Sergi Rubio Manrique, ALBA Synchrotron

Topics

- Dynamic Control Systems
- Dynamic Devices and UI's
- Users as Developers
- Scaling
- Tools and strategies

Dynamic Control Systems

dynamic adjective
dy·nam·ic | \ dī-ˈna-mik \
Definition of *dynamic* (Entry 1 of 2)

1a: marked by usually continuous and productive activity or change
a *dynamic* city
b: ENERGETIC, FORCEFUL
a *dynamic* personality

2 *or less commonly* **dynamical** \ dī-ˈna-mi-kəl \
a: of or relating to physical force or energy
b: of or relating to dynamics (see DYNAMICS)

3 *of random-access memory* : requiring periodic table refreshment of charge in order to retain data

https://www.merriam-webster.com/dictionary/dynamic

Dynamic Control Systems

ALBA is a Synchrotron in its 8th year of operation, with 8+3 beamlines.

Current changes and upgrades in our control system are:

Dynamic Control Systems

ALBA is a Synchrotron in its 8th year of operation, with 8+3 beamlines.
Current changes and upgrades in our control system are:

- Adding or removing elements from the machine:
  - Adding new Insertion Devices / removing gauges or ion pumps

Dynamic Control Systems

ALBA is a Synchrotron in its 8th year of operation, with 8+3 beamlines.
Current changes and upgrades in our control system are:

- Adding or removing elements from the machine:
    - Adding new Insertion Devices / removing gauges or ion pumps
- Expanding an existing installation:
    - Adding a new beamline or an experimental station in an existing beamline

Dynamic Control Systems

ALBA is a Synchrotron in its 8th year of operation, with 8+3 beamlines.
Current changes and upgrades in our control system are:

- Adding or removing elements from the machine:
    - Adding new Insertion Devices / removing gauges or ion pumps
- Expanding an existing installation:
    - Adding a new beamline or an experimental station in an existing beamline
- Controlling an existing system in a completely new way:
    - Replace RF control by FPGAs / RF IOT plants by Solid State Amps

Dynamic Control Systems

How we deal with these changes in the Control System:

- Once new Hardware is available for testing, it's introduced in our Cabling and Equipment databases.
- Control Engineers develop TANGO Devices to access the new hardware.
- If needed, High level devices are adapted to User needs, using Python Dynamic devices, like Processor, Composer or Façade Devices.
- Old user interfaces are modified by Control Engineers, while new UI's are developed by users using the Taurus GUI framework.

Tools: Python

Main programming language in TANGO community

Provided to scientists to replace matlab UI's and scripts.

Dynamic Typed Language

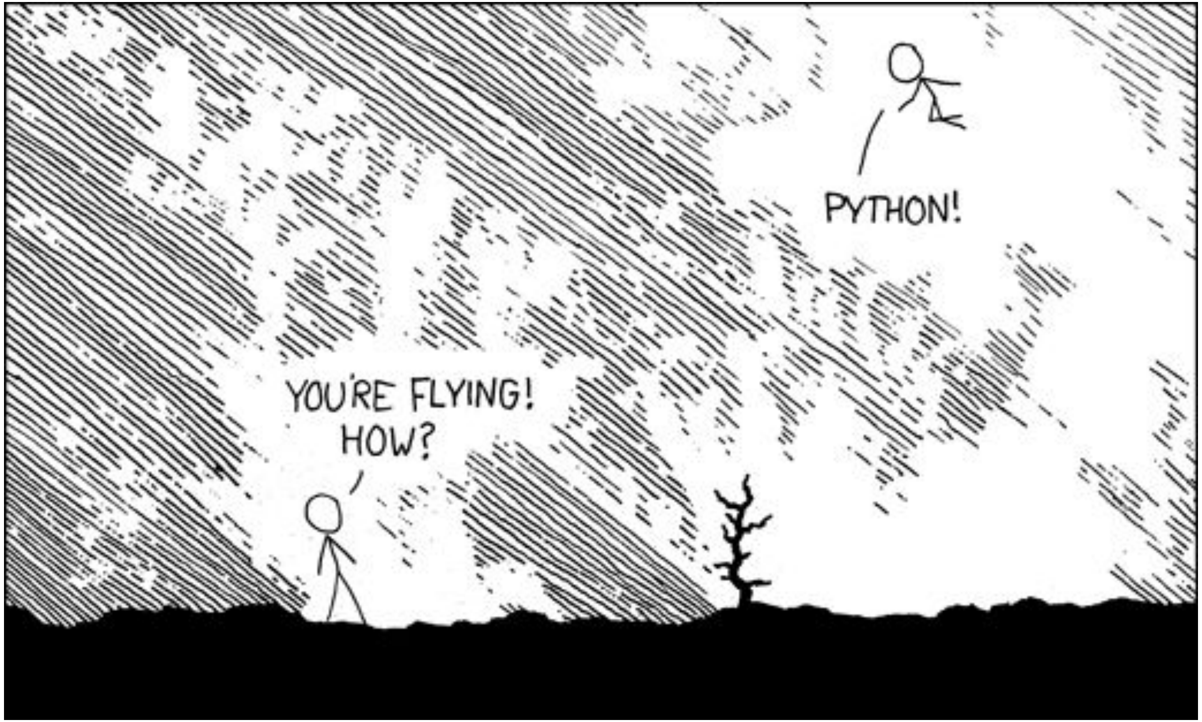
Functional programming

Compiled on runtime

SciPy / Science ecosystem

Modules are mutable objects

eval("import my_code")!



Tools: Taurus

Python + Qt

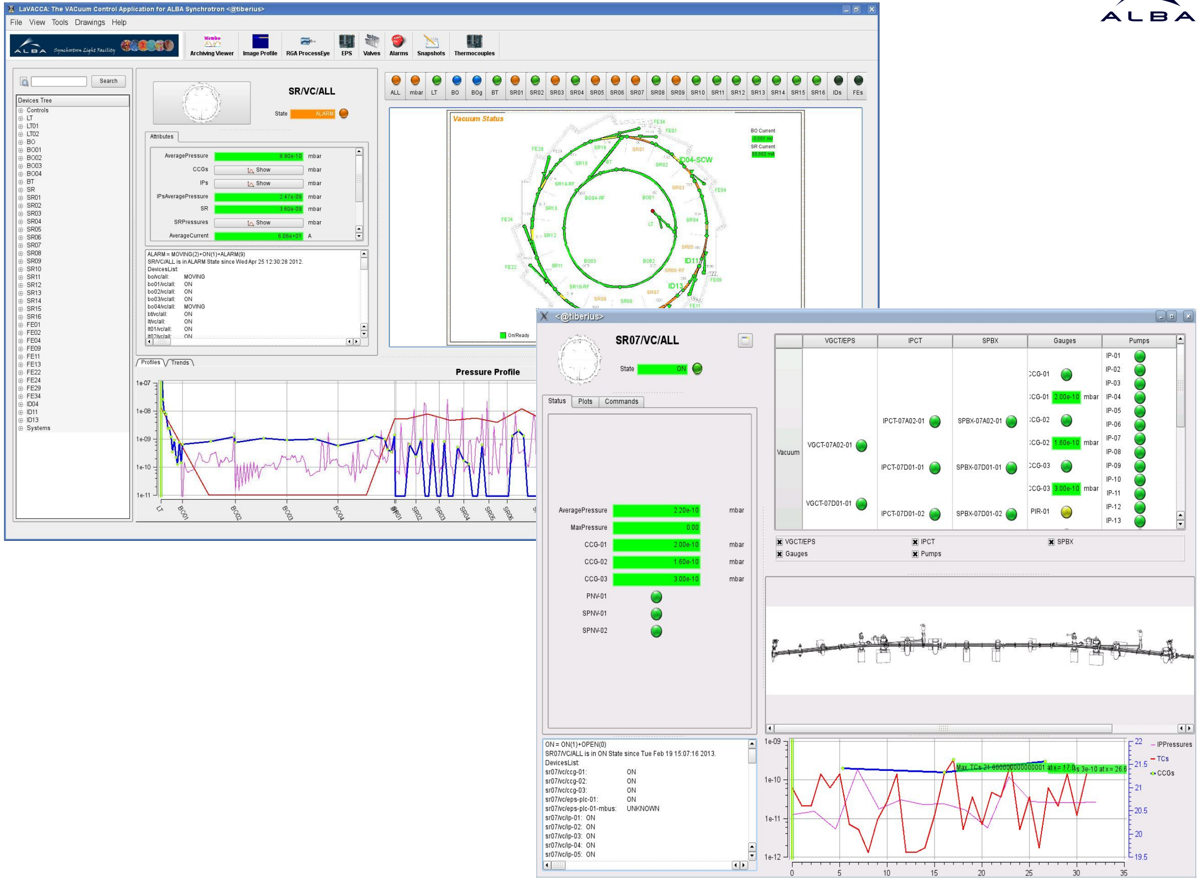
Windows/Linux

Core + Library of widgets

Tango / Epics / Sardana

Design by wizard / drag & drop

Few/none lines of code



MOBPP05

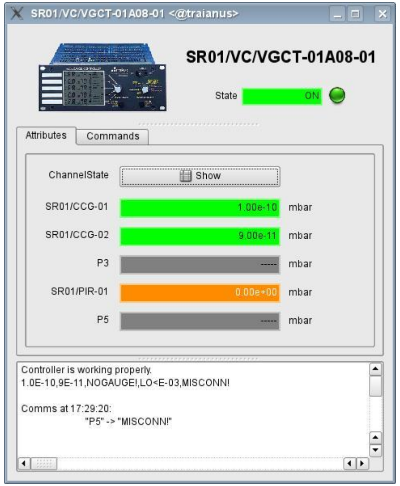
Tools: Dynamic Devices in TANGO

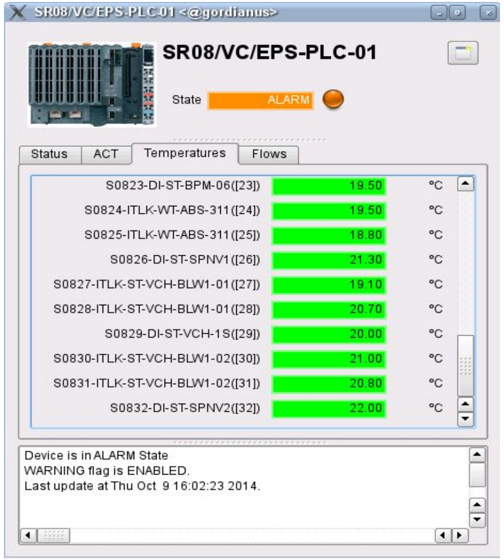
TANGO Control System can be adapted to change in different ways:

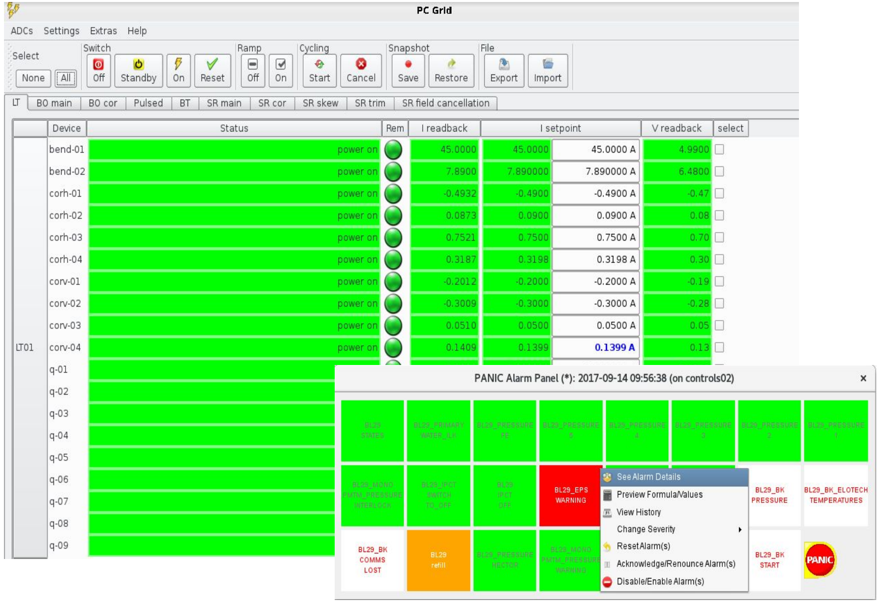
- **Dynamic Devices** create Attributes depending on Hardware channels (DAQ)
- **Processor Devices** allow to add/modify Attributes using existing attributes in Python formulas (PLC's)
- **Composer Devices** provide Attributes evaluated from formulas that access Attributes from other Devices (Alarm System)
- **Dynamic User Interfaces** load devices and create widgets on runtime to visualize all available information.

Device Attributes created on-the-fly

With Fixed or Variable number of attributes



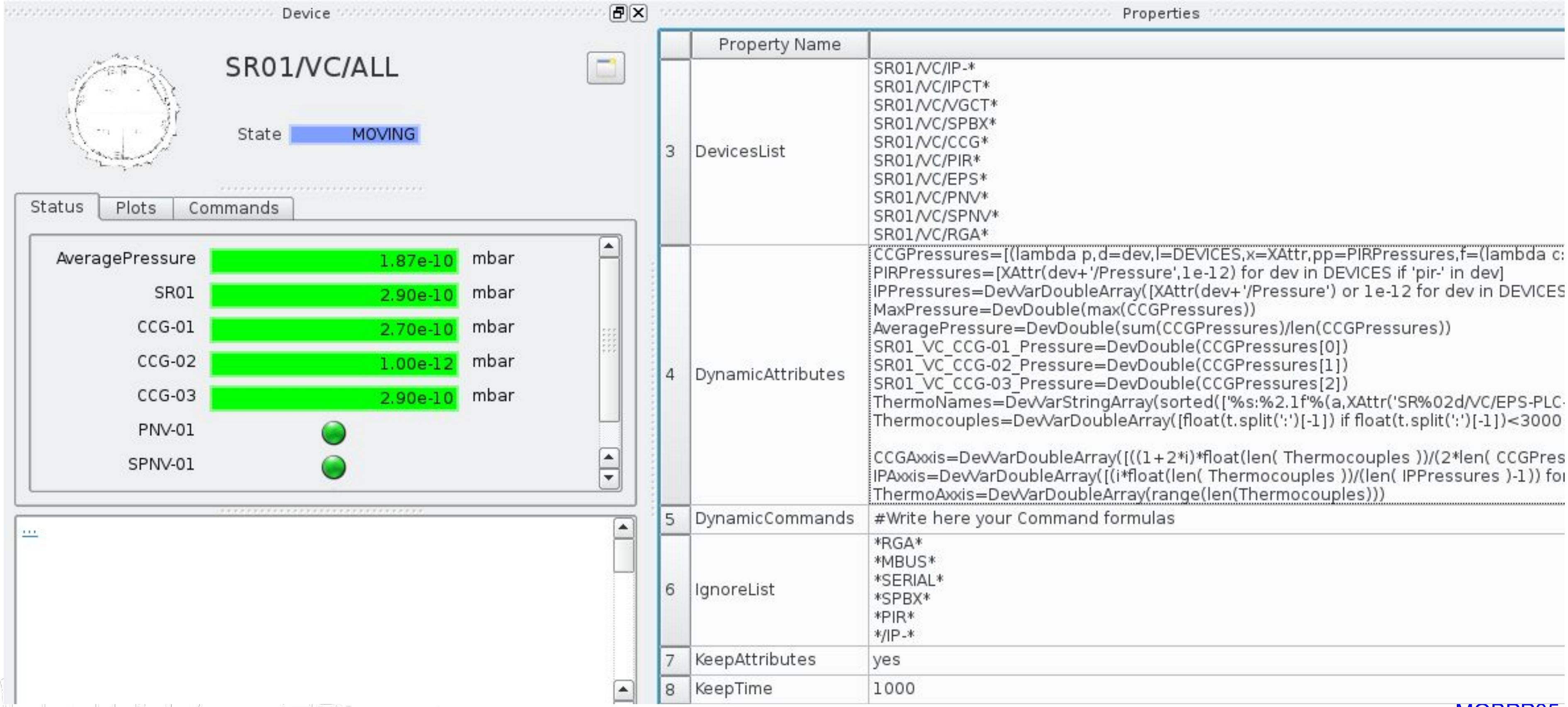




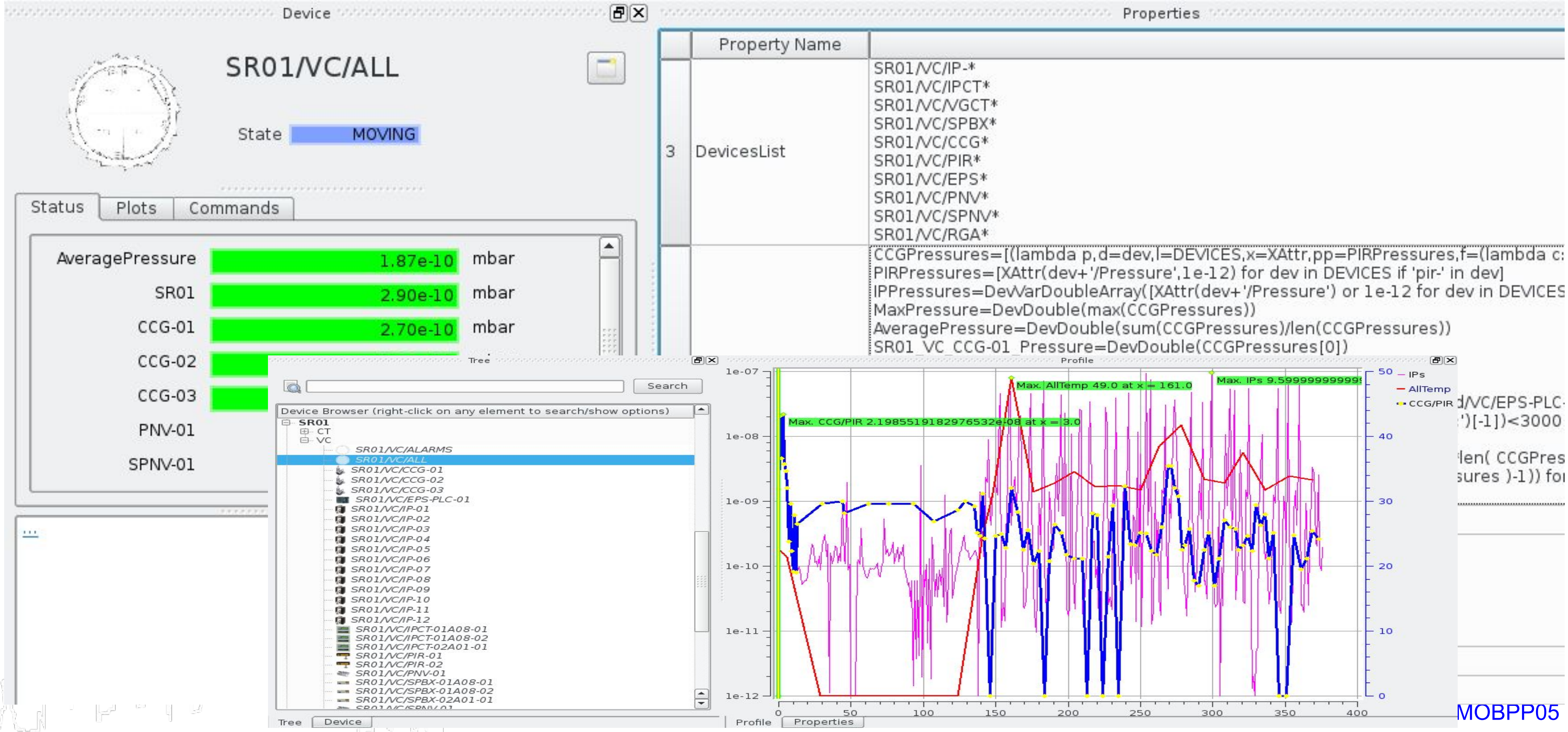
User Interfaces generated on-the-fly

MOBPP05

Composer Devices



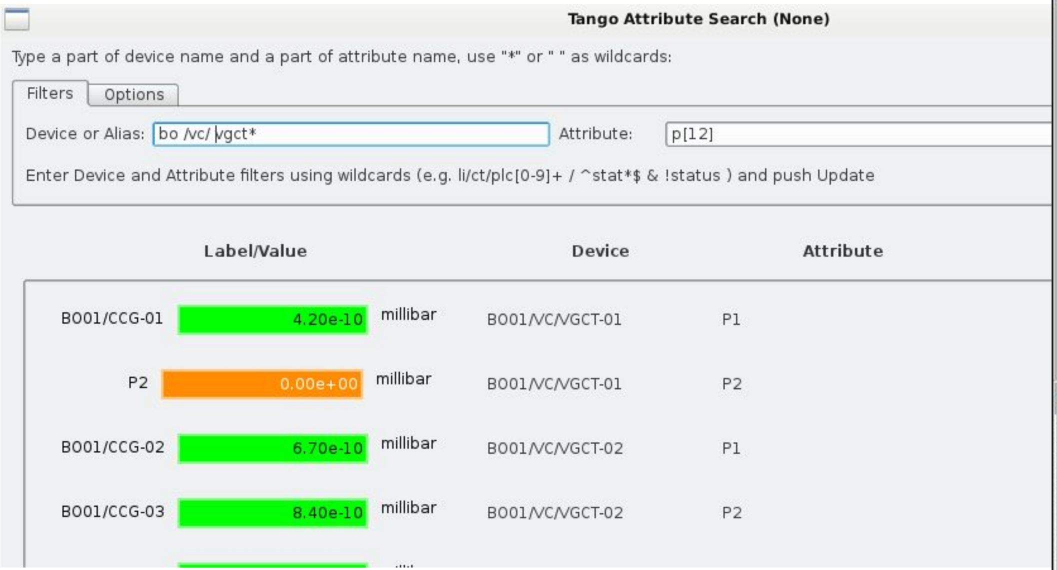
Composer Devices

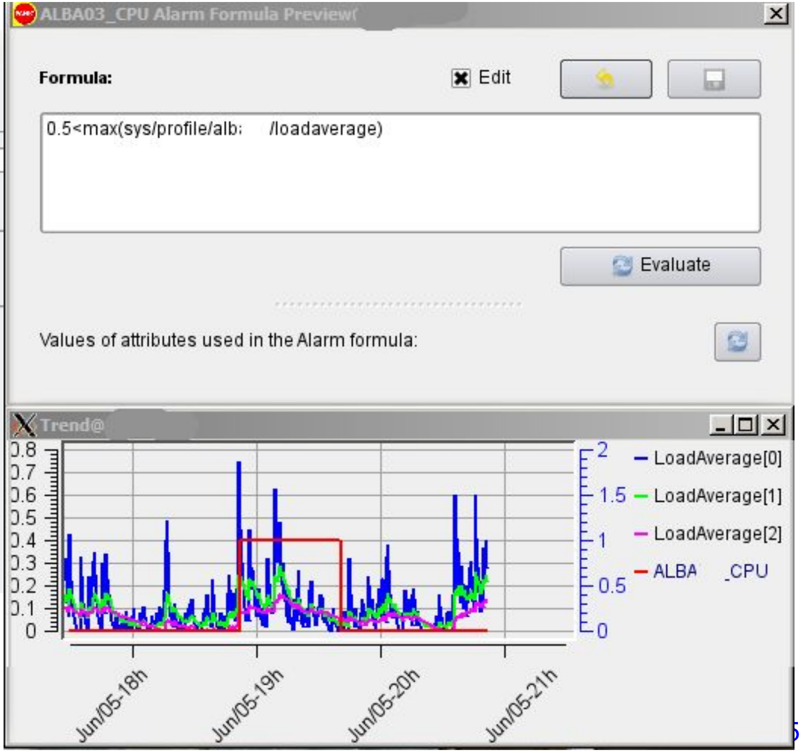


MOBPP05

User Interfaces generated on-the-fly

Either from searches or alarm formulas





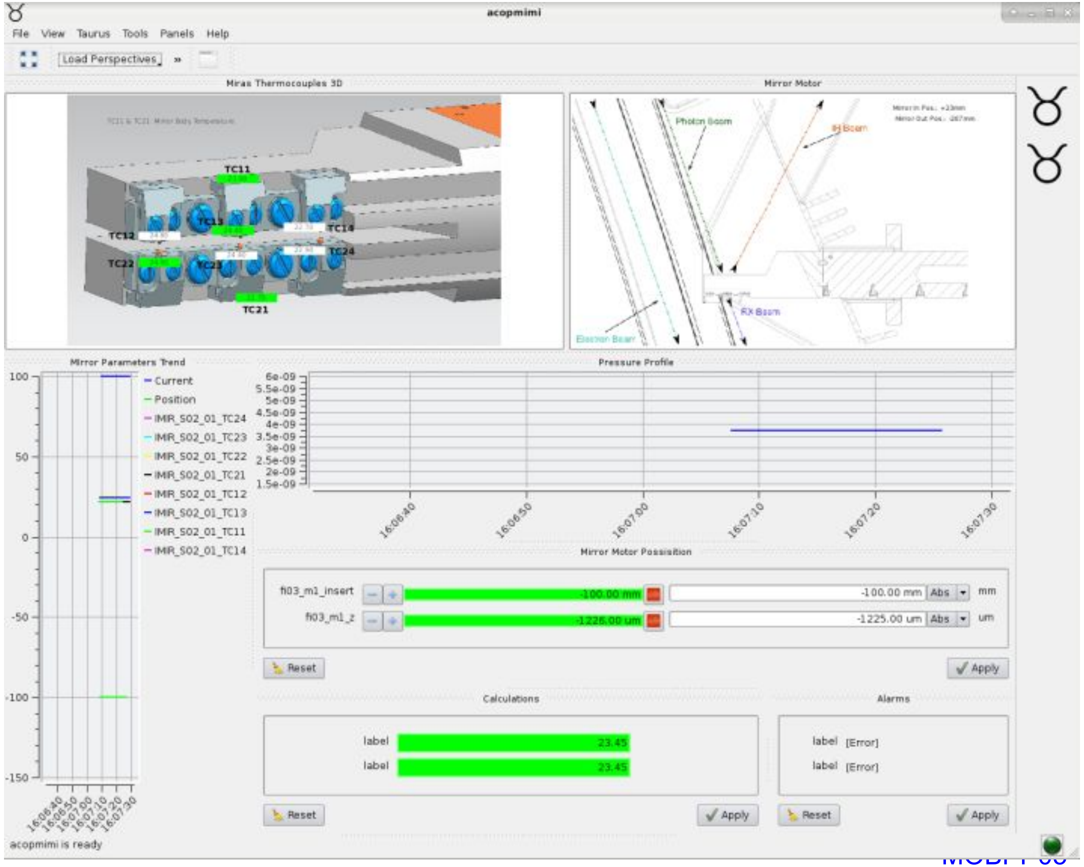
Users as developers: codeless applications

Operators develop their own app using drag and drop widgets.

Thus freeing control engineers for harder tasks

But, apps are stored locally:

- Hard to keep track of versions
- Hard to test after upgrades
- Solved enforcing a common shared folder for tools (with snapshots) and **git**



Scaling issues

Increase of cpu/memory problems on the client side

- huge applications generated automatically
- too many connections, high cpu/memory usage

cpu/memory problems on server side, caused by clients

- timeouts
- excessive polling, too many clients accessing the same devices
- deadlocks by interrupted commands (uncoherent state)

Users as developers: Dynamic Attributes

Many users are experimented programmers, although not worried about performance or scaling issues.

To allow them to write code freely, we provide Dynamic Devices for calcullations or building prototypes.

Calcullations results become available in Control System tools (User Interfaces, Archiving, Alarms).

Device properties [test/machinestatus/lifetime]

Property name	Value
DynamicAttributes	TIMES=DevVarDoubleArray([b.time.tv_sec+b.time.tv_usec*1e-6 for b in VAR('BUFF')]) VALS=DevVarDoubleArray([b.value for b in VAR('BUFF')]) AVG=numpy.average(VALS) DI=abs(VALS[-1]-VALS[0]) DT=TIMES[-1]-TIMES[0] Lifetime=ATTR('AVG') * ATTR('DT') / ATTR('DI') / 3600.0 if (DI > 0 and AVG > 0) else 0.0 LifeAvg=numpy.average(VAR('LT',VAR('LT')[-10:])) Current=ATTR('VALS')[-1] Product=Lifetime*Current
DynamicCommands	GET_BUFF=str(VAR('BUFF',XDEV('sr/di/dcct').attribute_history('averagecurrent',10))) ADD_LT=str(VAR('LT',default=[]).append(ATTR('Lifetime')))
__SubDevices	sr/di/dcct

Users as Developers: Controlled Scripts

Running an script within a TANGO DynamicDS Device helps us to "Control" how the script is executed

Device properties [SR06/CT/CALC-A]

Property name	Value
B	2.701
CheckDependencies	True
DEFAULT_VALUE	450
ExtraModules	PyTangoArchiving.Reader as HDB oserres as oriol
IgnoreList	
KeepAttributes	True
KeepTime	2000
LoadFromFile	/control/user-scripts/composers/sr_rf_calc_common.py
LogLevel	WARNING
Machine_Pforw	SR06/RF/FACADE-A/CAV_FW
PLANT	SR06A
PollingCycle	1000
SortLists	True
UseEvents	False
UseTaurus	False

Users as Developers: Controlled Scripts

Running an script within a TANGO DynamicDS Device helps us to "Control" how the script is executed

Device properties [SR06/CT/CALC-A]

Property name
B
CheckDependencies
DEFAULT_VALUE
ExtraModules
IgnoreList
KeepAttributes
KeepTime
LoadFromFile
LogLevel
Machine_Pforw
PLANT
PollingCycle
SortLists
UseEvents
UseTaurus

```
##################################################################################
#                              SIMULATION                              #
##################################################################################

#                              Constant Parameters                              #
##################################################################################
Uo = 1
Uc = 1.1
Rs = 3.3e6
Qo = 29500
PI = 3.1415927
Working_Frequency = 499650000
B06A = 2.701
B06B = 2.687
B10A = 2.4582
B10B = 2.5467
B14A = 2.678
B14B = 2.5639

#                              Select ID open or ID closed                              #
##################################################################################
# Flag to control if the IDs are open: True when open; False when closed
idsOpen = bool(VAR('idsOpen',VALUE) if WRITE else VAR('idsOpen') or False)
U = Uo if idsOpen else Uc

#                              Select Ibeam                              #
##################################################################################
machinelbeam = XATTR('sr/di/dcct/averagecurrent')
usrDeflbeam = float(VAR('UserCurrent',VALUE) if WRITE else VAR('UserCurrent',default=130))
setUsrDeflbeam = bool(VAR('usrlbeam',VALUE) if WRITE else VAR('usrlbeam', default=False))
```

Scripts versions can be managed by **git**

Scaling issues

Example of unexpected performance issues in the TANGO Database:

Scaling issues

Example of unexpected performance issues in the TANGO Database:

- Archiving the CPU usage of the TANGO Database, we correlated the problem to shutdowns and machine testing periods.

Scaling issues

Example of unexpected performance issues in the TANGO Database:

- Archiving the CPU usage of the TANGO Database, we correlated the problem to shutdowns and machine testing periods.
- No long query was producing the issue (typical search for removed devices/attributes).

Scaling issues

Example of unexpected performance issues in the TANGO Database:

- Archiving the CPU usage of the TANGO Database, we correlated the problem to shutdowns and machine testing periods.
- No long query was producing the issue (typical search for removed devices/attributes).
- Analyzing archived data with HDB++, we linked it to devices accessed by Matlab scripts (corrector magnets) during tests.

Scaling issues

Example of unexpected performance issues in the TANGO Database:

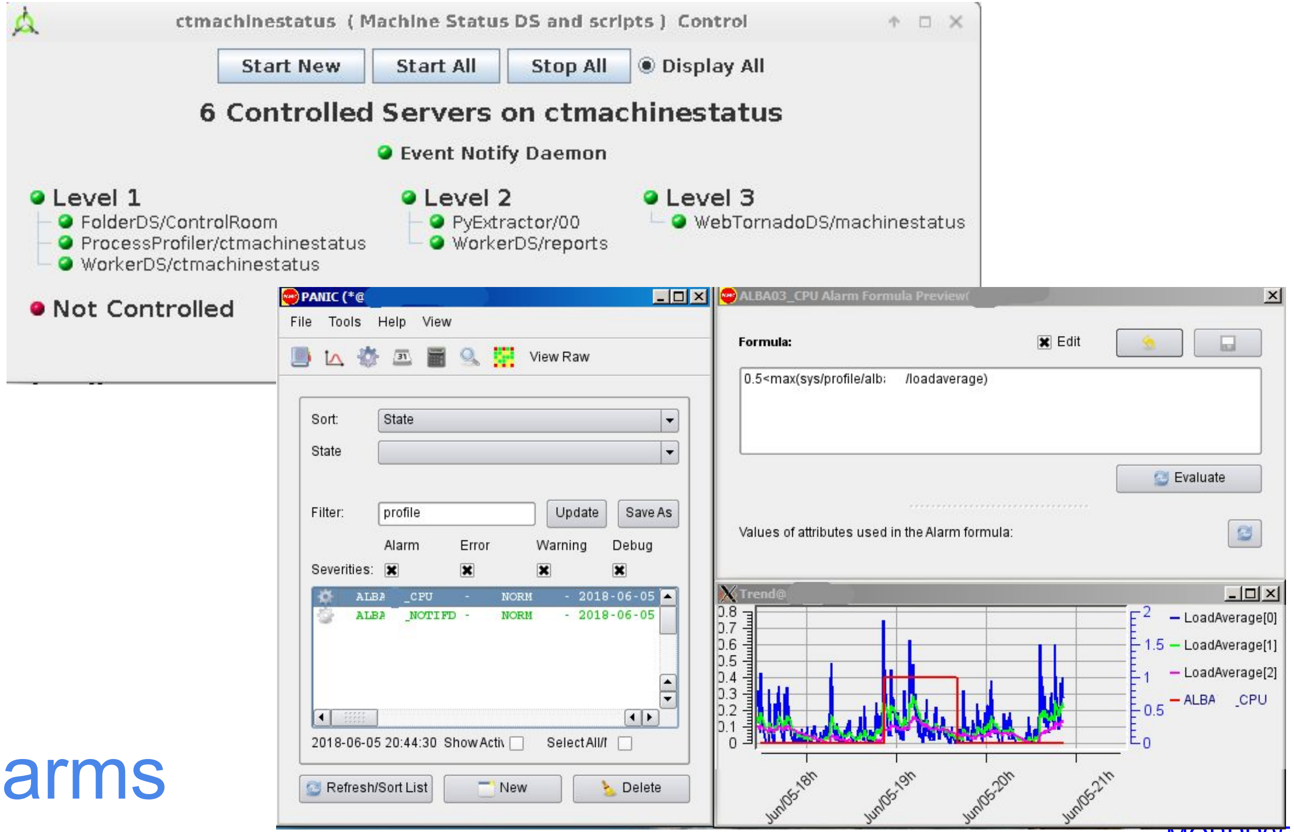
- Archiving the CPU usage of the TANGO Database, we correlated the problem to shutdowns and machine testing periods.
- No long query was producing the issue (typical search for removed devices/attributes).
- Analyzing archived data with HDB++, we linked it to devices accessed by Matlab scripts (corrector magnets) during tests.
- Finally, we found that, in the lack of settings, there were 88 devices checking their event configuration at 10Hz, doing short and fast queries but frequent and constant.

Tools: Process Profiler device

ProcessProfiler device provides cpu/ram stats.

Stats are provided as TANGO attributes, so it can be archived or used as alarm triggers.

It not only analyzes designed processes, but any process in the machine that takes too many resources.



Tools: PANIC Alarms

Tools: Using archiving for Diagnostics

TANGO HDB++ event-based Archiving provides diagnostic tools for evaluating the Control System Load. PyTangoArchiving API configures new archivers on demand.

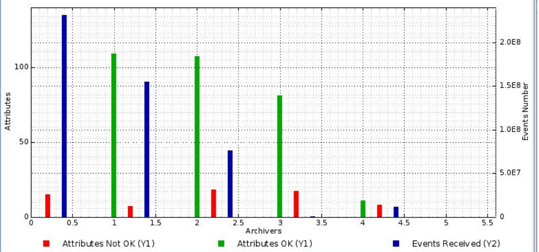
Statistics on 5 subscribers - 508 Attributes

Attribute Names	Ev. since re...	Av.Period.	Ev./2 h 00 m...	
sr07/di/bpm-10/say	12491943	---	588	
sr07/di/bpm-09/say	12491530	---	586	
sr07/di/bpm-10/sax	12489606	---	588	
sr07/di/bpm-09/sax	12488773	---	586	
fe11/di/xbpmlocum-01/ypos	10483801	---	413	
fe11/di/xbpmlocum-01/xpos	10483623	---	413	
fe11/di/xbpmlocum-01/yid	8950172	---	413	
fe11/di/xbpmlocum-01/xid	8950020	---	413	
fe29/di/xbpmlocum-01/xpos	6713038	---	304	
fe29/di/xbpmlocum-01/ypos	6712589	---	304	
fe29/di/xbpmlocum-01/xid	6542053	---	300	
fe29/di/xbpmlocum-01/yid	6542042	---	300	
sr01/di/bpm-02/xpossa	6400027	---	301	
sr01/di/bpm-02/zpossa	6400016	---	300	
fe22/di/xbpmlocum-01/ypos	6222583	---	289	
fe22/di/xbpmlocum-01/xpos	6222528	---	289	
fe22/di/xbpmlocum-01/yid	6221609	---	289	
fe22/di/xbpmlocum-01/xid	6221570	---	289	24.91 sec.
fe29/di/xbpmlocum-01/y	5723940	---	300	24.00 sec.
fe29/di/xbpmlocum-01/x	5723872	---	300	24.00 sec.
fe11/di/xbpmlocum-01/y	5442093	---	256	
fe11/di/xbpmlocum-01/x	5441969	---	256	
fe22/di/xbpmlocum-01/y	5194753	---	279	
fe22/di/xbpmlocum-01/x	5194752	---	279	
lt01/di/bcm-01/charge	4025322	---	188	
lt02/di/bcm-01/charge	4025268	---	188	
li/di/bcm-01/charge	4025258	---	188	
fe24/di/xbpmlocum-01/xpos	4020682	---	197	
fe24/di/xbpmlocum-01/ypos	4020041	---	197	
fe04/di/xbpmlocum-01/ypos	4015881	---	182	
fe04/di/xbpmlocum-01/xpos	4015877	---	182	
bo/di/dcct/extractedbeam	3996926	---	188	
bo/di/dcct/injectedbeam	3996926	---	188	
bo/di/dcct/transmission	3996926	---	188	
sr/di/dcct/averagecurrent	3996911	---	187	
sr/di/dcct/transmission	3996911	---	187	

508 Attributes distributed in 5 Subscribers

Attributes | Performances | Table

◉ Attributes Not OK ◉ Attributes OK ◉ Events Received



69 Controlled Servers on archiving04

- Level 1
  - PyHdbppPeriodicArchiver/hdbvc-01
  - PyHdbppPeriodicArchiver/hdbvc-02
  - PyHdbppPeriodicArchiver/hdbvc-03
  - PyHdbppPeriodicArchiver/hdbvc-04
  - PyHdbppPeriodicArchiver/hdbvc-05
  - PyHdbppPeriodicArchiver/hdbvc-06
  - PyHdbppPeriodicArchiver/hdbvc-07
  - PyHdbppPeriodicArchiver/hdbvc-08
  - PyHdbppPeriodicArchiver/hdbvc-09
  - PyHdbppPeriodicArchiver/hdbvc-10
  - hdb++es-srv/hdbvc-01
  - hdb++es-srv/hdbvc-02
  - hdb++es-srv/hdbvc-03
  - hdb++es-srv/hdbvc-04
  - hdb++es-srv/hdbvc-05
  - hdb++es-srv/hdbvc-06
  - hdb++es-srv/hdbvc-07
  - hdb++es-srv/hdbvc-08
  - hdb++es-srv/hdbvc-09
  - hdb++es-srv/hdbvc-10
- Level 2
  - PyHdbppPeriodicArchiver/hdbdi-01
  - PyHdbppPeriodicArchiver/hdbdi-02
  - PyHdbppPeriodicArchiver/hdbdi-03
  - PyHdbppPeriodicArchiver/hdbdi-04
  - PyHdbppPeriodicArchiver/hdbpc-01
  - PyHdbppPeriodicArchiver/hdbpc-02
  - PyHdbppPeriodicArchiver/hdbpc-03
  - PyHdbppPeriodicArchiver/hdbpc-04
  - PyHdbppPeriodicArchiver/hdbpc-05
  - PyHdbppPeriodicArchiver/hdbpc-06
  - hdb++es-srv/hdbdi-01
  - hdb++es-srv/hdbdi-02
  - hdb++es-srv/hdbdi-03
  - hdb++es-srv/hdbdi-04
  - hdb++es-srv/hdbdi-05
- Level 3
  - PyHdbppPeriodicArchiver/hdl
  - PyHdbppPeriodicArchiver/hdl
  - PyHdbppPeriodicArchiver/hdl
  - PyHdbppPeriodicArchiver/hdl
  - PyHdbppPeriodicArchiver/hdl
  - hdb++es-srv/hdbct-01
  - hdb++es-srv/hdbct-02
  - hdb++es-srv/hdbct-03
  - hdb++es-srv/hdbct-04
  - hdb++es-srv/hdbct-05
- Level 5
  - PyHdbppPeriodicArchiver/hdbpp-per-rf-01
  - hdb++es-srv/hdbrf
  - hdb++es-srv/hdbrf-02
  - hdb++es-srv/hdbrf-03
  - hdb++es-srv/hdbrf-04
- Level 6
  - PyExtractor/01
  - WebTornadoDS/VC
  - WebTornadoDS/report01
  - hdb++cm-srv/hdbacc
  - hdb++cm-srv/hdbct
  - hdb++cm-srv/hdbdi-01
  - hdb++cm-srv/hdbpc
  - hdb++cm-srv/hdbrf
  - hdb++cm-srv/hdbvc
  - hdbppes/test
- Level 7
  - PT104Pico/PT104Dev01
  - PT104PicoCh/Channels

hdbpp-configurator-3.7

File View help

HDB++ Diagnostics

	Faulty	Started	Paused	Stop...	Pendi...	ev/2 h...	Fail./...	Context
tango://alba03.cells.es:10000/archiving/hdbdi/es-01	15	150	0	0	0	10383.0(	68.00	ALWAYS
tango://alba03.cells.es:10000/archiving/hdbdi/es-02	7	116	0	0	0	7326.00	30.00	ALWAYS
tango://alba03.cells.es:10000/archiving/hdbdi/es-03	18	125	0	0	0	3537.00	8.00	ALWAYS
tango://alba03.cells.es:10000/archiving/hdbdi/es-04	17	98	0	0	0	24.00	0.00	ALWAYS
tango://alba03.cells.es:10000/archiving/hdbdi/es-05	8	19	0	0	0	554.00	0.00	ALWAYS

	Faulty	Started	Paused	Stopped	Pending	ev/2 h 00 ...	Fail./2 h 0...	Context
E.S. Manager	65	508	0	0	0	21824.00	106.00	Not initialised

MOBPP05

Future: from Virtual Machines to Containers

Dynamic Control Systems is a problem of scaling an ever-increasing need of resources:

faster cpu's -> more data -> bigger memory usage -> slower performance

Then, as we already create Devices on-the-fly:

- Should we explore the creation of new VM's or Containers on-the-fly?
- If devices and applications run on independent containers, how do we manager their growth?
- Using containers, limits between Control Engineering and System Administration get blurry.

Summary

- Enabling dynamicity on the Control System allow to adapt faster to change.

- Enabling users as developers reduces the gap between groups:

- Scaling requires to have the proper tools:

Summary

- Enabling dynamicity on the Control System allow to adapt faster to change.
  - if data repositories (Cabling/Control Databases) are updated accordingly
  - if IT resources are continuously monitored and scaling strategies are adopted
- Enabling users as developers reduces the gap between groups:
  - but gap must be further reduced through training and best practices enforcement
  - Users must be conscious of the effect of their applications in terms of performance
- Scaling requires to have the proper tools:
  - integration of IT resources with Control System tools (Archiving/Alarms)
  - new diagnostic Attributes must be needed on devices (MemUsage, Blackbox)

Thanks/Gràcies for your attention

?