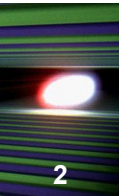


Visualization Tool Development

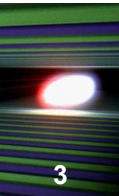
Control Software

Kerstin Weger, European XFEL GmbH
1st Meeting of the European XFEL Accelerator Consortium
April 17, 2012


What will be presented?



- basic technical and functional requirements
- main concepts & features
- initial prototype of the multi-purpose GUI
- current status
 - How to get users started as quickly as possible?



■ Technical Requirements

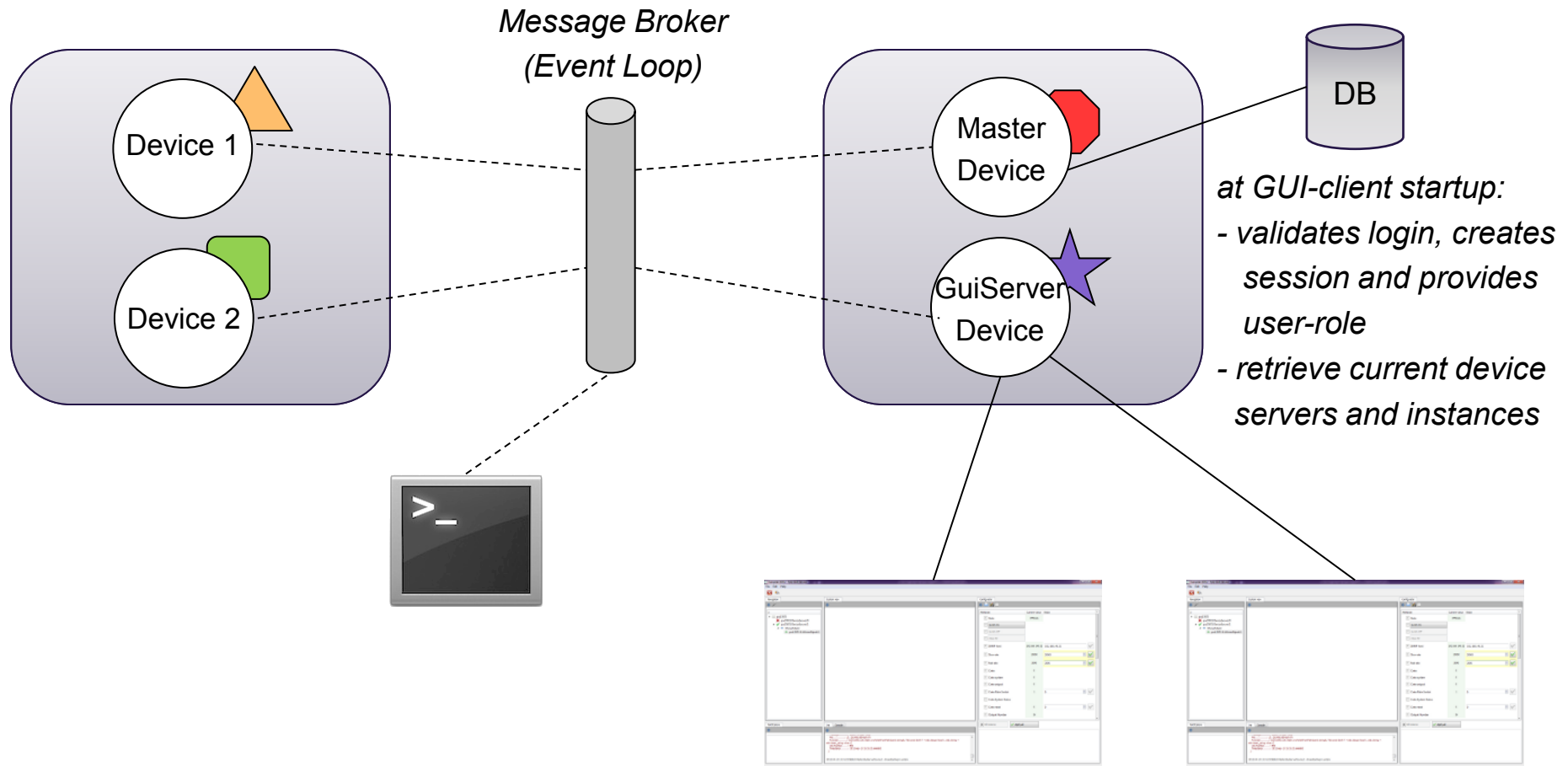
- cross-platform ( ,  , )
- remote control (client/server architecture)
- graphical views and embedded scripting (IP[y]:)
- integration to the software framework's signals/slots
 - immediate feedback of the framework's events

■ Functional Requirements

- automatic, self explaining default layout allowing interaction with any device
- allow preparation of highly customizable user panels
 - custom widget plugin loader (3rd party)
 - drag & drop layouting
- user-centralized and role-based system
- optimized network traffic (only send what needs to be displayed)
- full navigation and status overview within the distributed system

Connecting the bits and pieces

4



- MasterDevice: keeps DB up-to-date
- GuiServerDevice: registers all GUI logins, firewall (internal/external access)

Generic Default GUI Layout (Prototype)

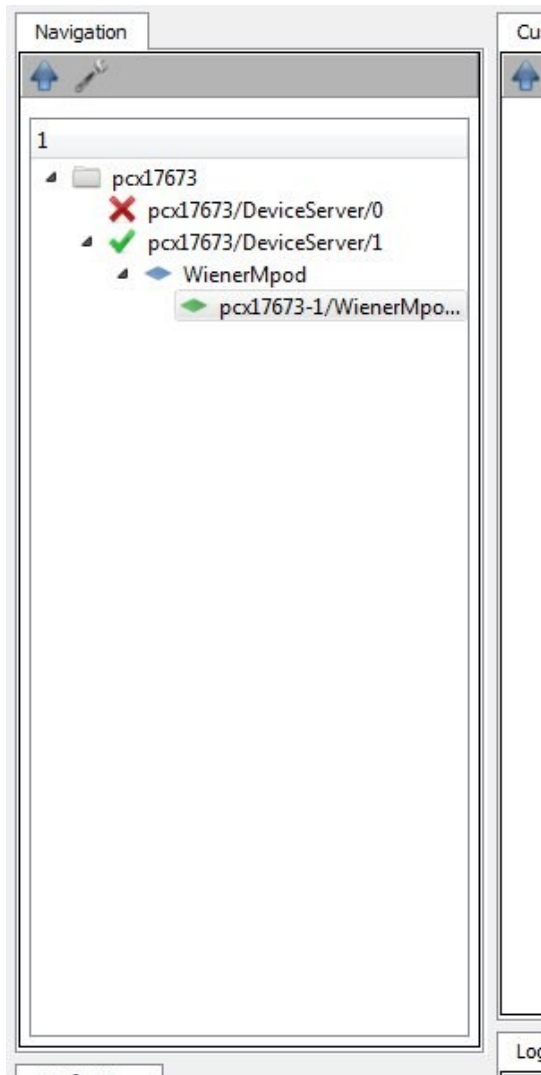
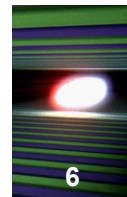
The screenshot displays the 'European XFEL - PyQt GUI Version' interface. The main window is divided into several sections:

- Navigation:** A tree view on the left showing a hierarchy of components: 'pcx17673', 'pcx17673/DeviceServer/0', 'pcx17673/DeviceServer/1', 'WienerMpod', and 'pcx17673-1/WienerMpod/1'.
- Customize composition area:** A large central area for visualizing the system components.
- Configuration:** A panel on the right showing a table of attributes and their current values, with a 'Value' column for manual input.
- Notifications:** A panel at the bottom left for displaying system messages.
- Logging / Scripting console:** A panel at the bottom center for displaying log messages.
- Documentation:** A panel at the bottom right for displaying documentation.

The **Configuration** panel includes a table with the following data:

Attribute	Current value	Value
State	OffState	
Switch On		
Switch Off		
Clear All		
SNMP host	192.168.140.32	192.168.140.32
Slow rate	10000	20000
Fast rate	2000	3000
Crate	0	
Crate system	0	
Crate output	0	
Crate Main Switch	1	1
Crate System Status		
Crate reset	0	0
Output Number	16	

Buttons at the bottom of the configuration panel include 'Kill instance' and 'Apply all'.



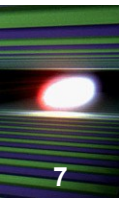
Hierarchical view

- Host (physical)
 - DeviceServerInstance
 - available DeviceClasses (i.e. plugins)
 - DeviceInstances

Features

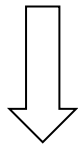
- different filter options
- search for specific instances
- synoptic view (“Google Maps” for device servers)
- ...

Creating devices

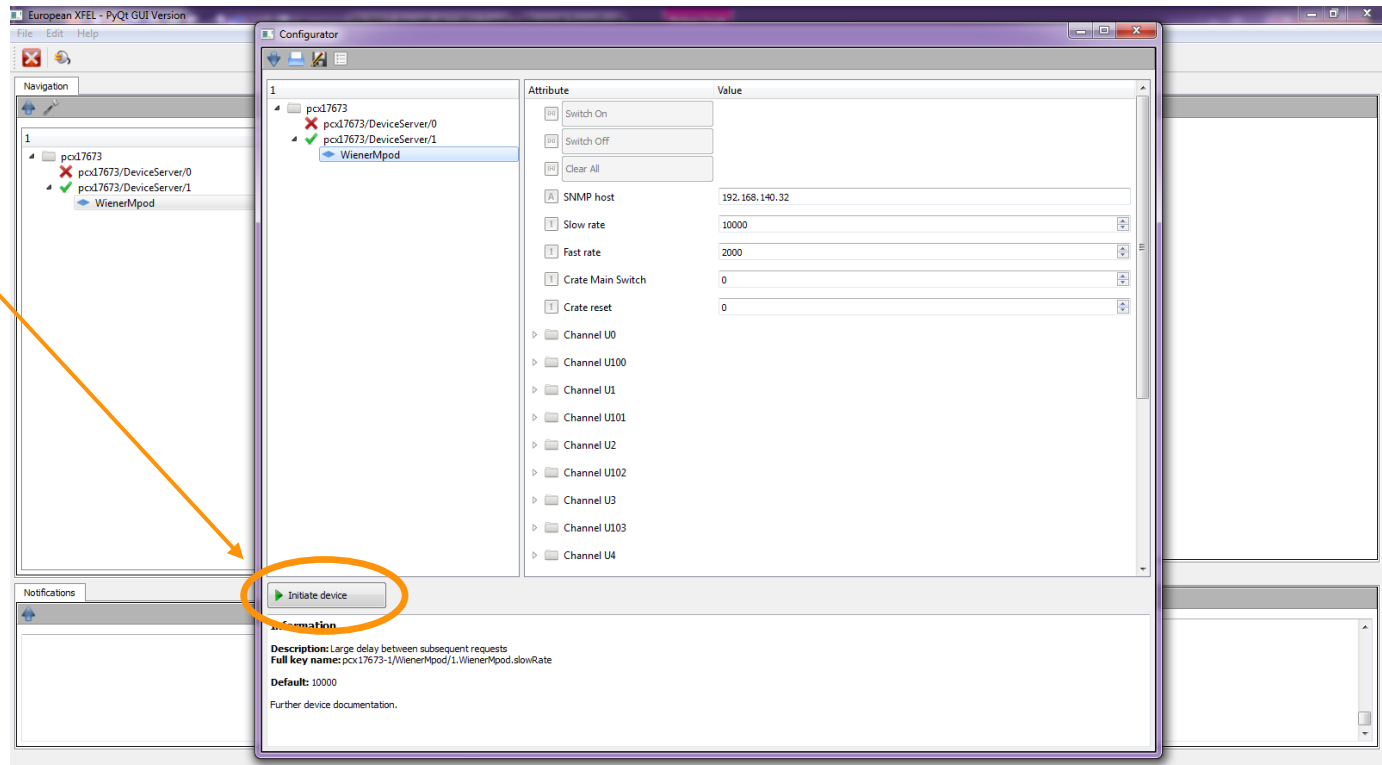


- undock configurator → standalone panel
- no custom coding needed
- automatic generation of configuration masks by using the device's inherent self description (the famous “static expectedParameters”)

Initiate device



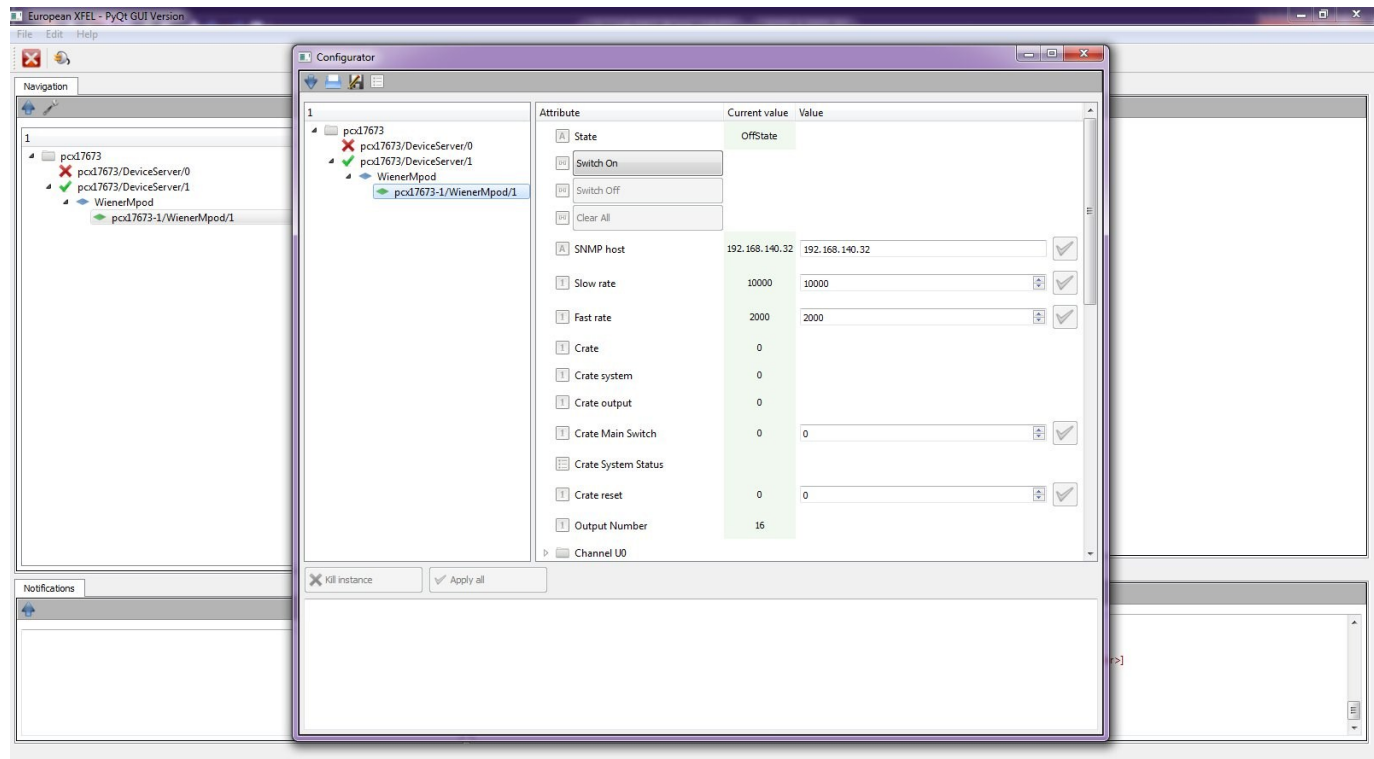
*create instance of this
class (as thread on the
device server)*



Interacting with devices 1/3

8

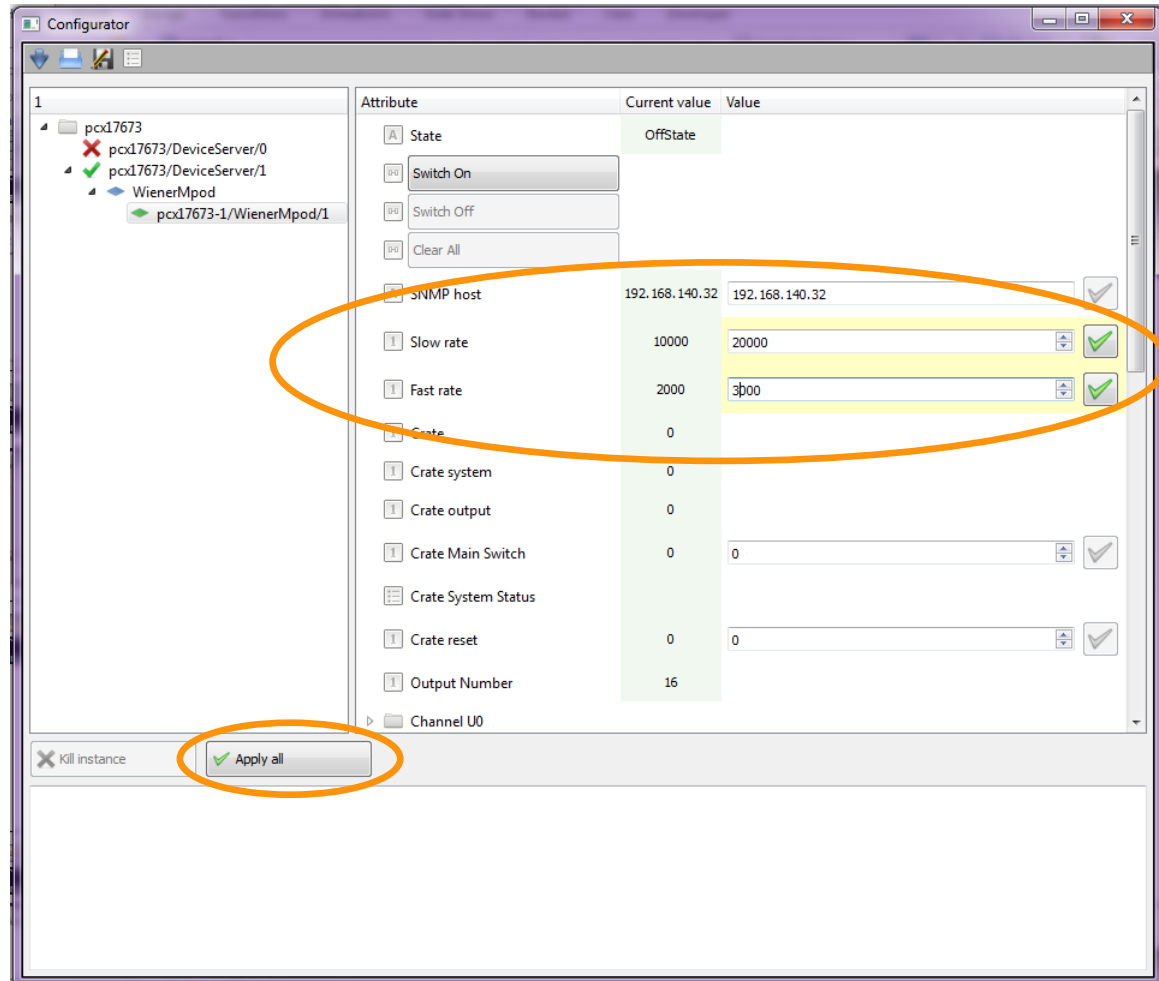
- device instance with attributes:
 - display name
 - current value
 - editable field (reconfiguration)



Interacting with devices 2/3

9

- apply changes:
 - individually
 - selection
 - all in one package
- save/open configuration

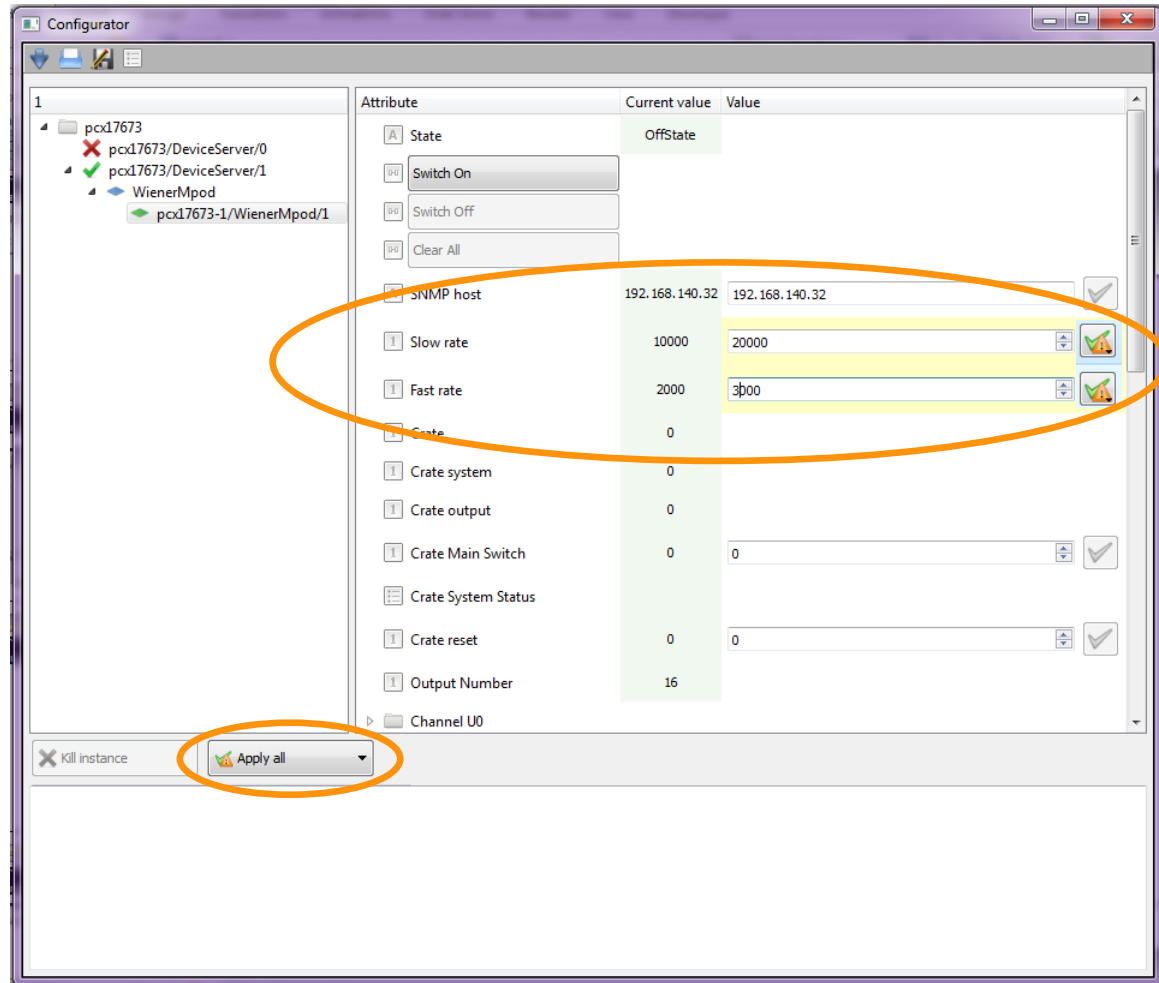


Interacting with devices 3/3

■ changes in distributed systems result to conflicts

■ 2 ways of solving:

1. Accept remote changes
2. Re-Apply local changes



Generic Default GUI Layout (Prototype)

The screenshot displays the 'European XFEL - PyQt GUI Version' window. The main interface is divided into several sections:

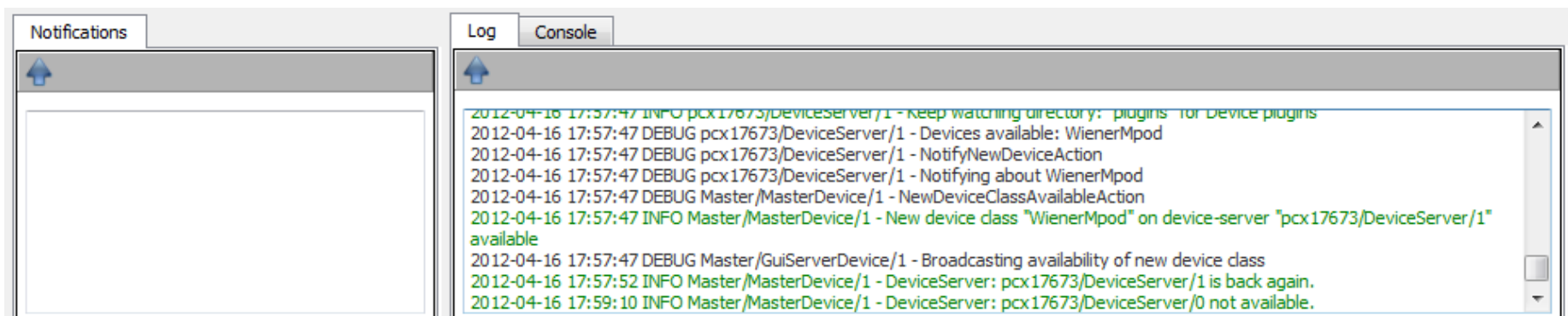
- Navigation:** A tree view on the left showing a hierarchy of components: 'pcx17673', 'pcx17673/DeviceServer/0', 'pcx17673/DeviceServer/1', 'WienerMpod', and 'pcx17673-1/WienerMpod/1'.
- Customize composition area:** A large central white area for visualizing the system layout.
- Configuration:** A panel on the right with a table of attributes and their values, and a set of controls below it.
- Notifications:** A panel at the bottom left for displaying messages.
- Logging / Scripting console:** A panel at the bottom center showing a log of system events.
- Documentation:** A panel at the bottom right for displaying help or documentation.

The **Configuration** panel includes the following table:

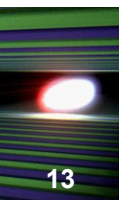
Attribute	Current value	Value
State	OffState	
Switch On		
Switch Off		
Clear All		
SNMP host	192.168.140.32	192.168.140.32
Slow rate	10000	20000
Fast rate	2000	3000
Crate	0	
Crate system	0	
Crate output	0	
Crate Main Switch	1	1
Crate System Status		
Crate reset	0	0
Output Number	16	

Below the table are buttons for 'Kill instance' and 'Apply all'.

- Logging panel
 - status of system
 - sent/received data
- Scripting panel
 - providing iPython interface
- Notifications
 - alarm, error, warning messages

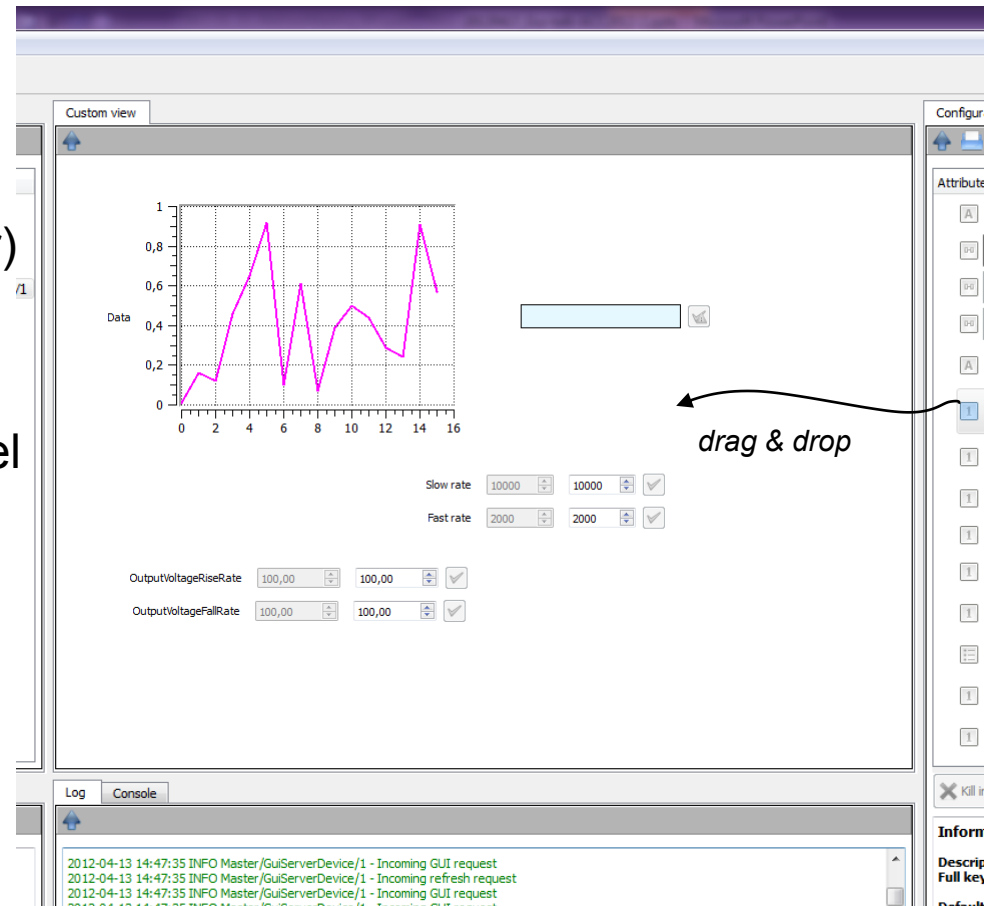


Custom user panels – attribute centric



13

- visual attribute composition across devices
- planned features:
 - drag & drop
 - ➔ “blue print” (bind to instances later)
 - ➔ from running devices
 - powerpoint like look and feel
 - saving and load any customized panel
 - tabbed view or undock feature for multiple panels
 - customizable labels, view widgets, editable widgets
 - ➔ provide defaults
 - ➔ interactive change of widget type
 - ➔ visualization complex attributes (images, arrays, ...)



Custom user panels – device centric

The screenshot displays the European XFEL PyQt GUI Version interface, which is divided into several panels:

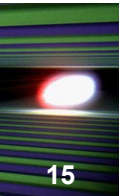
- Navigation:** A tree view on the left showing the device hierarchy: `pcx17673` (expanded) → `pcx17673/DeviceServer/0` (checked) → `pcx17673/DeviceServer/1` (checked) → `WienerMpod` (checked) → `pcx17673-1/WienerMpod/1` (checked).
- Custom view:** A central workspace showing a flowchart diagram. A dashed arrow labeled "drag & drop" points from the `pcx17673-1/WienerMpod/1` node in the Navigation panel to an `AndorCam` node in the Custom view. The flowchart consists of:
 - `AndorCam` (orange box) → `Correct` (orange box) → `Sum` (orange box)
 - `AndorCam` (orange box) → `Correct` (orange box) → `Visualize` (purple box)
- Configurator:** A panel on the right showing a table of attributes and their current values, along with configuration options.
- Notifications:** A panel at the bottom left showing a list of system messages.
- Log Console:** A panel at the bottom center showing a log of system events.

Configurator Table:

Attribute	Current value	Value
State	OffState	
Switch On		
Switch Off		
Clear All		
SNMP host	192.168.140.32	192.168.140.32
Slow rate	10000	10000
Fast rate	2000	2000
Crate	0	
Crate system	0	
Crate output	0	
Crate Main Switch	1	1
Crate System Status		
Crate reset	0	0
Output Number	16	

Information:

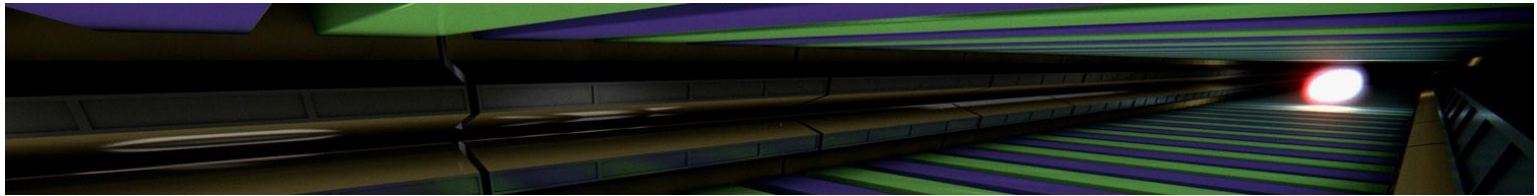
Description: Large delay between subsequent requests
Full key name: `pcx17673-1/WienerMpod/1.WienerMpod.slowRate`
Default: 10000
Further device documentation.



- most requirements full-filled:
 - generic, multi-purpose, cross-platform GUI
 - remote control
 - self explaining, default layout for any device
 - highly customizable user panels

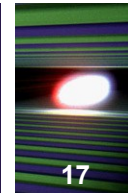
Further concepts currently developed.

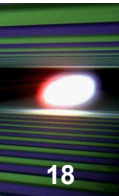
- FSM visualization
- user-role driven (database)



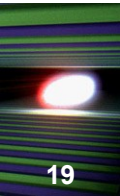
Thanks for your attention! 😊

Additional Slides





- save/open customized panels and latest composition/view of user interface
- showing attributes in different ways, e.g. array of values as plot or comma separated
- different filter options for navigation bar (hierarchical, all device servers, all device classes, ...)
- integration of iPython console
- visualization of complex attributes (plots, images, ...)
- integration of 3rd party widgets (Taurus, PyQwt, ...)
- FSM visualization
- user-role driven (database)



- PyQt (platform independent)
- Signals/Slots
- client/server architecture
- generic GUI (configure devices, XSD, XML)
- customized GUI panels:
 - plugin widgets
 - 3rd party widgets (Taurus, ...)
- design requirements:
 - save/open configurations (XML)
 - immediate notice of changes
 - customize GUI in a centralized panel
 - data traffic only for visible GUI items

Mainframe Status

Channel	Voltage	Current	Measured Sense Voltage
U 0	0 V	0 A	187.32 mV
U 1	0 V	0 A	0 V
U 2	0 V	0 A	148.82 mV
U 3	0 V	0 A	429.14 mV
U 4	0 V	0 A	375.39 mV
U 5	0 V	0 A	0 V
U 6	100.00 V	0 A	100.58 V
U 7	150.00 V	0 A	150.04 V
U100	4000.0 mV	0 A	0 V
U101	4000.0 mV	0 A	0 V
U102	4000.0 mV	0 A	0 V
U103	4000.0 mV	0 A	0 V
U104	4000.0 mV	0 A	0 V
U105	4000.0 mV	0 A	0 V
U106	4000.0 mV	0 A	0 V
U107	4000.0 mV	0 A	0 V

Mainframe Status

Channel	Voltage	Current	Measured Sense Voltage
U 0	0 V	2000.0 uA	0 V
U 1	0 V	2000.0 uA	0 V
U 2	0 V	2000.0 uA	0 V
U 3	0 V	2000.0 uA	0 V
U 4	0 V	2000.0 uA	0 V
U 5	0 V	2000.0 uA	0 V
U 6	0 V	2000.0 uA	0 V
U 7	200.00 V	2000.0 uA	200.00 V
U 8	0 V	2000.0 uA	136.42 mV
U 9	0 V	2000.0 uA	116.65 mV
U 10	0 V	2000.0 uA	127.00 mV
U 11	125.00 V	2000.0 uA	125.01 V
U 12	0 V	2000.0 uA	138.25 mV
U 13	180.00 V	2000.0 uA	180.02 V
U 14	155.00 V	2000.0 uA	155.01 V
U 15	0 V	2000.0 uA	151.64 mV

Waiting for 192.168.2.25...

Configurator

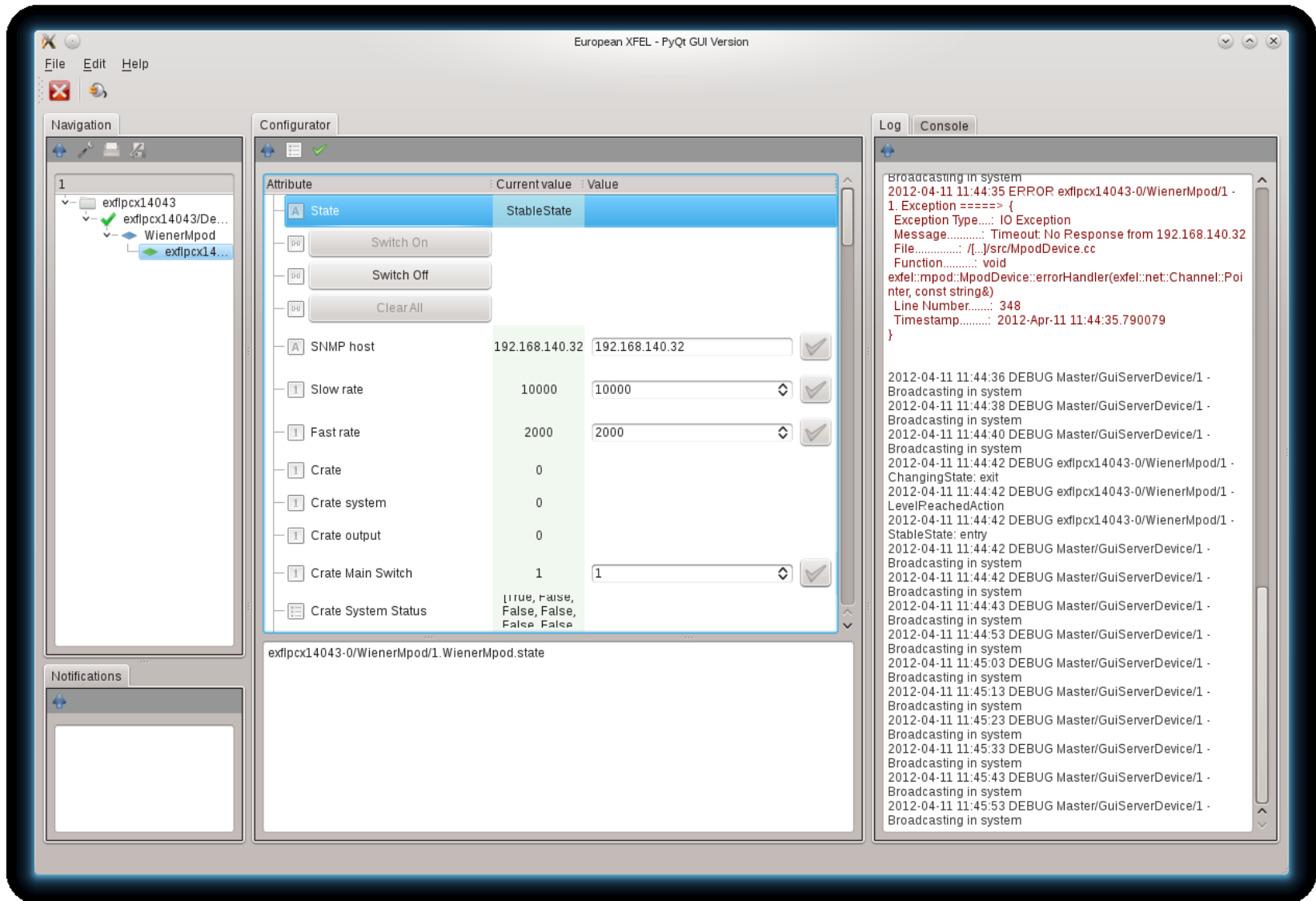
Attribute	Current value	Value
State	StableState	
Switch On		
Switch Off		
Clear All		
SNMP host	192.168.2.25	192.168.2.25
Slow rate	10000	10000
Fast rate	2000	2000
Crate	0	
Crate system	0	
Crate output	0	
Crate Main Switch	1	1
Crate System Status		

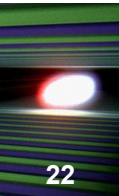
Log

```

2012-04-13 18:26:07 DEBUG Master/GUI/ServerDevice/1 - Broadcasting in system
2012-04-13 18:26:07 DEBUG exiftd14-0/WienerMpd/1 - ChangingState: exit
2012-04-13 18:26:07 DEBUG exiftd14-0/WienerMpd/1 - LevelReachedAction
2012-04-13 18:26:08 DEBUG exiftd14-0/WienerMpd/1 - StableState: entry
2012-04-13 18:26:09 DEBUG Master/GUI/ServerDevice/1 - Broadcasting in system
2012-04-13 18:26:11 DEBUG Master/GUI/ServerDevice/1 - Broadcasting in system
2012-04-13 18:26:13 DEBUG Master/GUI/ServerDevice/1 - Broadcasting in system
2012-04-13 18:26:15 DEBUG Master/GUI/ServerDevice/1 - Broadcasting in system
  
```

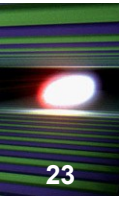
More Screenshots





- Standardized XSD and XML representations of any device
→ generic configuration and control
- distinction between configuration for device construction (XSD + XML) and (re-)configuration of an existing device instance (XML)
- both can be saved/opened for reusage

Connecting the bits and pieces



- Python bindings:
 - PyQt (platform independent)
 - homogeneous in-house framework