Gui-Building

Goals and how to reach them

A Case Study of *Goal-Oriented Gui-Building*:

The Technical Subsystem ***XFEL HPRF***

Mark Lomperski MIN

13.Nov.2019

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

DESY.

Gui Building: Agenda

01 The XFEL HPRF Modulators
- Some Background Info

02 Goals
- Use Case: *Technical Subsystem*
- Priorities (HPRF, *mine*: ex-BKR; an opinionated consultant)

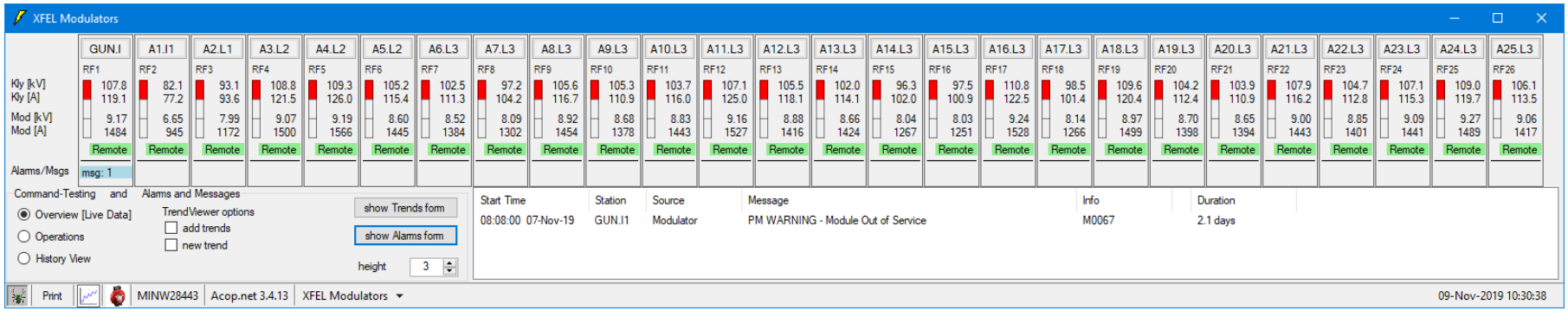
03 Control System Gui Functionality
- Control
- Monitoring *a Technical System*
  - Data are Interesting when:
    - **A Failure / Trip**
    - Running but ***out of tolerance***
  - **Fore**ground (users)
    - A.k.a "Live Data"
  - **Back**ground (servers)
    - A.k.a. "Archiving"

04 Building the Gui
- Tools: VS and ACOP.NET
  - Program a Rich Client
  - ***Gui's*** in the ACOP.NET Status Bar
    - The Spider
    - The Trend Viewer
    - The Alarms Viewer
- Foreground Monitor-Data
  - Organizing the data flow
  - Overview (single values)
  - Selected Devices (traces)
- Background Monitor-Data
  - Trip Events are of greatest interest
  - Integrating System Services in Display
    - Alarm System and the AlarmsViewer
    - Event Archiver

05 Summary and Conclusions

06 Demonstration (Clicking around a bit)

The XFEL Modulator Gui: revisited

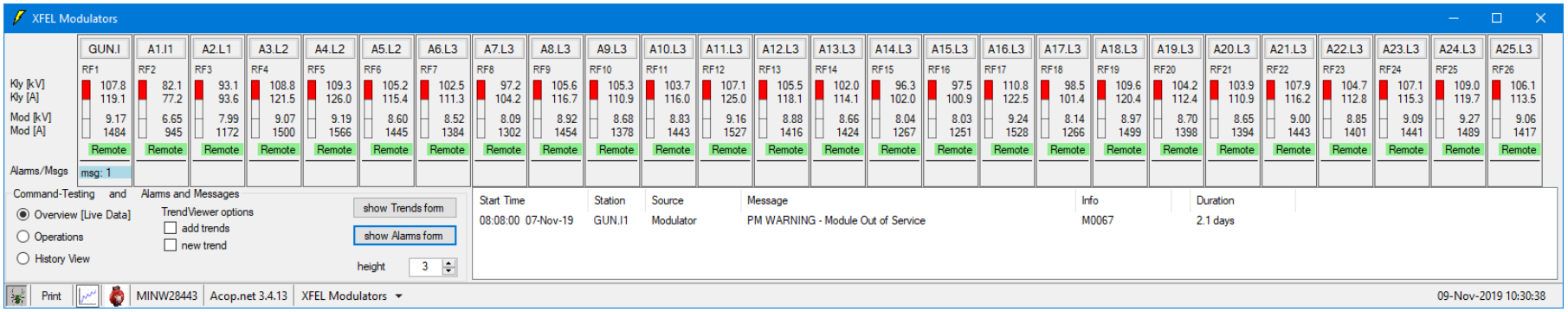


The XFEL HPRF Modulators

- High Power RF pulsed at 10 Hz
- 26 “devices” - Ampegon front-ends
- Details: *nearly* identical devices
- Standard Operating Gui: written in jddd by Torsten Grevsmuehl
- Set higher goals: try to reach using Visual Studio, TINE and ACOP.NET

The Target Group / Customers

- The members of the HPRF group
- Focus on day-to-day monitoring issues, not on Control
- This Gui is a good, uebersichtliche example of ***Setting your Goals*** and ***Reaching Them.***

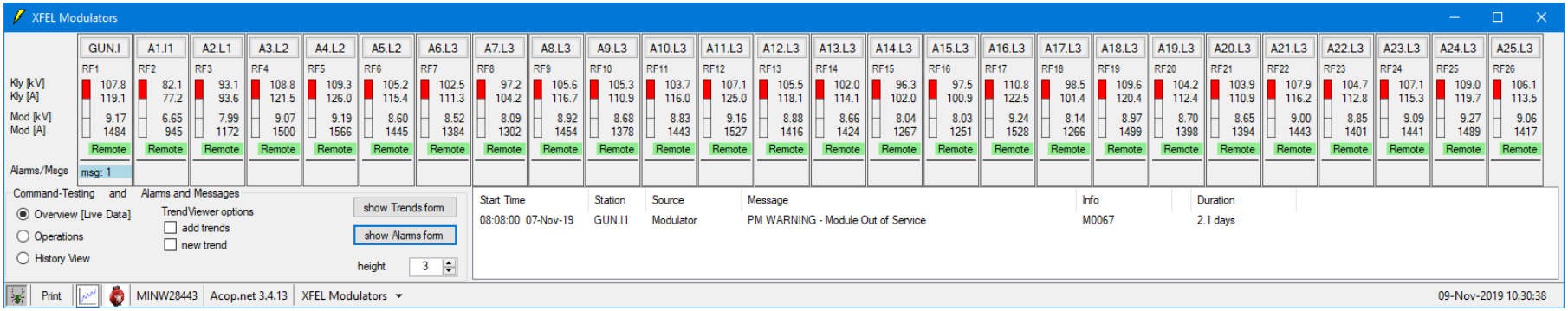


The XFEL Modulator Gui: Perspective, Goals

- **General Goals: My Perspective, World View, Design Principles**
  - ***One-Stop-Shopping*** : User can find all required information without needing to start other applications
  - ***Cut-to-the-Chase*** : (i.e. *get to the point*)
    - Show clearly what is needed, everything else “out of the way”

Basic Idea: Invest ***my*** time with the goal to save the time of the ***Users*** (the HPRF Group).

Requirements, Needs, Wishes for Functionality” – Experience, from Customer Interviews



The Gui: Functionality Requirements

- Control
  - A Light Bulb (with dimmer switch)
  - The Front-End controls the pulsed MW
- Monitoring: *Foreground* “Live Data”
  - Data-Transport @10Hz, Browsing Issues
  - Overview: 4 floats + 4 Integers per Station
    - 8 values * 26 stations = 208 per update
  - Selected Device: Traces (2000 floats)
    - Modulator: 4 RF: 8
  - Status Bits:
    - Modulator ~150 bits
    - RF-Interlock: ~150 bits
  - Control System Status Information
- Monitoring: *Background*
- A Technical System
  - does not require e.g. Data Collection for Number-Crunching for On- or Off-line Analysis

If the System is
Running and within Tolerances
then
Data Storage not necessary/of much interest
Else
Store ***Status*** and ***Traces: for Detective Work***

- Need to Store
  - ***Out of Tolerance*** Pulses
  - Status i.e. Interlock Information
- Present this information clearly

Reaching Goals: *Configure or Program?*

New Tricks, but I'm an Old Dog with *Principles*!

My Tool-Kit of Choice: Visual Studio VB.NET, ACOP.NET and TINE.

Try Something New: "Configure" the Gui

- Use ACOP ***Smart*** Graphical Components [Labels connected to a TINE Address]
- Simplify (?) Gui-Building by configuring - not programming? Give it a try, see how far I can go
- But: my Starting Point putting together a Gui:

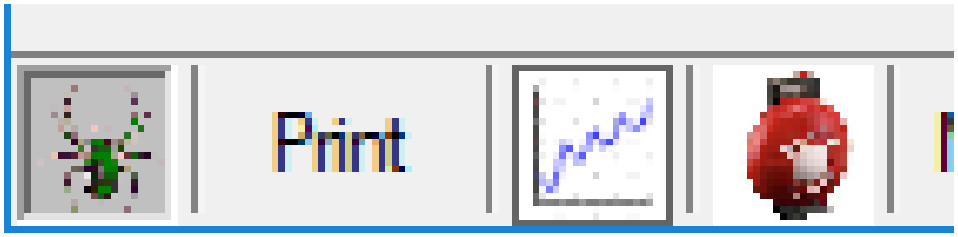
  The Data-Flow from Control System Servers
- I have been doing this too long: not yet ready to give up *all* responsibility to the "engine" running "under the hood"

My Strategy: Stay with *Programming* in Visual Studio VB.NET

My Message: Using these tools, it is easier than ever to reach the goals.

Reaching Goals: *The ACOP.NET StatusBar*

Rich Client **Functionality** which you *need*, but don't need to do yourself!



The Spider

- Status of Control System Activity
  - A clear and simple view (easier said than done)
  - Debug, remote access to Server
- Life without the Spider? Does anyone remember *Life **before** the Spider ?*
- ACOP ***Smart*** components "know" how to show their link-status
- The Spider's got new Functionality!

The Alarms Viewer: *A Game Changer*

- Access *from the Gui* to the Local- and Central Alarm Servers!
- ***Start Times*** and ***Durations*** of Status-Bit (a.k.a. Alarms) Changes!
  - Current Status shown in the Gui
  - Provides a Quick view of the timeline of Status Changes
  - An Archive-Viewer of Status-Changes/Alarms!
- Just the Basics of the General Purpose Alarm Viewer (*Cut-to-the-Chase*)

The Trend Viewer

- Gui integrates Central Archiver and Local History Data Viewing into the Gui
- Just the Basics of the General Purpose Archive Viewer (*Cut-to-the-Chase*)

Monitoring in the Foreground: Efficiency

With the help of *The Combobulator :* a System-Standard Middle-Layer

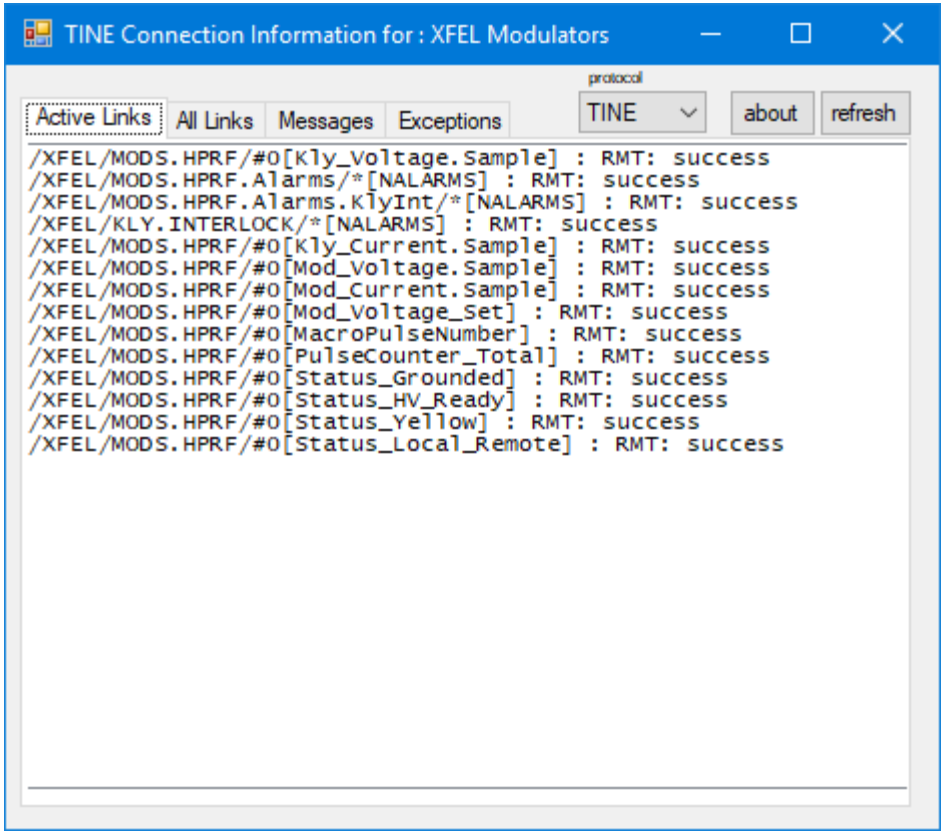
Constructing Multi-Channel Arrays

- 26 F.End Servers with identical Props
- **Configure** a Combobulator to collect and “re-export” data as MCA Arrays
- Clients [Archiver, all Gui’s] connect to Combobulator for data

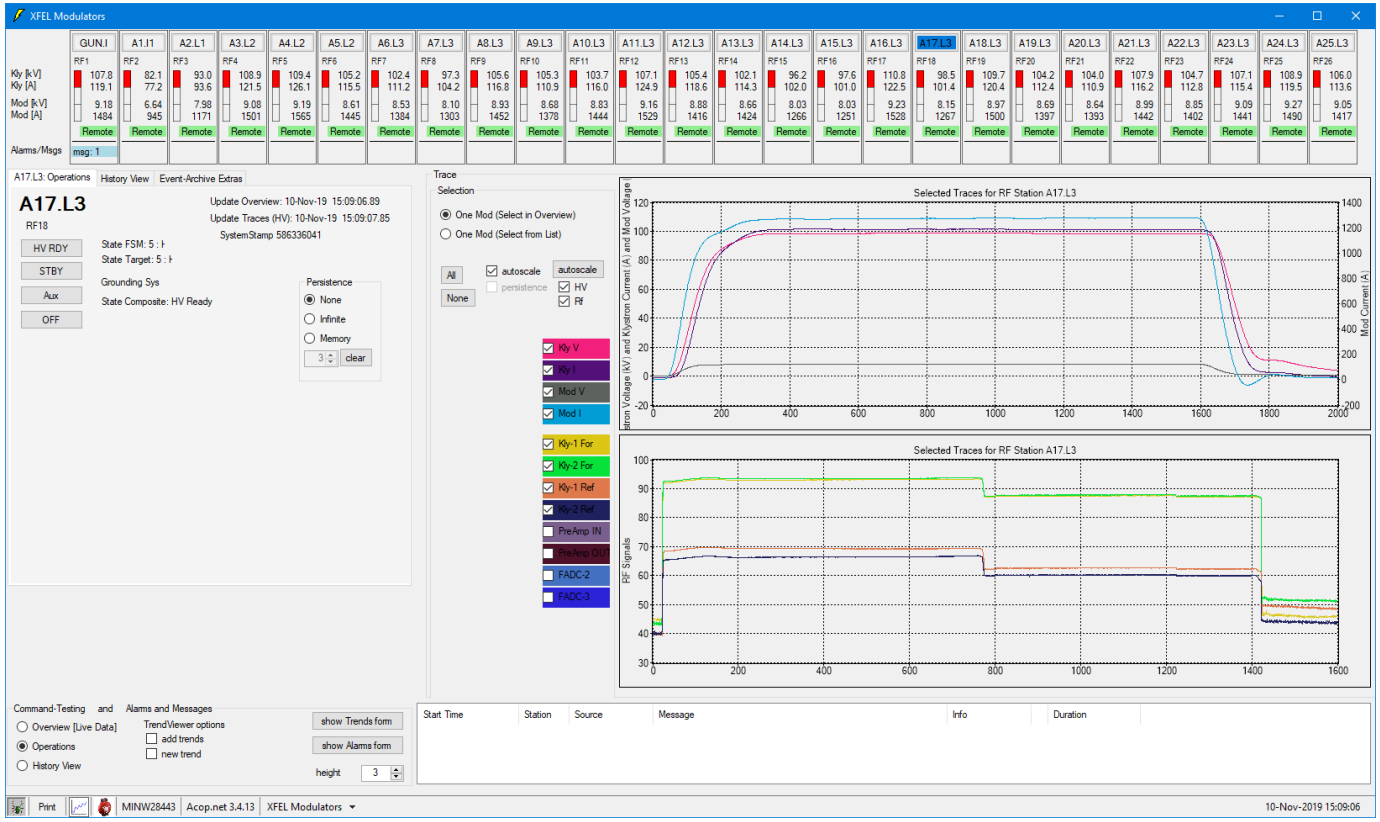
- The Gui has 8 Data Links (grouped!) instead of 208
- “support” services: Status and Quality Statistics of Links, Local Histories, …

- “Plug [configure] and Play” !

- Helps you reach Goals for Efficiency
  - Leaves a Small “Foot Print” on the Front Ends



Monitoring in the Foreground: Browsing



Displaying Traces / Pulse Data from the Modulator and the RF-Interlock System

- Select a device to view the corresponding traces
- The General-Purpose ***Scope Trace Viewer*** could be used to compare selected trace properties from selected stations
- But providing a simplified, limited browsing saves time for the User!
- Cut-to-the-Chase [flexible trace selection] and One-Stop-Shopping [without the STV] !

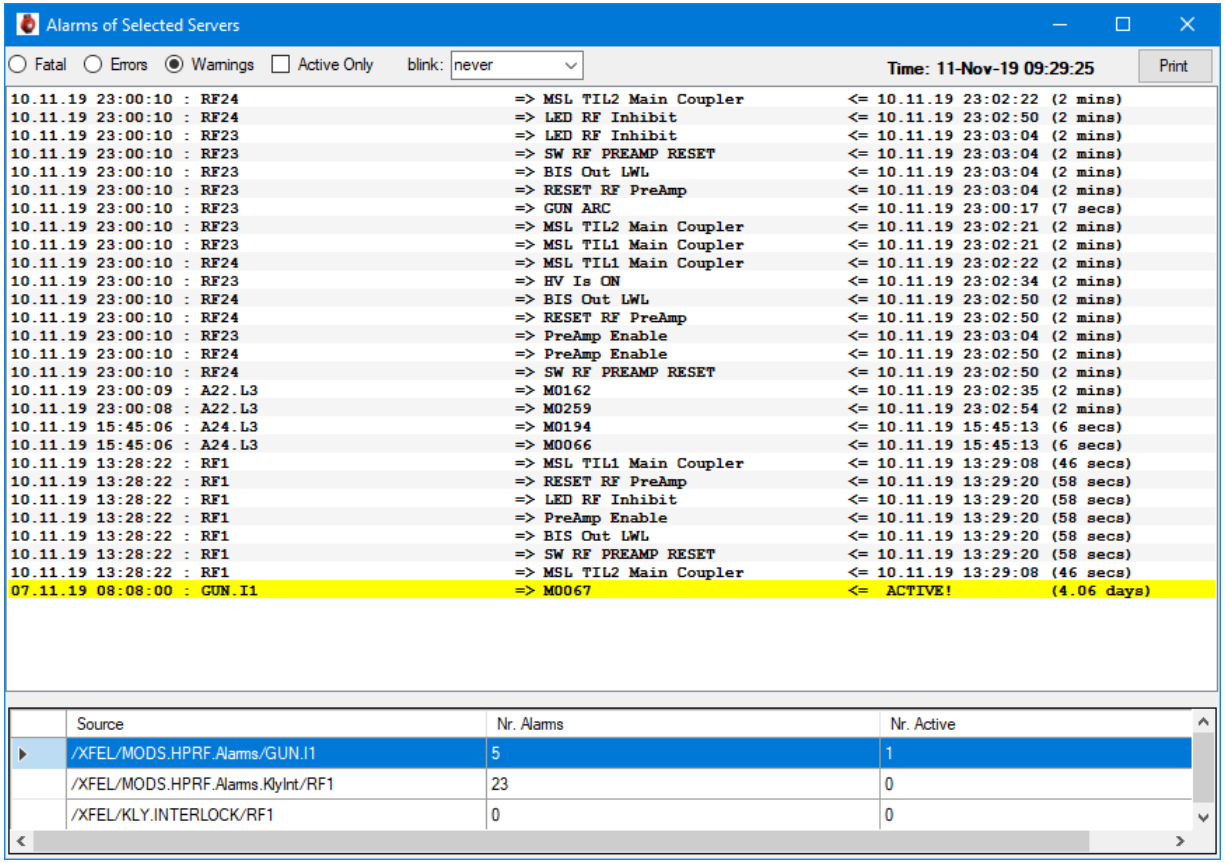
Monitoring Status Info: The Alarms Viewer

Click on the Icon:
Open a Gui for the Alarm-System

Active Alarm from 26 Front-End Servers: started 4 days ago!

“Status Changes” earlier in the day – from various RF Stations

Without coding anything!



Most Operating Programs do not have a “History” Mode. This is a Rich Client which provides a general view of **live *and history*** of alarms!
Operations in BKR: *Status is red? Now Green? When did that happen? How long was it red?*
The other Dingie is also Red? Which went South first?
Full Information available in CAS Viewer – but Cut-to-the-Chase!

Maintenance of the Alarm System: Helps CS Bozos integrate Status-Information into CAS

Monitoring: Trends with the TrendViewer

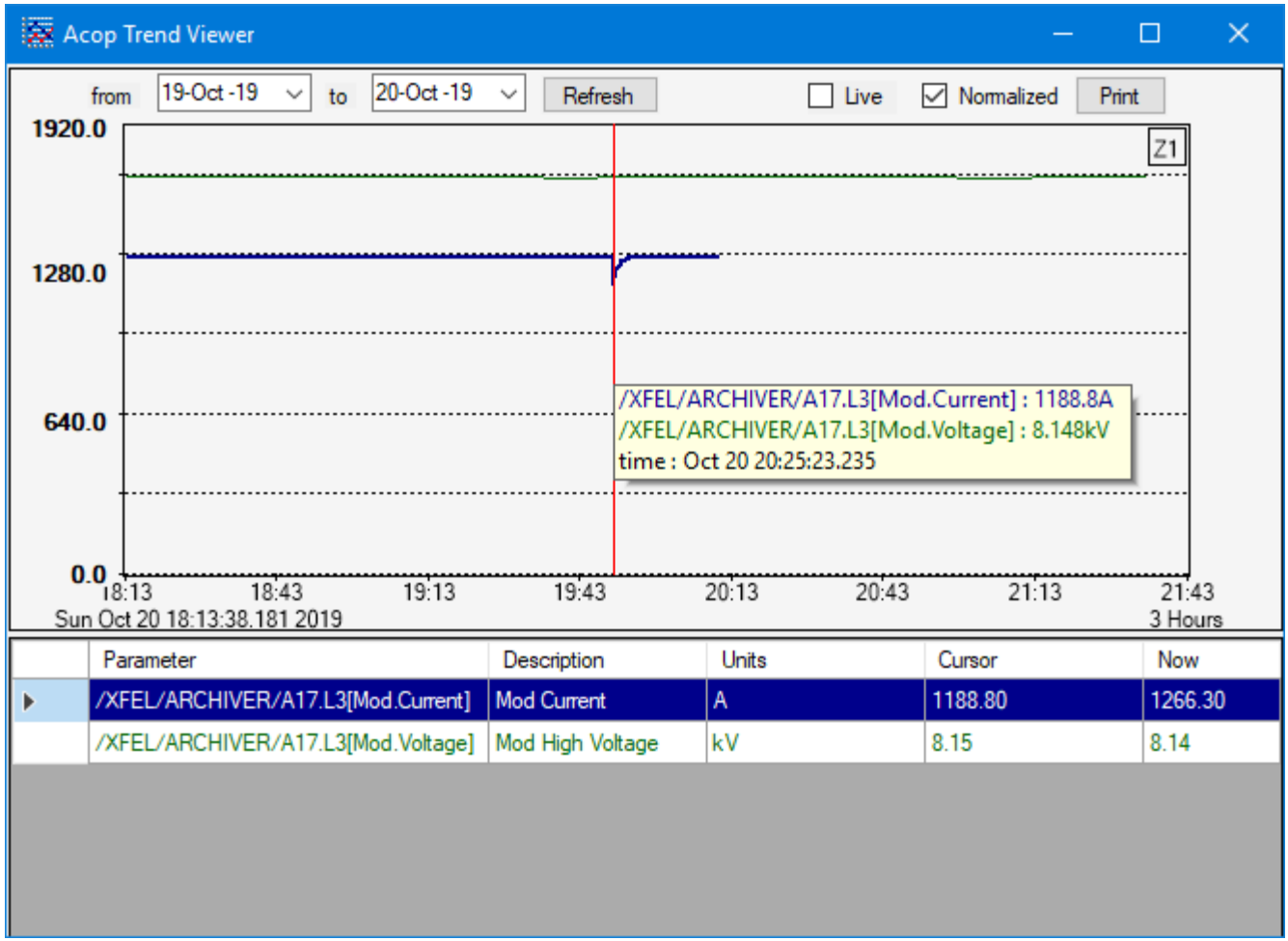
Click on the Icon
Open a gui for the Central and Local Histories

Drag and Drop onto form to select Properties

Assist Browsing with Coding in Gui

I planned on a consistency check:
View a Station-Trip with
- Status Info
- Pulse Data
- Trends

But Trends not (yet?) interesting in the HPRF Use-Case



Monitoring and Displaying Status Information

How are status information stored? Different Strategies…

01 Status Information as Integers

- Read-out of Hardware as 32-Bit Integers
- Packed into Integers for convenience

02 The Strategy used in the Current Gui on:

- Trip of RF Station **A17.L3** on **20.Oct.2019**

03 Display *Foreground* Values (Live)

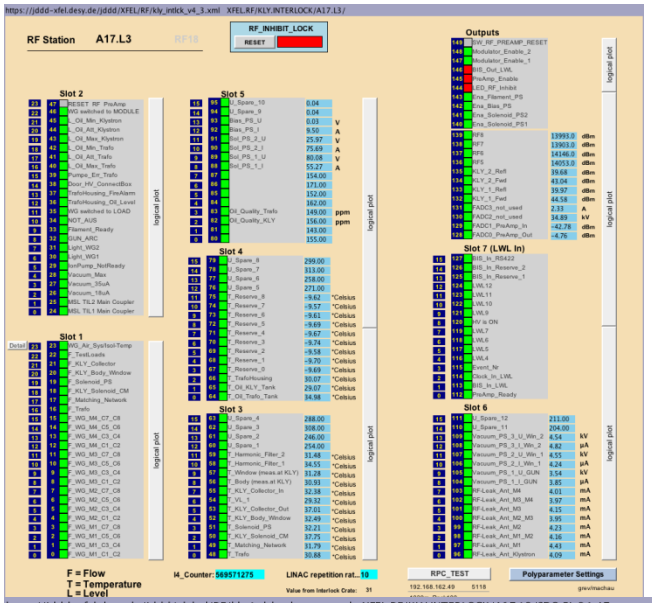
- Current Status: Bit Value: 1/0 Red/Green
- RF-Interlock Info but not Modulator Info

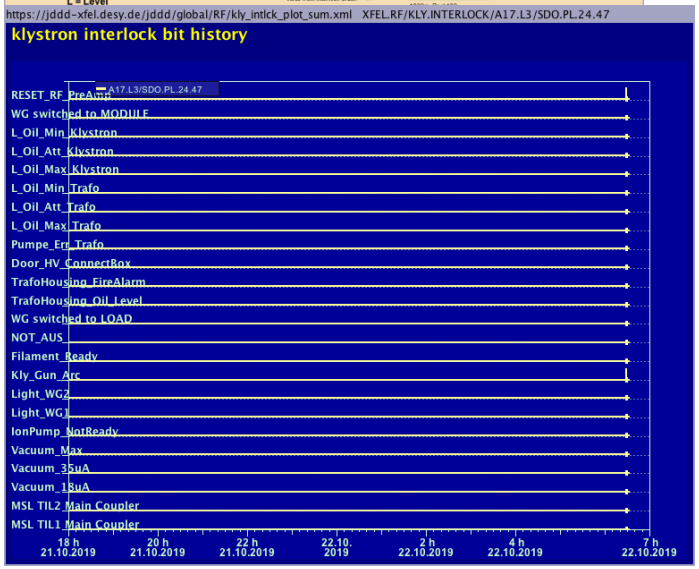
04 Display *Background* Values (Archived)

- Histories of the values of each bit of the selected Status-Word.

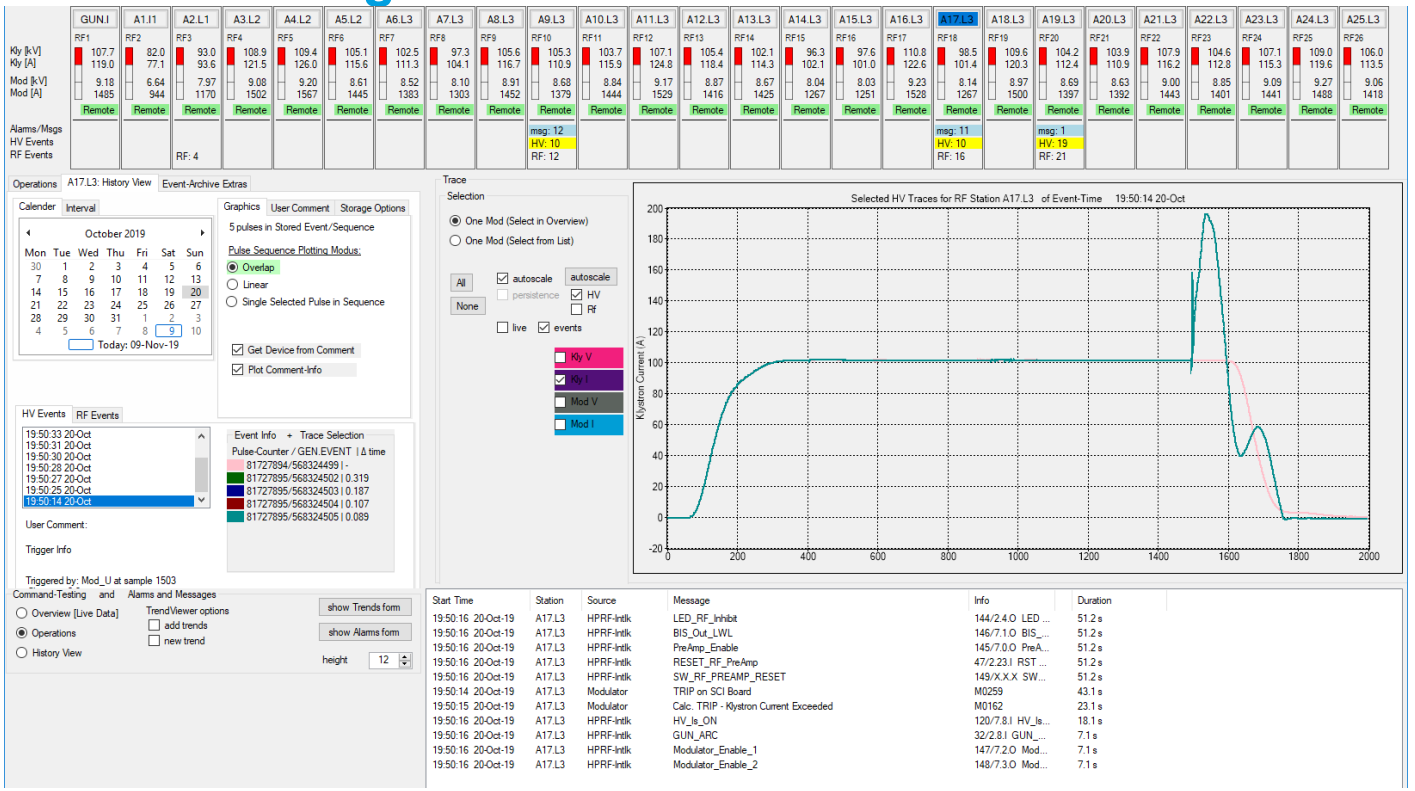
05 Documentation

- Screenshots posted in the XFEL Logbook as documentation of the Trip-Event





Display of Background Monitoring Info from the Alarm- and the Event-Archive Systems.



- **Status Changes** are stored as *alarms* [not integers] in the Alarm System with a **Start-Time** and a **Duration**: just the Information needed!
  - View with Alarm Viewer.
  - Game Change: Easy to integrate the Alarm Data into a customized Alarm-View!
- **Out-of-Tolerance Traces** stored in the Event Archiver: just the traces needed!
  - View with Event-Archive Viewer
  - Easy to integrate the Event Data into a customized Event-Viewer!
- *Cut-to-the-Chase* [CAS and Events] and *One-Stop-Shopping* [integrating in Gui] !

Summary and Conclusions

01 Revisiting the XFEL HPRF Gui: The Strategy

- *One-Stop-Shopping* and *Cut-to-the-Chase*
  - Provide access to all data needed for Subsystem Monitoring
  - Present in an easy-to-digest fashion.
- Utilize the Alarm System and the Event Archiving System
  - Foundation: Connect the HPRF Servers to these Central Services
  - Integrate these Services into the Gui

02 Tool-Kit of Choice: Rich Client in VS 2015 ACOP.NET TINE

- Goals and Functionality not new
  - Requirements of pulsed HPRF haven't changed (TTF)
- But the Tools in the Tool-Kit are better than ever!
  - Combobulator to easily build MCA Arrays for efficient transport/browsing
  - Can do a lot with the ACOP Smart Graphical Elements
  - Easily integrate Alarm- and Event Archive- Systems in the Gui
    - Rich client programming to tailor to the use case

3 Status

- Gui is still a Baustelle, but…
- Well on the way to successfully fulfilling the Goals!
- I am working on getting the AlarmsViewer in all my Rich-Client Programs

Contact

DESY. Deutsches
Elektronen-Synchrotron

www.desy.de

Mark Lomperski
Department MIN
Mark.Lomperski@desy.de
Phone 00 49 40 8998 2088