Software upgrade for FRESCA and other Ic test benches in B163 and B288

Piotr Koziol, Amalia Ballarino, Christian Barth, Hubert Reymond, Odd Oyvind Andreassen,
Why?

- Modularize and reuse the software between FRESCA, FRESCA2, IC test benches, Liquid Nitrogen test bench etc.
- Many Operating systems, One Software
- Compartmentalized, class-based code
- Standalone instruments
  - Separate interfaces
  - Isolated and coordinated operation
  - Reconfigurable
- Multiple interfaces
  - Expert tools, terminal, operational tools etc.
Common problem

We’re here

green line – not modular approach, orange line – fully modular
Why LabVIEW?

• Existing NI Equipment
• LabVIEW is a graphical high-level language
• Supports most of the instruments on the market
• Possibility to combine the code with Python and C/C++
• Possibility to write the code for FPGAs in LabVIEW
Development – Evolutionary Design

- Growing a system in a natural way, by adding the minimum amount of the code to satisfy the needs in an iterative and incremental approach.
- Work split in small work packages updated on JIRA.

https://www.industriallogic.com/blog/evolutionary-design/
Startpoint – Endpoint

GUI PC #2 (Windows 10):
- plots
- analysis

GUI PC #1 (Linux):
- plots
- analysis
- measurement control
- instrument remotes

GUI PC #2 (Linux):
- plots
- analysis

GUI PC #1 (Windows 10):
- plots
- analysis
- measurement control
- instrument remotes

PXI #1 (RT OS):
- 1x K2000 (via RS232)
- 2x K182 (via GPIB)
- 1x K7001 (via GPIB)
- 1x FPGA
- 1x Multifunctional DAQ
- 1x Sample (AIO)

FGC (16 kA) – VIA RDA3

PXIe #1 (RT OS):
- 2x DMM
- 8x 2182A (via RS232)
- 2x FPGA
- 2x matrix 32x16
- 2x multifunc. DAQ
- 2x counter card

CompactRIO (FPGA):
- 1x current input
- 1x current output

PXI #3 (RT OS / Linux):
- 12x uQDS (via USB)
- 2x multifunc. DAQ

PXI #2 (Linux):
- 2x matrix 32x16
- 2x transient recorder
- 1x multifunc. DAQ

Cryo values

FGC (32 kA) – via UDP

Gateway

Cryo values

FGC (16 kA) – VIA RDA3
Factory Method – design pattern

- Design common and reusable methods that reduces the need to redevelop interfaces for new drivers and equipment
- Object Oriented and scalable base software (OOP)

https://refactoring.guru/design-patterns/factory-method
Factory Method – DMM – example

KeithleyDMM
- ResourceName: Ref
- +Initialize
- +Version
- +Configure
- +Fetch
- +Close
- +Configure Trigger

Keithley2182A
- ResourceName: Ref
- +Initialize

Keithley2000
- ResourceName: Ref
- +Initialize

KeysightDMM
- ResourceName: Ref
- +Initialize
- +Version
- +Configure
- +Fetch
- +Close
- +Configure Trigger

Keysight34420A
- ResourceName: Ref
- +Initialize

niDMM
- InstrumentHandle: Ref
- +Initialize
- +Version
- +Configure
- +Fetch
- +Close
- +Configure Trigger

ni4071
- InstrumentHandle: Ref
- +Initialize

ni4070
- InstrumentHandle: Ref
- +Initialize

https://refactoring.guru/design-patterns/factory-method
Modularity

• Every module will be running independently in a separate thread
• Uniform behavior of each module regardless of platform and runtime

Simplified software architecture
Error Handling

- **Rade-Logger** – used to log action and error messages
- **Crucial errors** – local storage, publish if possible
Frameworks

Decision about the architecture before the development of the code. Different methods on the market:

- Actor Framework
- QMH – queue message handler
- DQMH – Delacor queue message handler
- Etc.

Queued State Machine Design Pattern
Communication

• Communication class
  - CMW at CERN
  - OPC UA outside CERN

• CMW provides communication infrastructure for all CERN accelerators. CMW has been chosen as a main communication protocol. Possibility to get, set, subscribe.

• Communication protocol encapsulated into class (to be easily maintainable and changeable).

https://wikis.cern.ch/display/MW/CMW++Controls+Middleware
NXCALS

• Centralized database for data from CERN accelerators
  • CALS (CERN accelerator logging service):
    • Java API (supported by BE-CO)
    • GUI application called TIMBER
  • NXCALS (Next CALS) replaces CALS in future:
    • LabVIEW, Java, Python API
    • Backwards compatible

https://wikis.cern.ch/display/NXCALS
Analysis tools in python?

• Python – open source programming language - widely used at CERN

• From LabVIEW 2018, Python can be called natively
  • Thin GUI client calling python algorithms

• Many operating systems supports Python

• Huge community

• Simple syntax – readable and maintainable

• Supported by BE/CO
Documentation

• Everything documented on readthedocs
Conclusions

• Main goal
  • Fully modular, extendible and class-based code
• Possibility to add new behaviour to existing interfaces, minimizing the development effort
• Re-use existing functionality
• Fully documented code for an easy transition
• Conforming to CERN standards and best practices