Karabo

Integrating Control, Data Acquisition, Data Management, and Data Analysis Tasks into a Single Software Framework

Burkhard Heisen
Control and Analysis Software Group – European XFEL GmbH
European XFEL User’s Meeting
Hamburg, January 28, 2015
Why?
- One software framework, one user interface, one programming interface
- Less learning, less teaching, less maintenance, less changes, less conversions, more freedom

Why else?
- Our requirements are different to those of classical synchrotron or high energy physics experiments
- We invest a lot of money for better science, a small fraction is for better software
A typical use case:

Control
- drive hardware and complex experiments
- monitor variables & trigger alarms
  - allow some control & show hardware status
  - show online data whilst running
  - setup computation & show scientific results

DAQ
- data readout
- online processing
- quality monitoring (vetoing)

DM
- storage of experiment & control data
- data access, authentication, authorization etc.

DA
- processing pipelines
- distributed and GPU computing
- specific algorithms (e.g. reconstruction)

Tight integration of applications

Accelerator, Undulator, Beam Transport, Sample Injector
Karabo – Integrating control, data acquisition, data management and data analysis

The vision – From a data perspective

Not shown is technical infrastructure such as switches. Alignment datasets are shipped with the data products and tools for coordinate system conversion are provided by the facility.

Burkhard Heisen (CAS Group)
Karabo is a “device” based system

- DAQ Equipment
  - e.g. commercial camera
  - e.g. 2D-detectors (AGIPD, LPD, DSSC)
- Equipment Control
  - e.g. motor, pump, valve, sensor
- User interface(s)
- Service Device
  - e.g. calibrationManager, configurationManager
- Message Broker
- Data Storage Node
  - e.g. storage of data from runs
- Analysis Node
  - e.g. calibrationManager, projectManager, GUI server
- Composite Device
  - e.g. complex detector motion control

Burkhard Heisen (CAS Group)
Device classes can be loaded at runtime (plugin technology)
- Can be written in C++ or Python
- Devices completely describe themselves. Allows automatic GUI creation
- Runtime-extension of properties, commands and attributes is possible
- Hierarchical groupings of parameters are possible
- Any property setting or command execution can be restricted to a set of allowed states
- The GUI reflects state limitations by enabling/disabling buttons and properties dynamically
Devices can act as modules of a streaming data pipeline

- Configurable generic input/output channels on devices
- Developers just need to code the `onData` and (optionally) the `onEndOfStream` method
- IO system is decoupled from processing system (process whilst transferring data)
- Input channels configure whether they share the sent data amongst all input channels or whether they receive a copy of each data token
- Broker-based communication transparently establishes a point-to-point connection
- Any pipeline device has full access to the live control-system (feedbacks easily possible)
Once resources are available, input channels request new data from connected output channels.

Configurable output channel behavior in case no input currently available: error, queue, wait, drop.
Karabo – Integrating control, data acquisition, data management and data analysis

Graphical interface - Overview

- User centric and access-controlled setup (login at startup)
- Dock-able and resizable multi-panel, all-in-one user interface
- Live navigation showing all device-servers, plugins, and device instances
- Automatically generated configuration panel, allowing to
  - PowerPoint like, drag & droppable, tabbed custom scene
  - Project panel for persisting configurations, macros, scenes, resources, etc.
  - Centralized logging information, notification handling, documentation, etc.
Pipeline configuration
- Whole devices can be **dragged** (from left side) as pipeline nodes
- Dragging individual parameters from right is still possible (e.g. control parameters)
- Devices can be **grouped** and edited as group (connections and configurations)

Macro editing
- Editing and execution from within GUI or using regular terminal
- Macro parameters and functions **integrate** automatically into **configuration** panel
- Macro API can be **interactively executed** in embedded IPython interpreter
- Macro writing is something we hope our **users can** easily **do** themselves
Karabo – Integrating control, data acquisition, data management and data analysis

Cameras, power supplies, heaters...

- **Supported standards**
  - **GenICam** (generic interface for cameras) - industry standard
  - **Lima** (library for image acquisition) – ESRF project
  - **SPCI** (Standard Commands for Programmable Instruments)

- **Cameras**
  - Basler (all GigE), Photonic Science (sCMOS)
  - IMPERX (Bobcat B4020), Andor (Neo)
  - AVT (all Prosilica), Axis (all)

- **Other devices**
  - FUG, Heinziger (power supplies), TEM (beam lock)
  - Amphos (laser power stage)
  - EnergyMax (laser sensors)
  - Lakeshore (heater), STS (spectrometer)
Karabo – Integrating control, data acquisition, data management and data analysis

Motors, pumps, valves... and real-time control

- Karabo itself does not directly provide real-time processes/communications -> uses Beckhoff layer
- Beckhoff PLC-firmware blocks are mapped to Karabo devices and can be **auto-generated** from PLC self-description
- Integrated devices (now extending to different vendors):
  - Turbo-, ion-, and scroll-pumps ✔
  - Valves, gauges, temperature sensors ✔
  - Stepper-, piezo- and servo-motors ✔

![Diagram of Karabo and Beckhoff integration](image)

*Courtesy of P. Geßler*
Digitizers, ADCs, Timing interfaces...

- Digitizers (used in XGMs, PESs)
- Fast ADCs (analog to digital converters)
- Timing interfaces (in progress)
- Simulink/Matlab based FPGA programming framework with **Karabo integration**

![Diagram of FPGA and XML file](image)

**Final Design**

**XML FILE** User Registers and Memories Description

**Source Files** with script

**Bus protocol logic generated automatically**

**Software (Karabo)**

**Courtesy of P. Geßler**

Burkhard Heisen (CAS Group)
Pump probe laser setup (laser group)
-Runs burst mode according to XFEL (4.5 MHz intra burst)
-Control of motors, cameras, MicroTCA based ADCs, image processing devices
-Controls pulse arrival time in a closed loop (motor position correction depending on ADC signal)

Vacuum setups
- Control of e.g. differential pumping sections
- Typically includes several valves, pumps, and PLC-interlocks for safety
- Pressure monitoring and history trend line plotting
Large Pixel Detector (LPD) prototype testing
- Detector control (Python based)
- Data acquisition and storage (HDF5) (C++)
- Online calibration processing, analysis and correction (Python and CUDA based)
- Interaction with calibration database (Oracle)
- Full integrated run control expected in summer

LPD quadrant prototype

Server setup necessary to harvest the data

Visualization of a LPD quadrant in Karabo

Courtesy of S. Hauf

Burkhard Heisen (CAS Group)
Serial femtosecond crystallography

- Aim is to build an **SFX data processing pipeline** to match the high repetition rate offered at the European XFEL
  - 5000 images/sec
  - @ 2.3 sec per image
  - 30% hit rate
  - 3500 cores needed

Linear speed up with number of cores!
Karabo is unique in **seamlessly combining** instrument control, experiment configuration, streamed data analysis and visualization.

Its is **designed for being extended** easily and will be ready to execute your algorithms during the experiments.

Karabo proves capable to run complex installations even today.

Currently the focus is on supporting our in-house users and external collaborators -> access is on request.

Public release of the core system under the GNU GPL license will follow in 2015.

Stay tuned…
Thank you for your kind attention.

Contact details: burkhard.heisen@xfel.eu