

INTENSE Monthly Meeting - Nov/2022

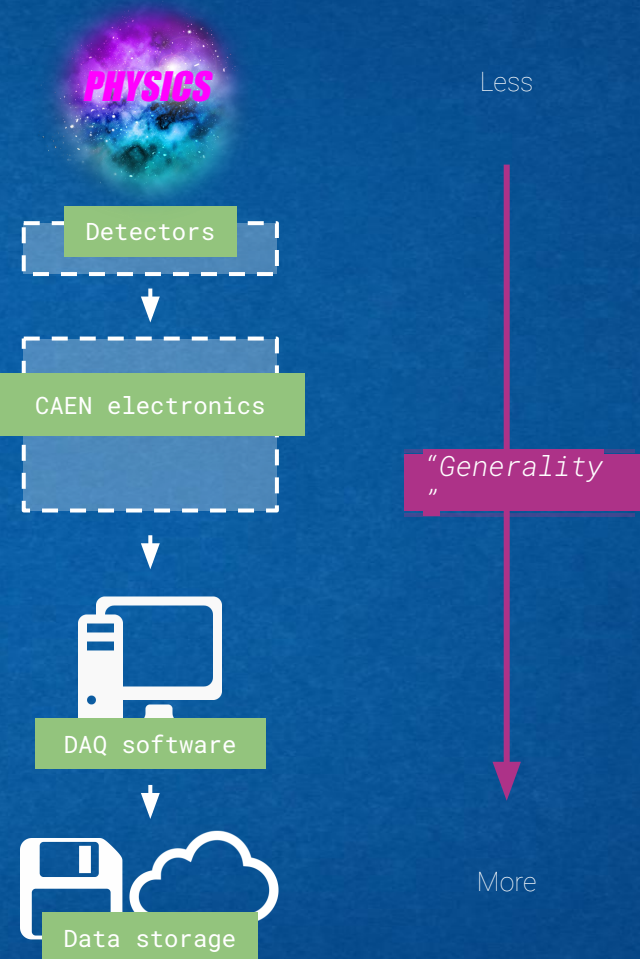
Development of a data acquisition platform based on CAEN digital electronics

Matías Simonetto



Data acquisition platform

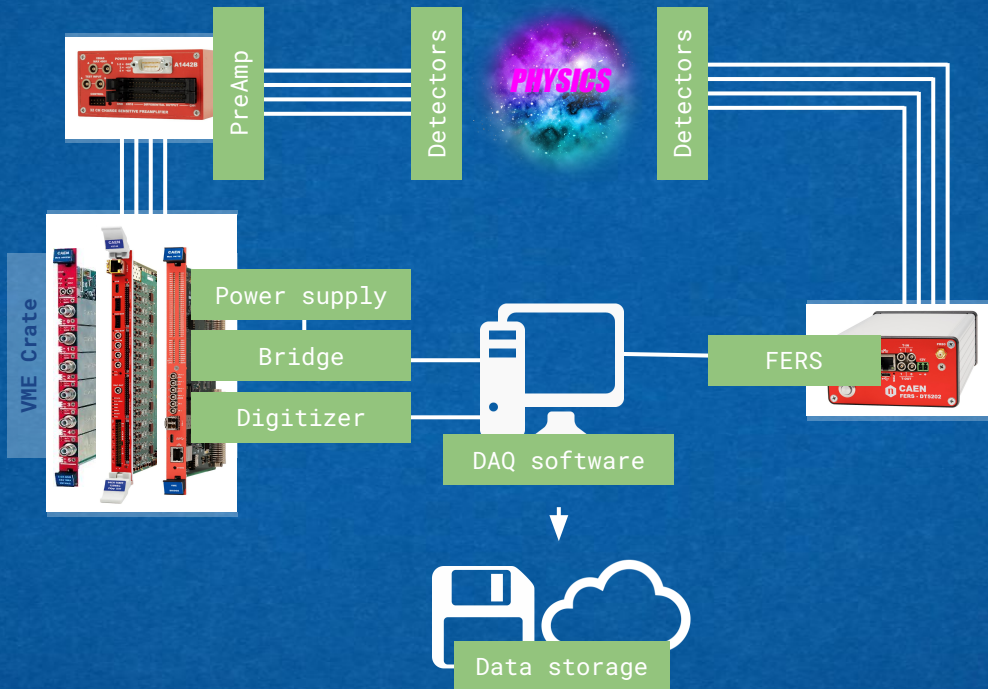
- From detectors to data storage.
- CAEN electronics
 - Power supply.
 - Signal conditioning.
 - Digitalization.
 - Communication.
- DAQ software
 - Device configuration and control.
 - Data readout and storage (eventually in a cloud database).
 - Integrated, versatile, high performance and easy-to-use.



< Digitizer-based

ASIC-based >

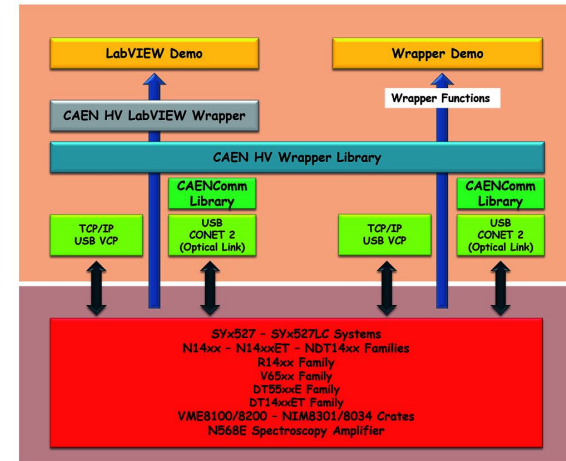
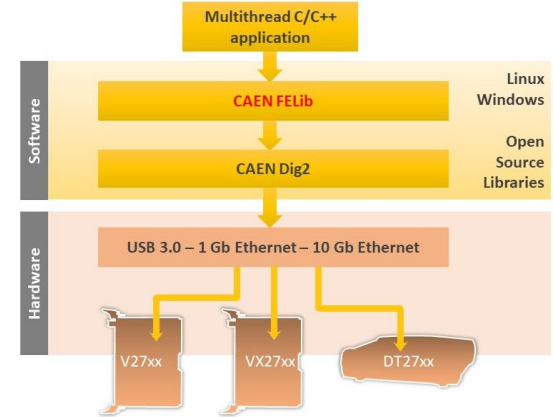
- *Signal conditioning*
CAEN A1442
16/32 Channel charge sensitive preamplifier.
- *Power supply*
CAEN V6519
6 Channel 500 V/3 mA VME
- *Communication*
CAEN V4718
VME to USB 3.0/Ethernet/Optical Link Bridge
- *Digitizer*
CAEN V2740
64 Channel 16 bit 125 MS/s.



- **CAEN FERS 5202: Front-End Readout System**
 - Citiroc 1A 32-channel front-end ASIC (x2), working in conjunction with a ADC.
 - Onboard power supply: CAEN A7585D +85 V/10 mA.
 - Several communication interfaces: USB, Ethernet and TDlink.

DAQ software

- Current CAEN GUI softwares
 - Geco, Compass, WaveDump, Janus.
 - Communication (device control and data readout) in a simple and complete way with the *different* components of an acquisition system.
- CAEN intermediate level libraries
 - FELib library, HV Wrapper Library, FERSLib.
 - Easy development of application softwares.



Qt Framework: overview

Your Qt Application/Device

Languages

C++

Python

Qt QML

Javascript

Framework

Qt Libraries Add-Ons

Qt Libraries Essentials

Target Platforms

Desktop & Mobile

Embedded & RTOS

WebAssembly

Embedded Tools

Dev Tools

IDE & Tooling

User Interface
Tools

Design Tools

Digital Content
Creation
Tool Integration

Qt Framework: small example

```

1  #ifndef INCLUDE_DEV
2  #define INCLUDE_DEV
3
4  #include <QObject>
5
6  #include "utils.hpp"
7
8  class Dev : public QObject
9  {
10     Q_OBJECT
11     public:
12     void longOperation()
13     {
14         CUR_QTHREAD_OUT << "processing...";
15         emit processing();
16
17         // Slow operation...
18         QThread::currentThread()->sleep(2);
19
20         QString a = "42";
21         CUR_QTHREAD_OUT << "finished... answer is " << a;
22         emit finished(a);
23     }
24
25     signals:
26     void processing();
27     void finished(const QString& a);
28 };
29
30 #endif

```

```

1  #include <CoreApplication>
2  #include <QThread>
3
4  #include "dev_qt.hpp"
5  #include "utils.hpp"
6
7  int main(int argc, char* argv[])
8  {
9     QCoreApplication ap(argc, argv);
10
11     QThread t;
12     Dev d;
13     t.start();
14     d.moveToThread(&t);
15
16     QObject::invokeMethod(&d, &Dev::longOperation);
17
18     QObject::connect(&d, &Dev::processing, &t,
19         [] {
20             CUR_QTHREAD_OUT << "ok... I'll do other things in the meantime...";
21             QThread::currentThread()->sleep(1);
22         });
23
24     QObject::connect(&d, &Dev::finished, &t,
25         [&ap, &t](const QString& a) {
26             CUR_QTHREAD_OUT << a << "??";
27             t.quit();
28             t.wait();
29             ap.quit();
30         });
31
32     ap.exec();
33 }

```

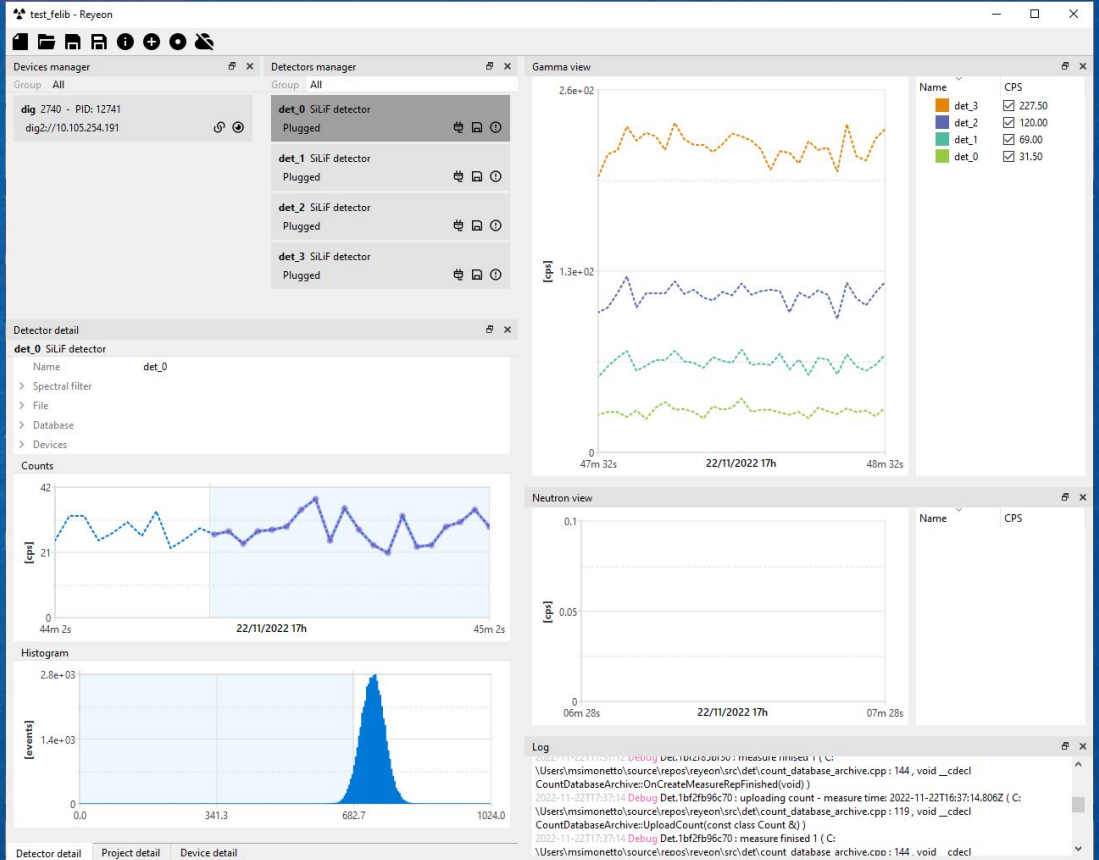
```

[ QThread(0xd265aff7f8) ] processing...
[ QThread(0x25897889a10) ] ok... I'll do other things in the meantime...
[ QThread(0xd265aff7f8) ] finished... answer is "42"
[ QThread(0x25897889a10) ] "42" ??

```

New DAQ software

- **Modular design**
 - Detectors and devices of different type can be easily added/removed.
- **Device management**
 - All devices of the platform can be configured and controlled from within the software. No need of additional programs.
 - Configurations are saved and properly reapplied on each run.
- **Detector management**
 - Simple and clear identification of the detectors and their relations with the devices.
 - Straightforward visualization and saving of the read data.



(Previous) Future work

Bug fix

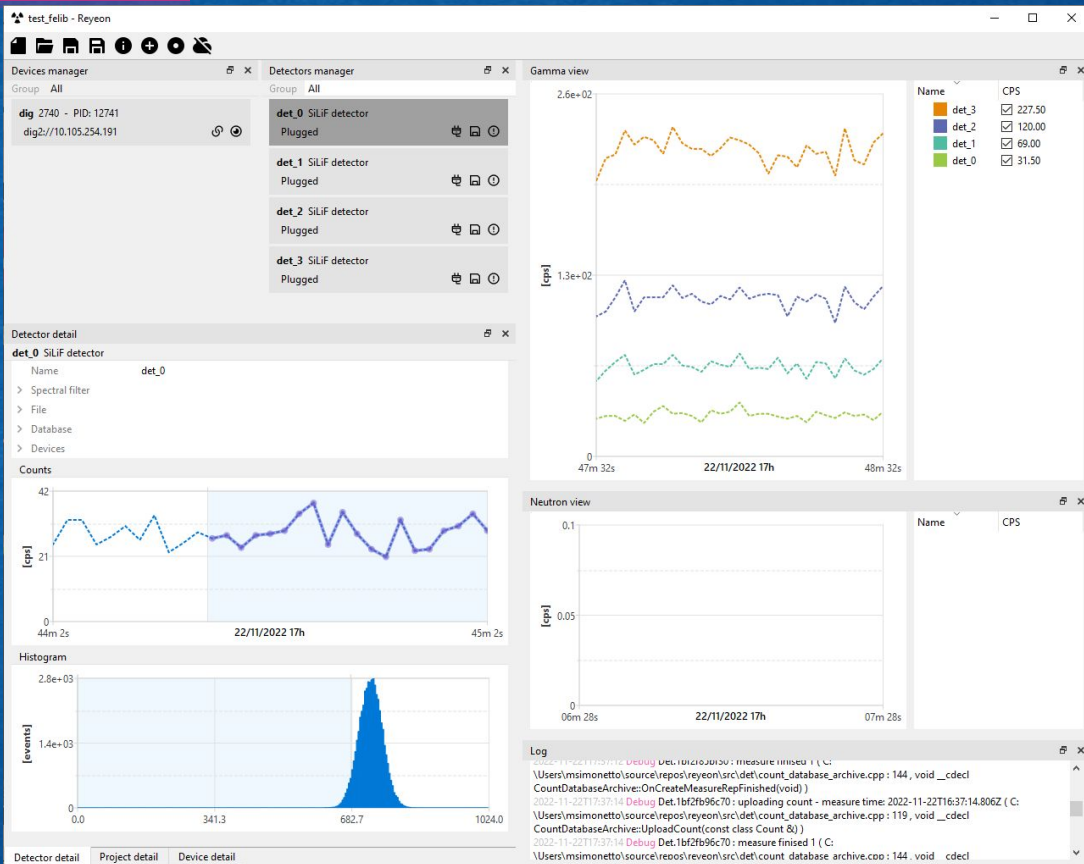
- New DAQ software.
- Underlying libraries. (including Qt)

Finish implementations

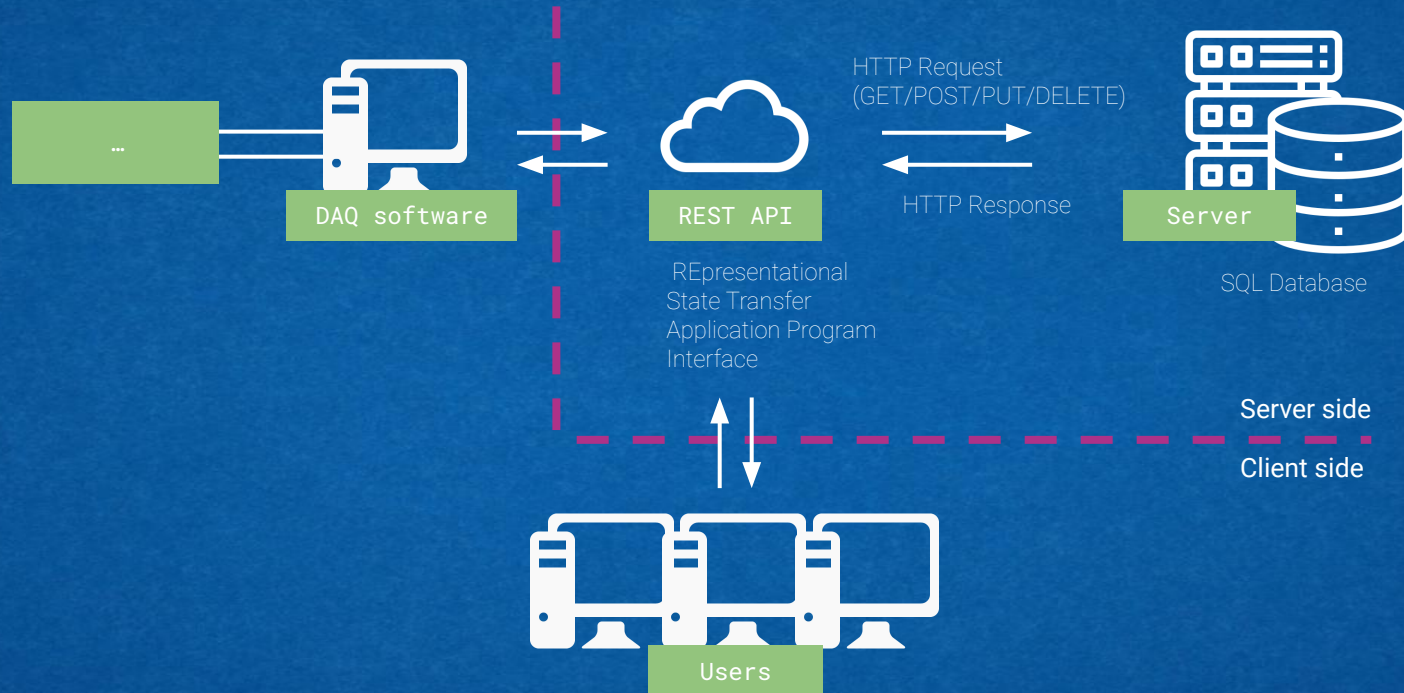
- Cloud database communication

- Alarms

Improve user experience



Database communication



Database communication

- Client side (DAQ Software):
 - HTTP client capabilities: implemented using Qt Network module.
 - Added support for authentication.
- Server side:
 - Starting point: RadBASE.
 - Java (Spring framework).
 - REST API endpoints (including authentication).
 - Web application.
 - Added features necessary for the platform.

RadBASE Accounts Items Locations Devices Utils Profile Logout

List of Items

Name: Any Category: Any Status: Any Creator: Any

RFID tag: Enter RFID tag text Filter All

ID	NAME	CATEGORY	STATUS	CREATOR	CONTAINED ITEMS	RFID TAG
1	aaa	drums	created	admin	1	X
2	bbb	bags	created	admin	0	X
3	ccc	boxes	created	technician	0	X
4	ddd	B-25	created	technician	0	X
5	eee	drums	measured	admin	0	X
6	fff	bags	measured	admin	0	X
7	ggg	boxes	measured	technician	0	X
8	hhh	B-25	measured	technician	0	X

Spectrum #1

Item: E2806D12000000021F4727EC

Radio nuclides: Cesium-137

Creator: admin
Device: Model: F300 SN: P3001