AlarmHandler

G.Scalamera
L.Pivetta

http://www.tango.controls.org

What is AlarmHandler

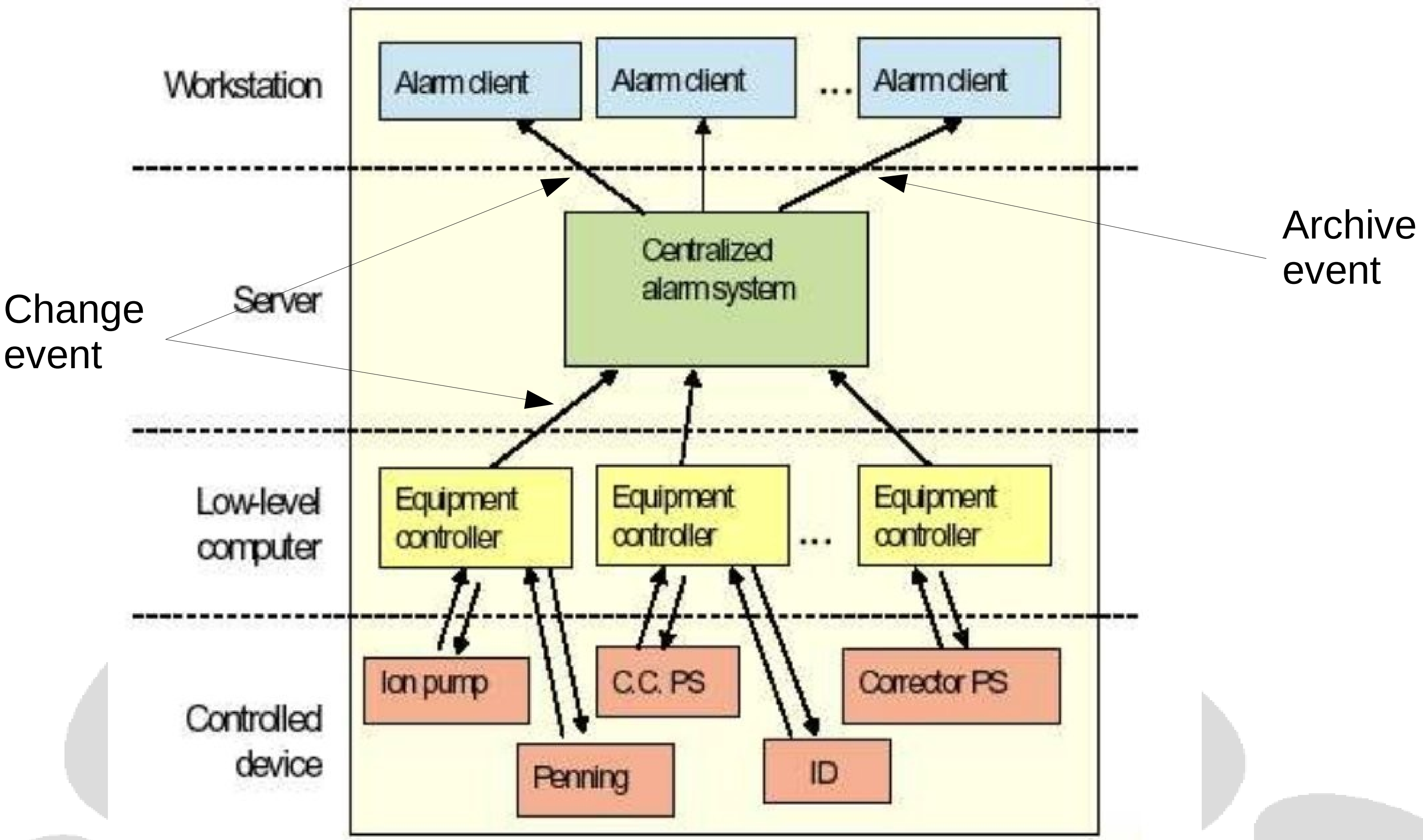
- a Tango device server

- the evolution of the Alarm device server developed at Elettra

- an efficient event-driven, higly configurable rule-based engine

- based on IEC 62682: alarm state values, key concept and functionalities adapted to the standard

Event based: change event (+archive event)

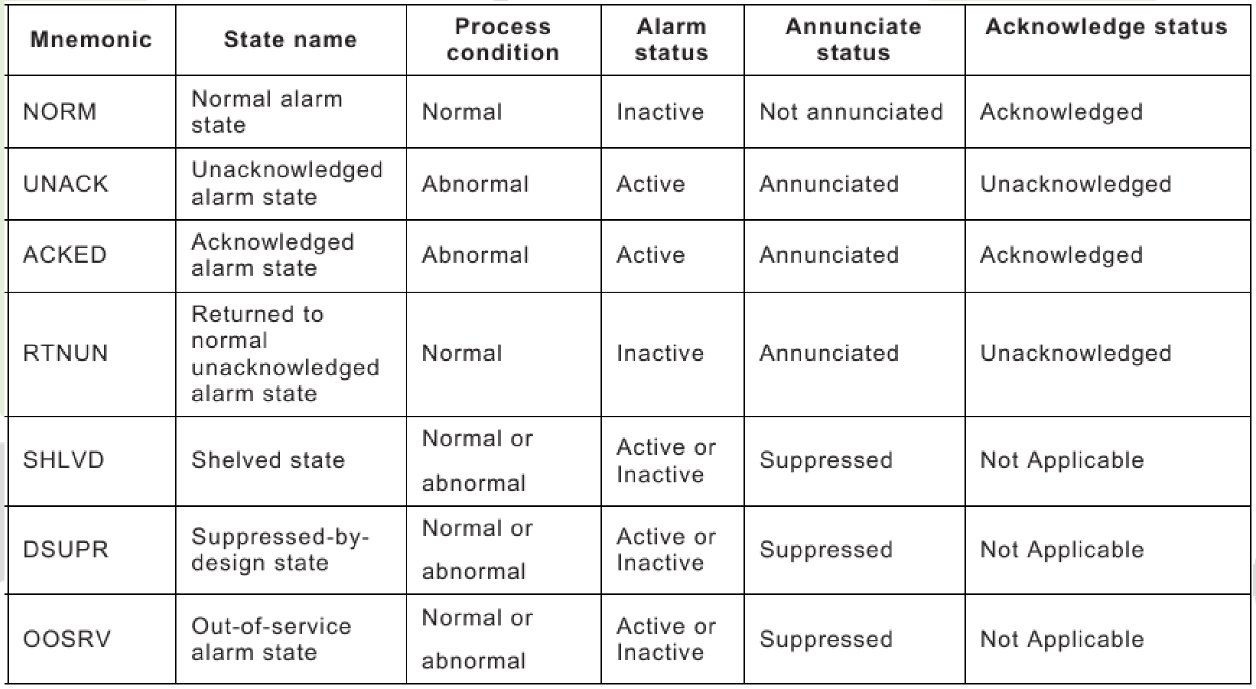


What’s new in AlarmHandler

Interface:

- for every alarm a dynamic attribute is created, change/archive events are pushed in the code every time its value change

- attributes are DevEnum type so that alarm status, acknowledged, enabled informations are coded into; possible values are: NORM, UNACK, ACKED, RTNUN, SHLVD, DSUPR, OOSRV



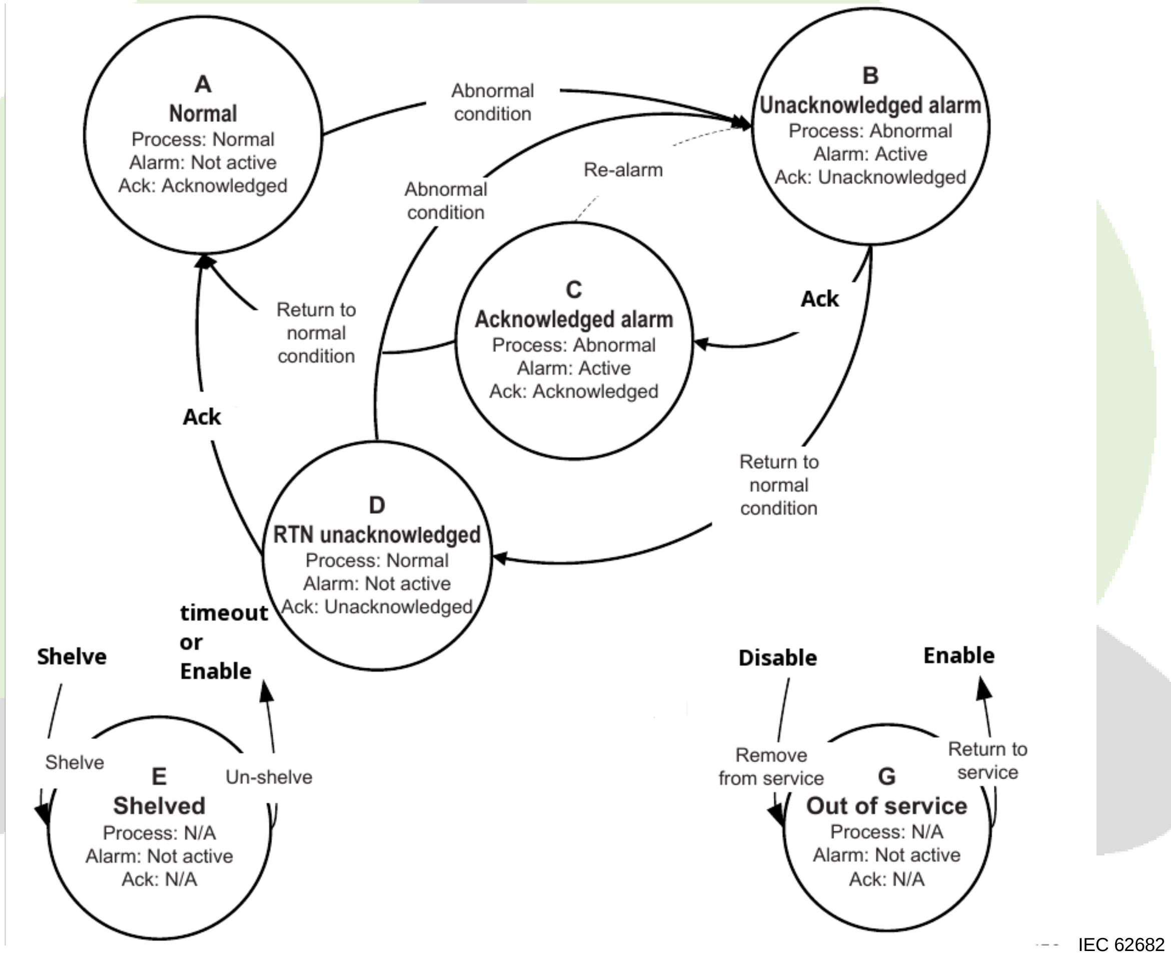
Mnemonic	State name	Process condition	Alarm status	Annunciate status	Acknowledge status
NORM	Normal alarm state	Normal	Inactive	Not annunciated	Acknowledged
UNACK	Unacknowledged alarm state	Abnormal	Active	Annunciated	Unacknowledged
ACKED	Acknowledged alarm state	Abnormal	Active	Annunciated	Acknowledged
RTNUN	Returned to normal unacknowledged alarm state	Normal	Inactive	Annunciated	Unacknowledged
SHLVD	Shelved state	Normal or abnormal	Active or Inactive	Suppressed	Not Applicable
DSUPR	Suppressed-by-design state	Normal or abnormal	Active or Inactive	Suppressed	Not Applicable
OOSRV	Out-of-service alarm state	Normal or abnormal	Active or Inactive	Suppressed	Not Applicable

IEC 62682

What’s new in AlarmHandler

Interface:

- added Enable, Disable, Shelve commands



What’s new in AlarmHandler

Alarm formula:

- quality is supported in formula with 2 syntaxes:
  - name/of/device/attr**.quality**
  - **quality(**expression involving attributes**)**

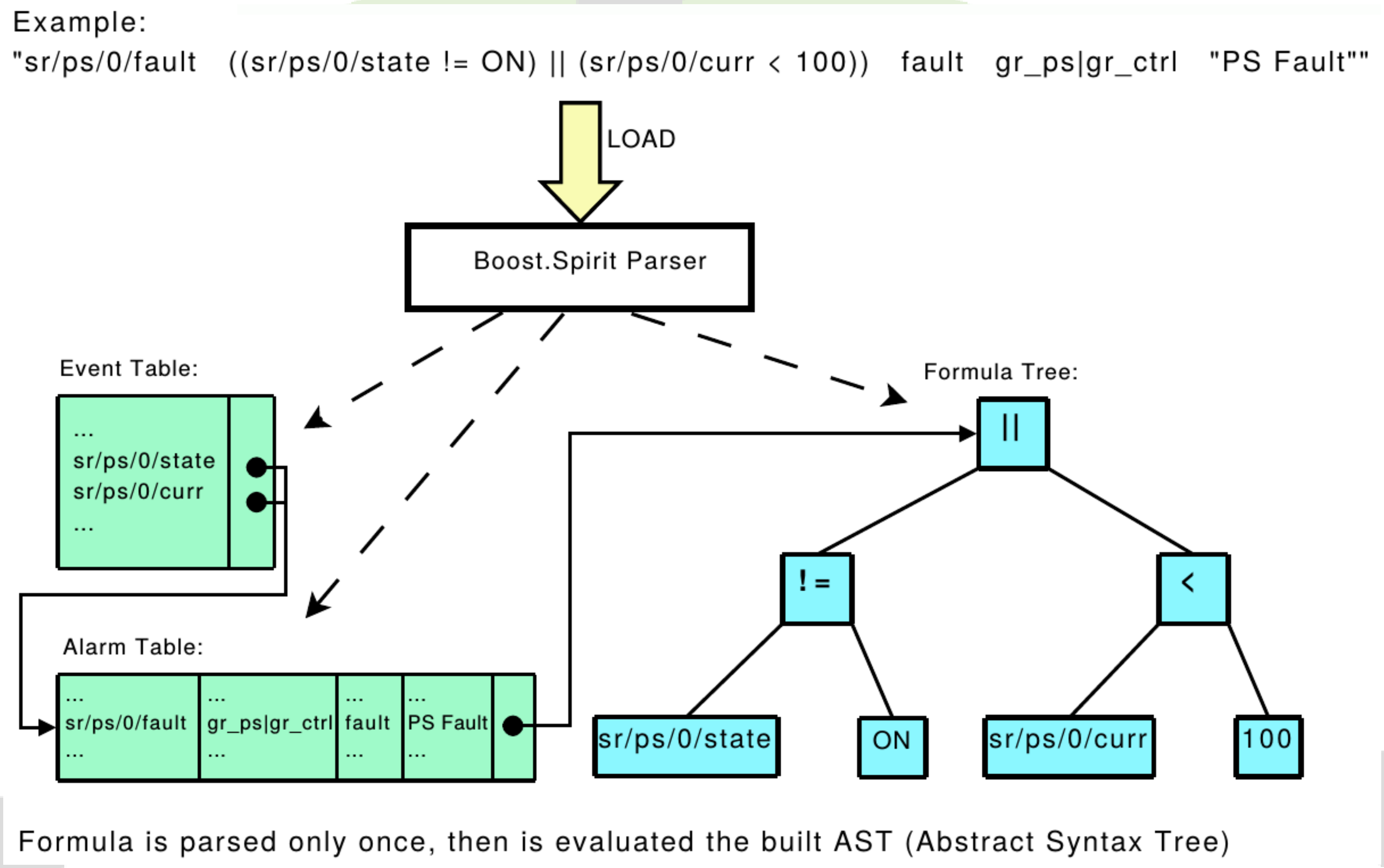
- to ease the use of alarm attributes in another alarm formula the following are supported:
  - name/of/device/attr**.alarm** evaluates true if ==UNACK || ==ACK
  - name/of/device/attr**.normal** evaluates true if ==NORM || ==RTNUN

- added ternary if condition: *(expr1* ***?*** *expr2* ***:*** *expr3)*

- added some functions: ***max(****expr1,expr2****)***, ***min(****expr1,expr2****)***, ***pow(****expr1,expr2****)***

Boost.Spirit Parser

High performance parser handles easily ~1000 alarms in one instance at Elettra

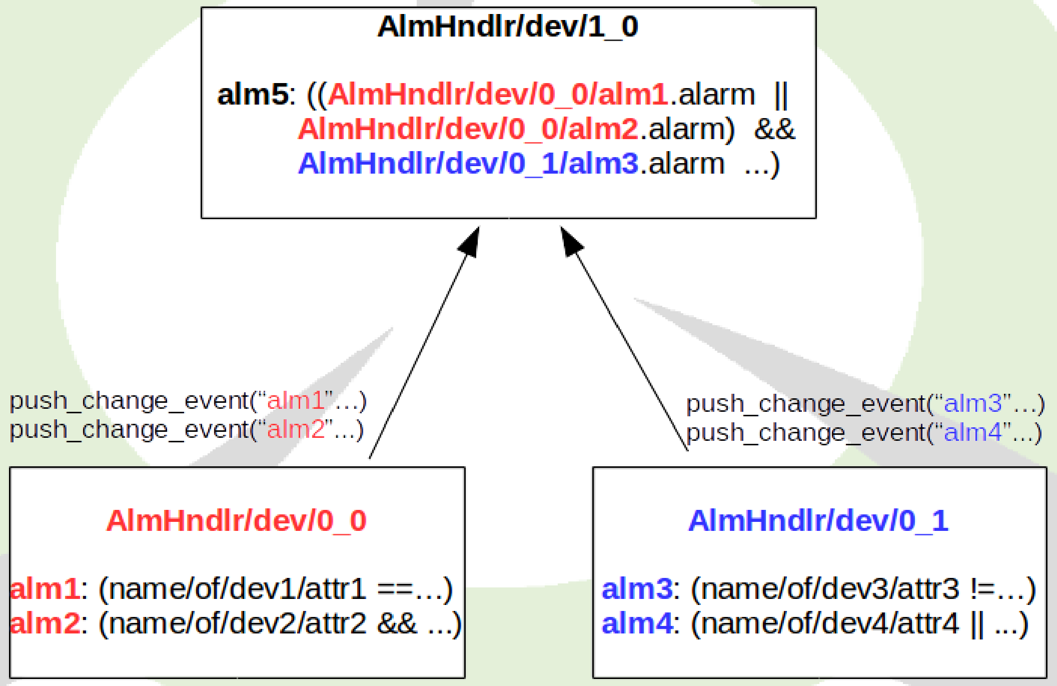


Formula is parsed only once, then is evaluated the built AST (Abstract Syntax Tree)

What's new in AlarmHandler

Alarm hierarchy:

- with an attribute for each alarm it is much easier to build an hierarchy of AlarmHandlers


AlmHndlr/dev/1_0
alm5: ((AlmHndlr/dev/0_0/alm1.alarm ||
AlmHndlr/dev/0_0/alm2.alarm) &&
AlmHndlr/dev/0_1/alm3.alarm ...)
push_change_event("alm1"...)
push_change_event("alm2"...)
push_change_event("alm3"...)
push_change_event("alm4"...)
AlmHndlr/dev/0_0
alm1: (name/of/dev1/attr1 ==...)
alm2: (name/of/dev2/attr2 && ...)
AlmHndlr/dev/0_1
alm3: (name/of/dev3/attr3 !=...)
alm4: (name/of/dev4/attr4 || ...)

What’s new in AlarmHandler

Error reporting:

- each attribute re-throws the exceptions received while evaluating the formula (if more than one attribute in the formula is in error just the first is reported)

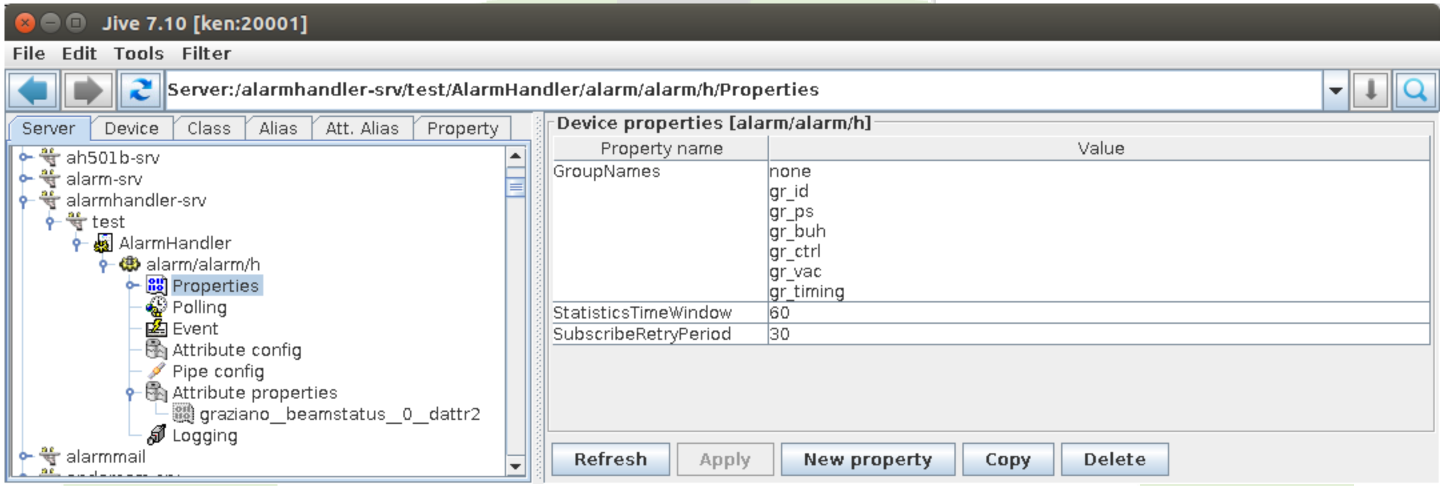
- quality of the attribute is the result of the combination of attributes in the formula (INVALID if at least one invalid, ALARM if at least one in alarm, WARNING if at least one in warning, CHANGING if at least on in changing, VALID otherwise)

What’s new in AlarmHandler

Configuration:

- the dedicated MySQL DB has been dropped
- alarms configuration is in the Tango DB
    - device properties are used for configuration of the device
    - attribute properties are used for configuration of alarms
- configuration of alarms at runtime with key=value; syntax in the Load command

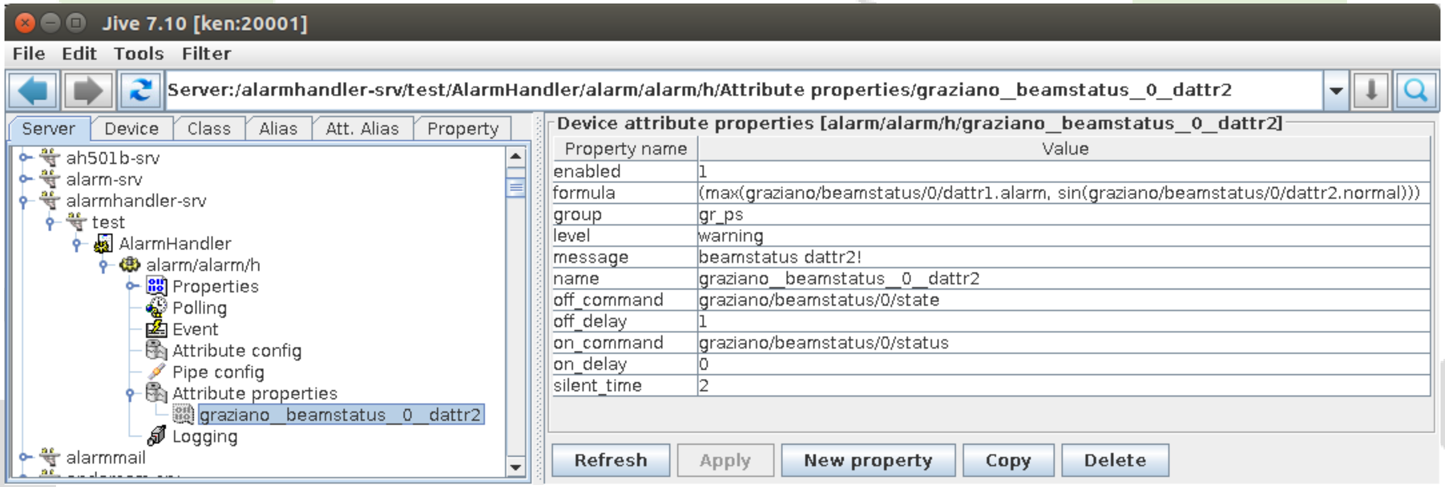
What's new in AlarmHandler



Server:/alarmhandler-srv/test/AlarmHandler/alarm/alarm/h/Properties

Device properties [alarm/alarm/h]

Property name	Value
GroupNames	none gr_id gr_ps gr_buh gr_ctrl gr_vac gr_timing
StatisticsTimeWindow	60
SubscribeRetryPeriod	30



Server:/alarmhandler-srv/test/AlarmHandler/alarm/alarm/h/Attribute properties/graziano__beamstatus__0__dattr2

Device attribute properties [alarm/alarm/h/graziano__beamstatus__0__dattr2]

Property name	Value
enabled	1
formula	(max(graziano/beamstatus/0/dattr1.alarm, sin(graziano/beamstatus/0/dattr2.normal)))
group	gr_ps
level	warning
message	beamstatus dattr2!
name	graziano__beamstatus__0__dattr2
off_command	graziano/beamstatus/0/state
off_delay	1
on_command	graziano/beamstatus/0/status
on_delay	0
silent_time	2

http://www.tango-controls.org

What’s new in AlarmHandler

Alarm history:

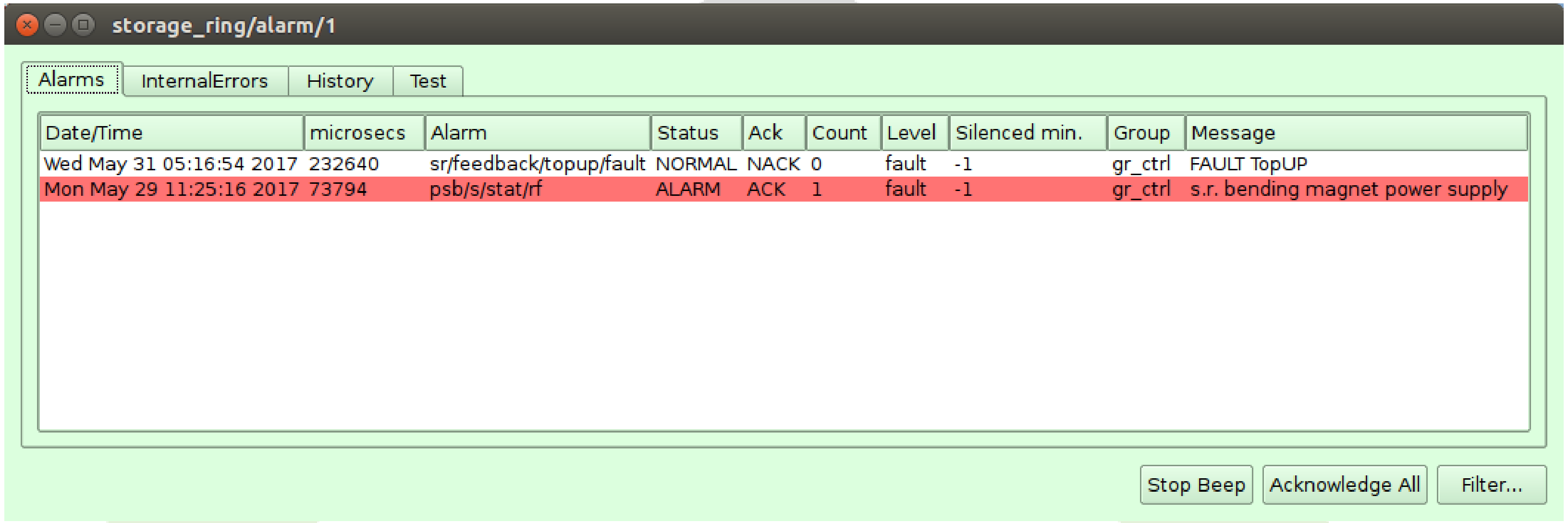
- the dedicated MySQL DB has been dropped

- every attribute created for an alarm push archive event in the code

- alarm state can be archived with HDB++

  → easy to correlate with other data in HDB++

Alarm GUI:

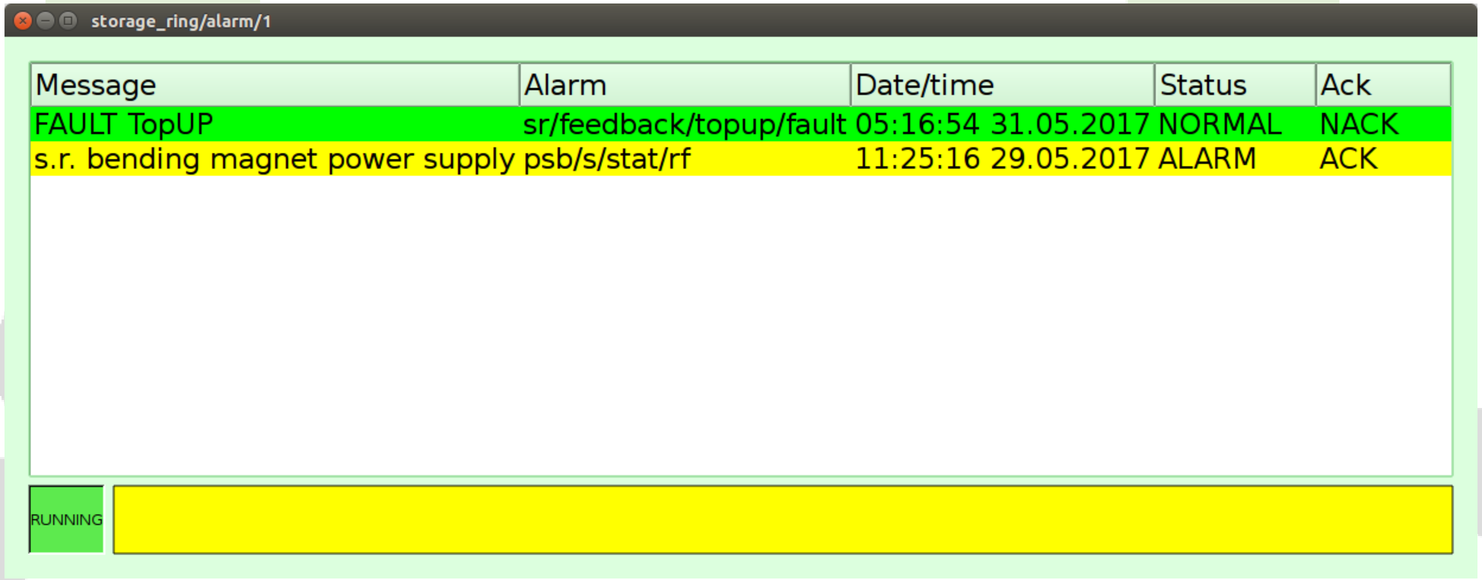


storage_ring/alarm/1

Alarms | InternalErrors | History | Test

Date/Time	microsecs	Alarm	Status	Ack	Count	Level	Silenced min.	Group	Message
Wed May 31 05:16:54 2017	232640	sr/feedback/topup/fault	NORMAL	NACK	0	fault	-1	gr_ctrl	FAULT TopUP
Mon May 29 11:25:16 2017	73794	psb/s/stat/rf	ALARM	ACK	1	fault	-1	gr_ctrl	s.r. bending magnet power supply

Alarm Monitor:



storage_ring/alarm/1

Message	Alarm	Date/time	Status	Ack
FAULT TopUP	sr/feedback/topup/fault	05:16:54 31.05.2017	NORMAL	NACK
s.r. bending magnet power supply	psb/s/stat/rf	11:25:16 29.05.2017	ALARM	ACK

What’s next

- more tests, bug fixing, ..

- update QT alarm GUI

- add the possibility to automatically configure alarm attributes in HDB++ for history

- integrate in PANIC → add/rename properties, attributes, commands, …

- https://github.com/ELETTRA-SincrotroneTrieste/alarmhandler

PANIC integration

	AlarmHandler	PyAlarm	Integration
Alarm configuration	Alarm attributes	Free/Device/Attribute properties, csv files, custom DB, ...	Test if Attribute properties performant with ~1000 alarms
Alarm attributes	name, formula, level, message	tag_name, formula, severity, description	Define common names: tag, formula, priority, message
Alarm group	grouped using group alarm attribute	grouped using class/device/view	Support both
Alarm actions	on_command, off_command	receivers	AlarmHandler should support more receivers, agree on syntax
Alarm severity	level	severity	priority
Interface: enum values	NORM, UNACK, ACKED, RTNUN, SHLVD, DSUPR, OOSRV	Bool attributes	Support enum values with NORM=0
Interface: attributes	Enum attributes one per alarm + one string array for each alarm state	Bool (+quality) attributes one per alarm + string arrays with coded information in it	Enum attributes + AlarmSummary string array with everything needed coded in it as key=value;...

Thank you!

Any questions?

http://www.tango.controls.org