DAQ based high level software applications using MATLAB

V. Kocharyan, V. Rybnikov,

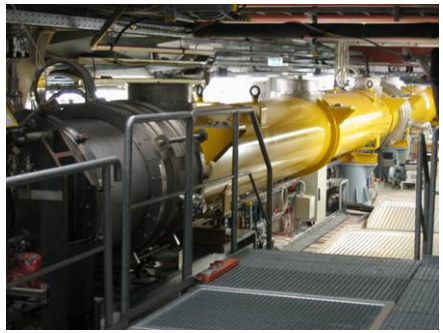
K. Rehlich, R. Kammering

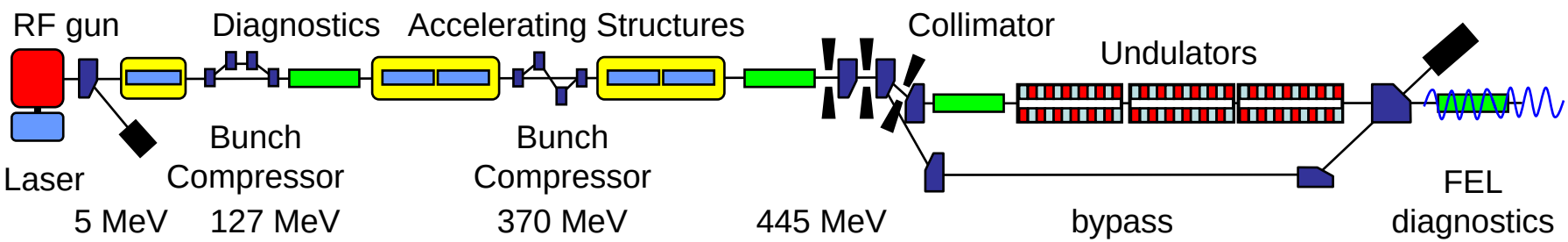
Outline

- The FLASH facility @ DESY
- The DOOCS control system
- The FLASH DAQ system
- The feedback and monitor API (FBM API)
  - Motivation
  - Architecture
- An example: the energy server
- Summary and conclusions

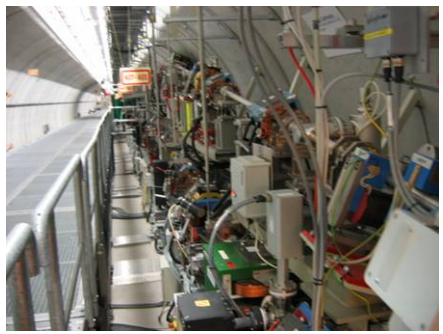
The FLASH facility @ DESY

FLASH
Free-Electron Laser in Hamburg








250 m

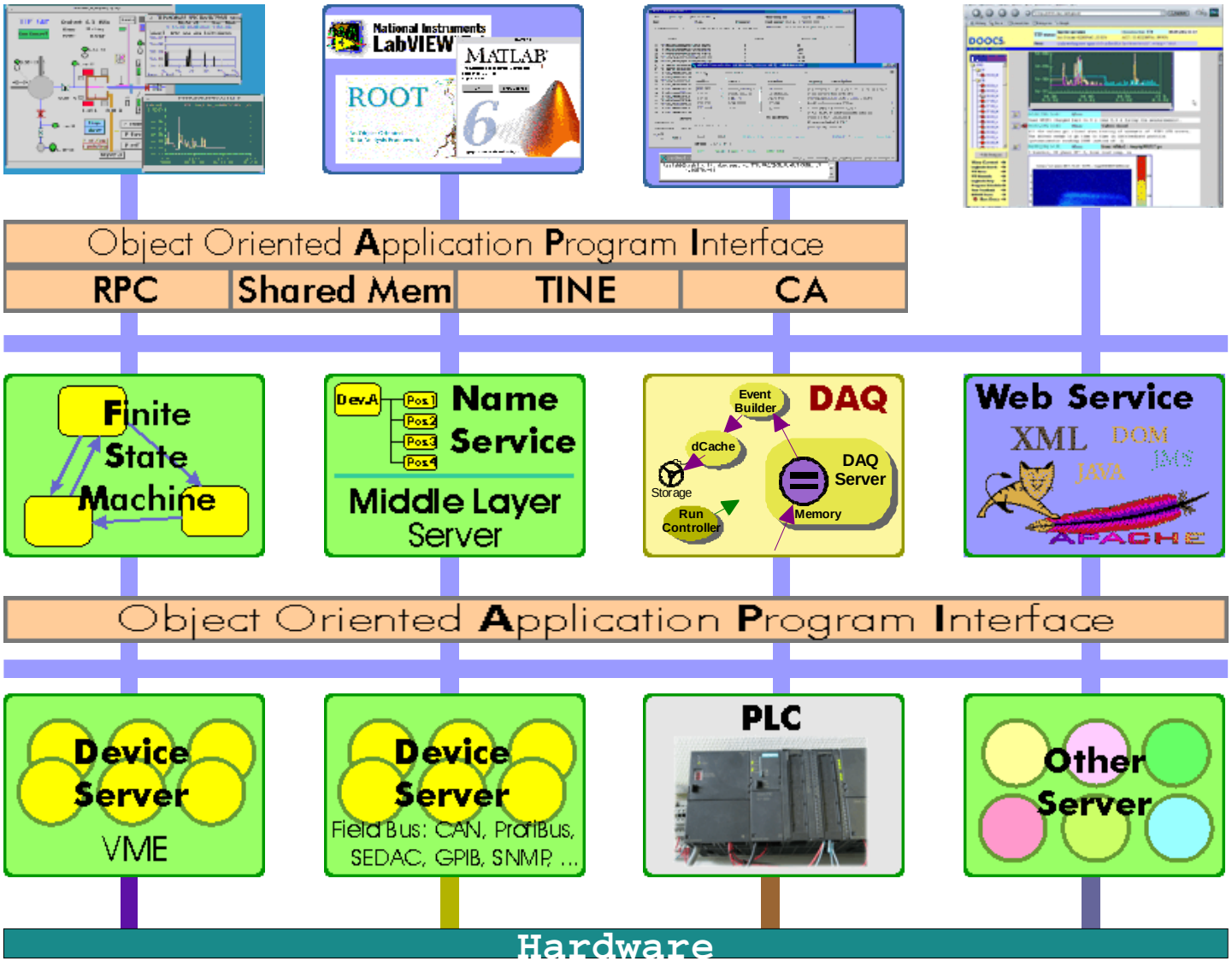
The DOOCS control system

The Distributed Object Oriented Control System:

- Object oriented
  - Modern technology
  - Able to handle huge number of devices
  - Device objects on server and display
- Well defined separation of tasks in *three* layers
  - Device servers **|** middle layer servers **|** client applications
- Modular
  - Based on libraries (C++)
- Self-contained device servers
  - Auto restart with previous settings
- All parameters on-line configurable

Implements the required technology for the XFEL

The DOOCS control system



User interface

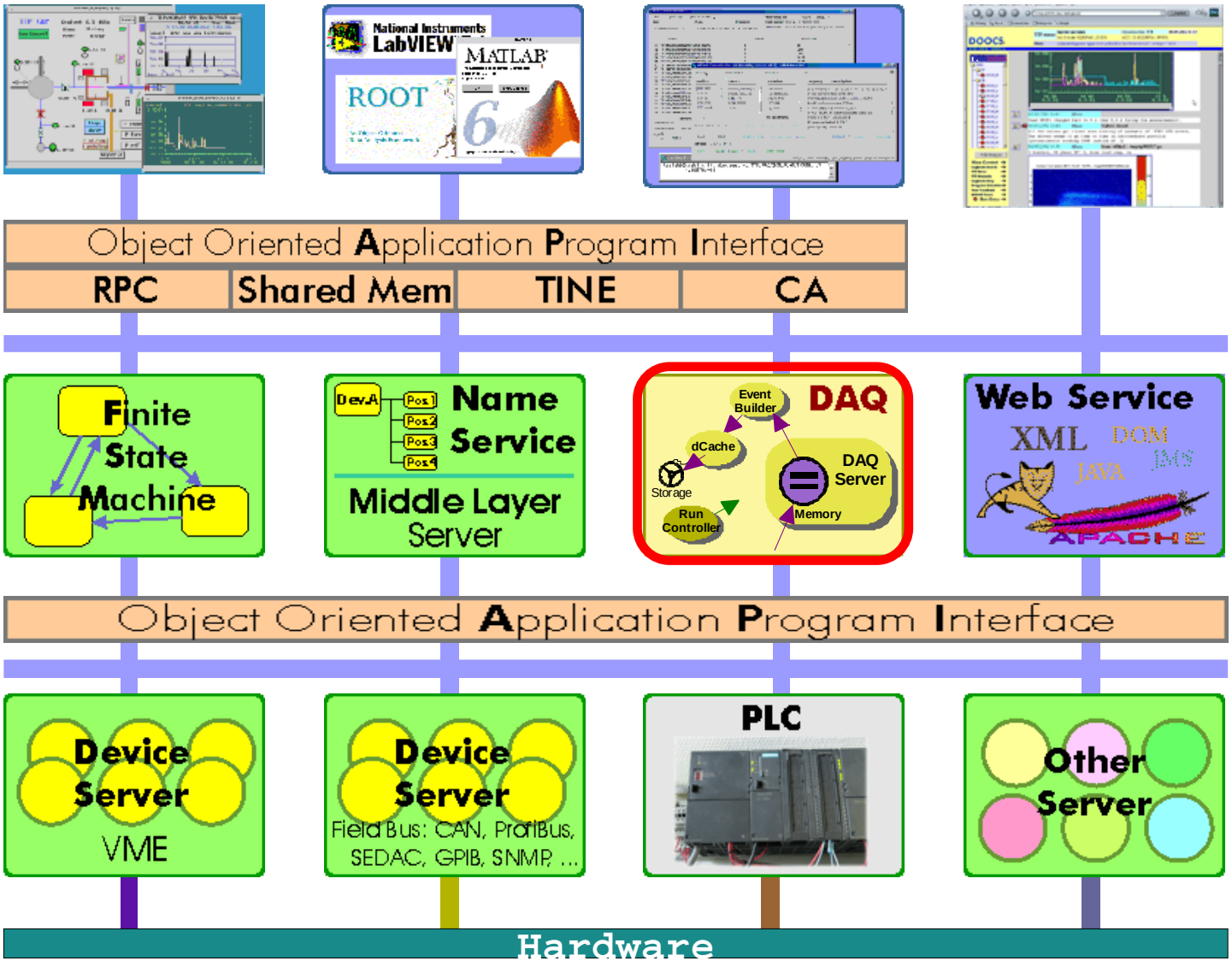
Program interface

Middle layer

Program interface

Hardware interface

The FLASH DAQ system



User interface

Program interface

Middle layer

Program interface

Hardware interface

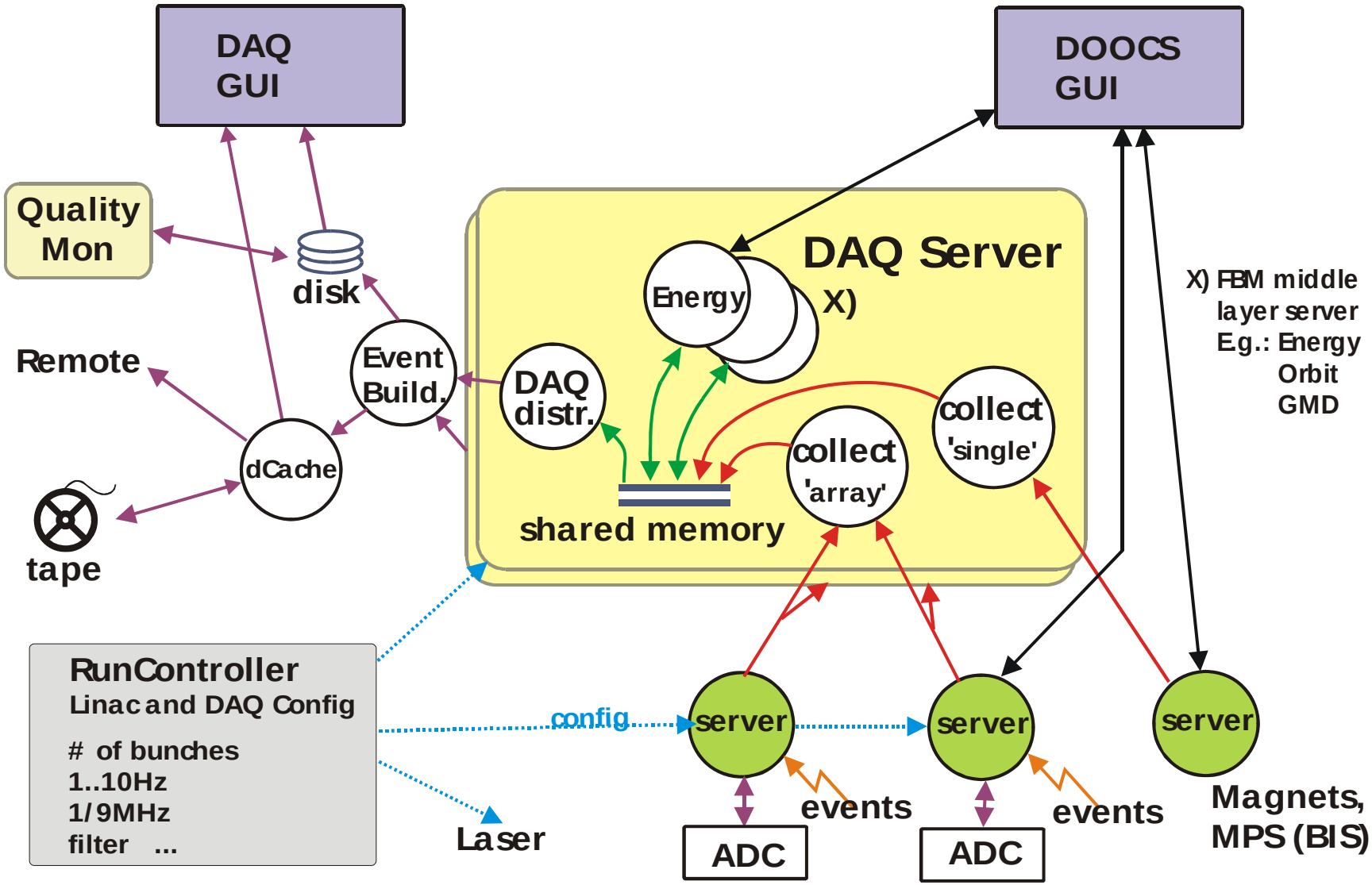
The FLASH DAQ system

Idea and motivation

- Improve, better understand, and maintain the linac
  - Error statistics: find reasons of faults, improve reliability
  - Operation optimization, find best parameters
  - Experiments can correlate measurements with the machine
- **Store** the data of the **experiments** and all beam relevant data of the **linac**
- Provide the **tools to analyze** the stored data for local *and remote users*
- *Novel integration* of the **HEP** DAQ and **accelerator controls**
- **Collaboration**: Cornell, Michigan, Zeuthen + Hamburg

Central pool for **middle layer services** (e.g. feedbacks)

The FLASH DAQ system



The FLASH DAQ system

The main DAQ server

- 8 double core SPARC CPUs
- 32 GB common memory
- 4 x 1Gbit Ethernet
- 1.7 TB local storage
- fully redundant fan, PS, …

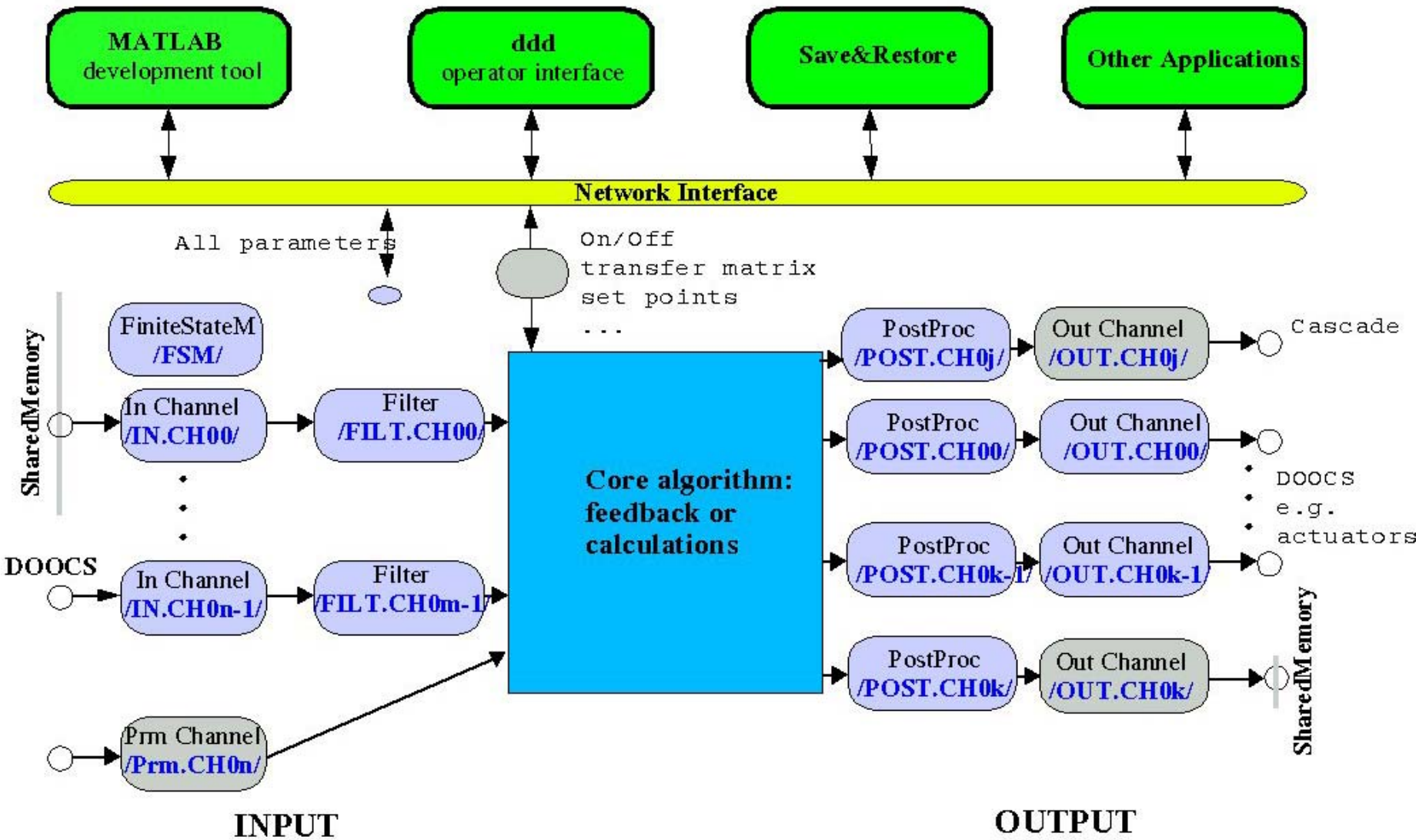
The FBM API - Motivation

The Feedback and monitor API

- Control and operate FBs from DOOCS
  - Avoid *wildly* running FBs
  - Have one common interface for FBs
- Benefit from standard DOOCS features (histories, …)
- Reduce load on front ends (using central DAQ SHM)
- Have common exception handling
  - e.g. bunch pattern generation
- Generic skeleton for high level software applications
  - Can be used for various purposes (auto calibration, …)

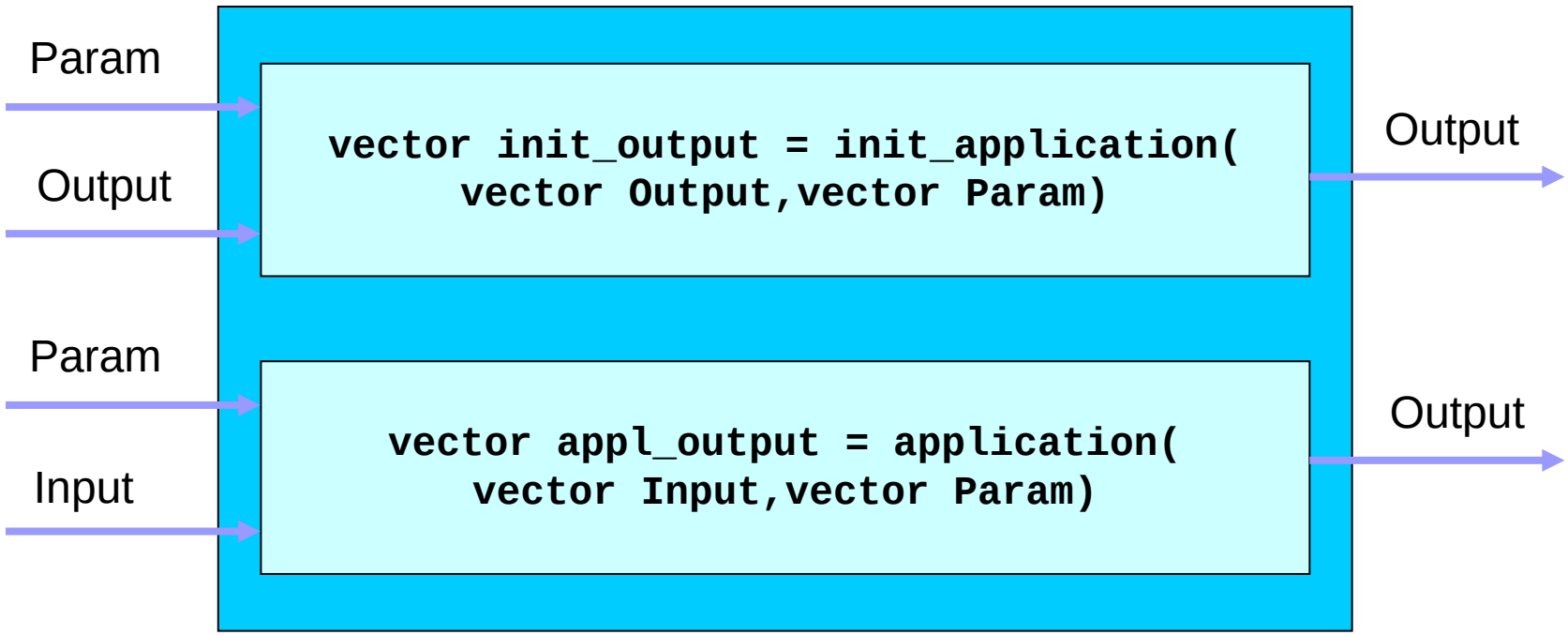
Attach all high level software applications at the FBM

The FBM API - Architecture



The FBM API - Architecture

Core algorithm: C++ or MATLAB



Same interface in both cases

An example: the energy server

Task: Measure energy in dispersive section

1. **Read magnet currents**

   → **calculate nominal trajectory through the section**

2. **Read beam position after the dispersive section**

   → **calculate displacement (Δ x) from nominal trajectory**

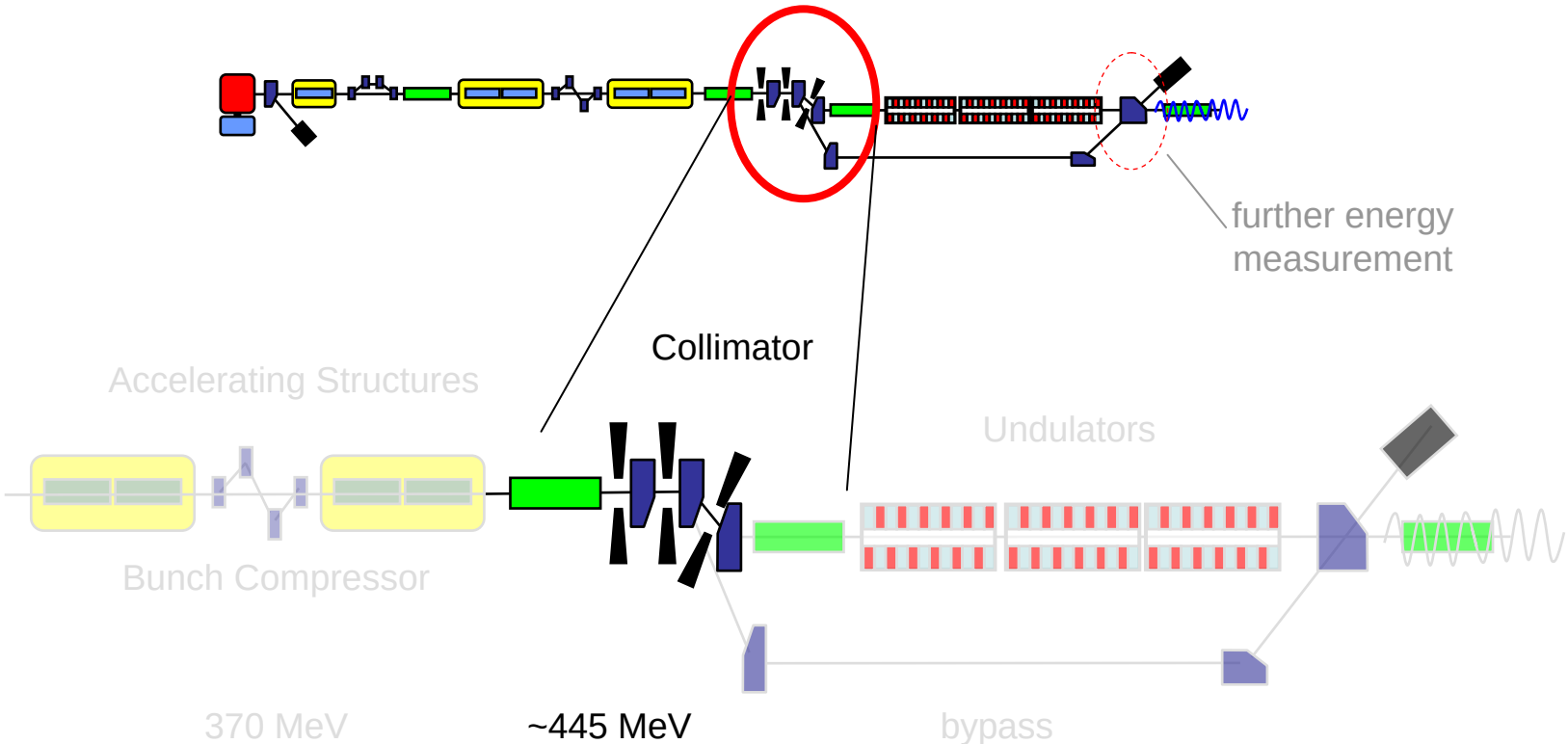
3. **Calculate ΔE from Δ x**

   Determine E_{total} from $E_{nominal}$ + ΔE

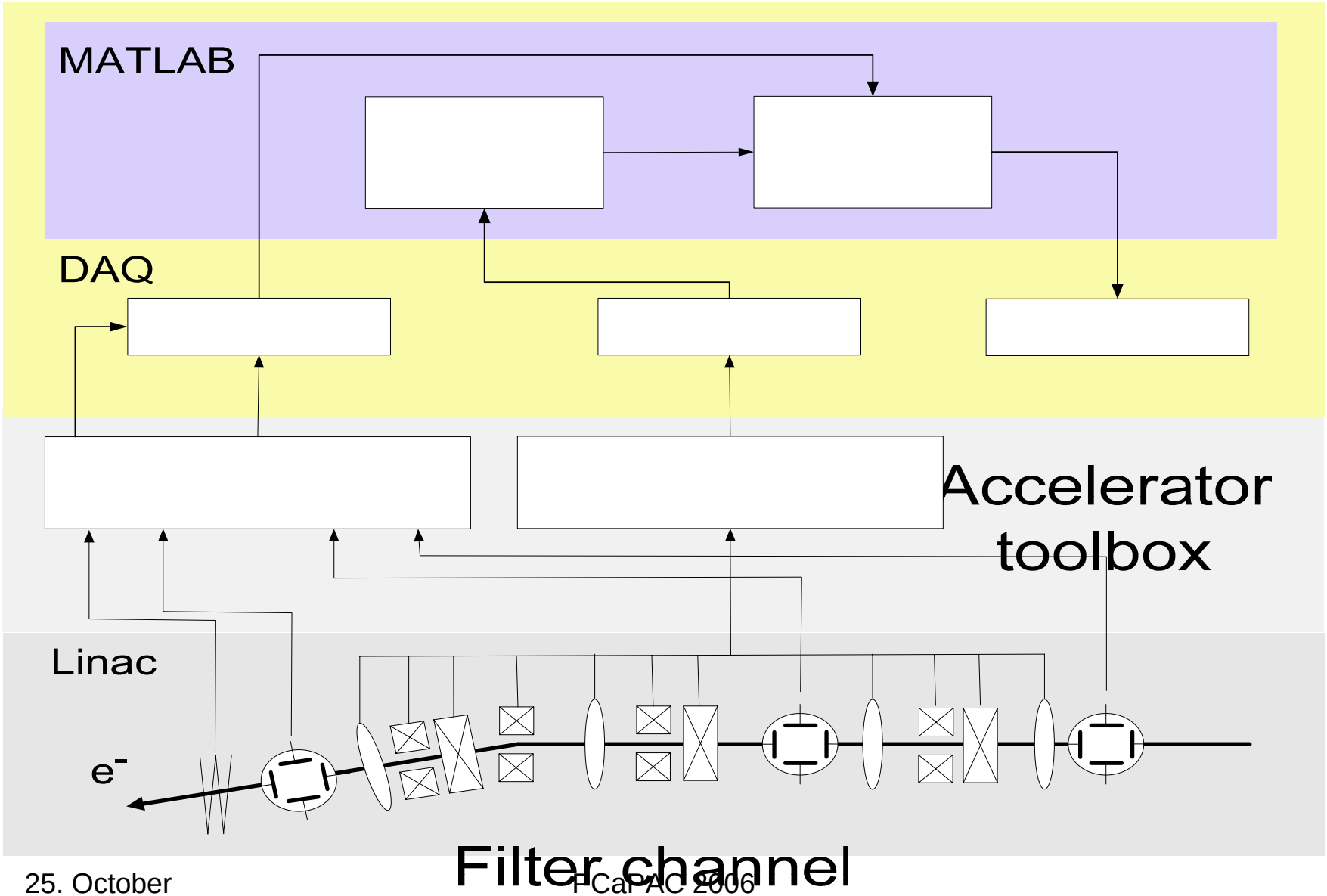
all with single bunch resolution

An example: the energy server

Location of energy measurement

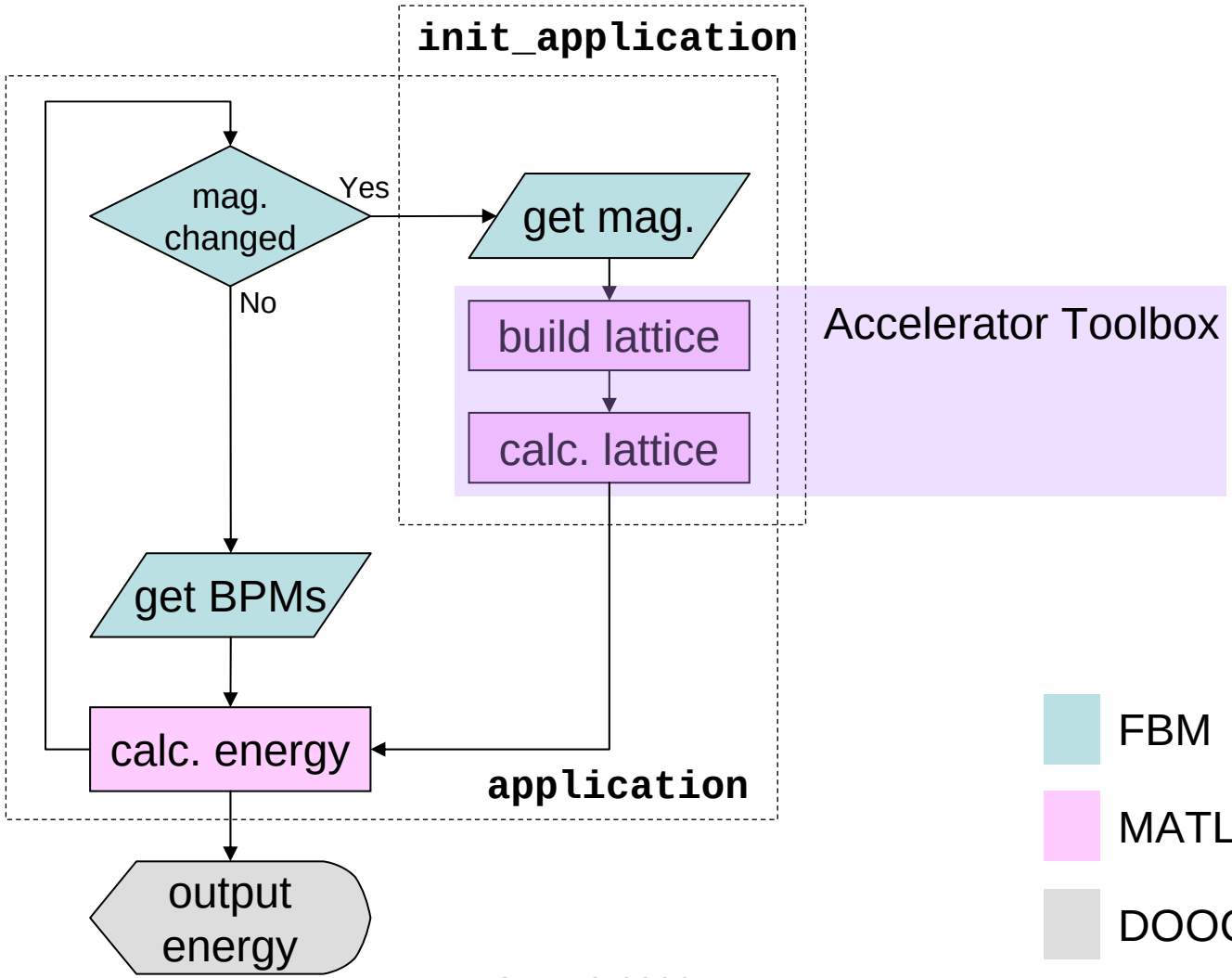


An example: the energy server



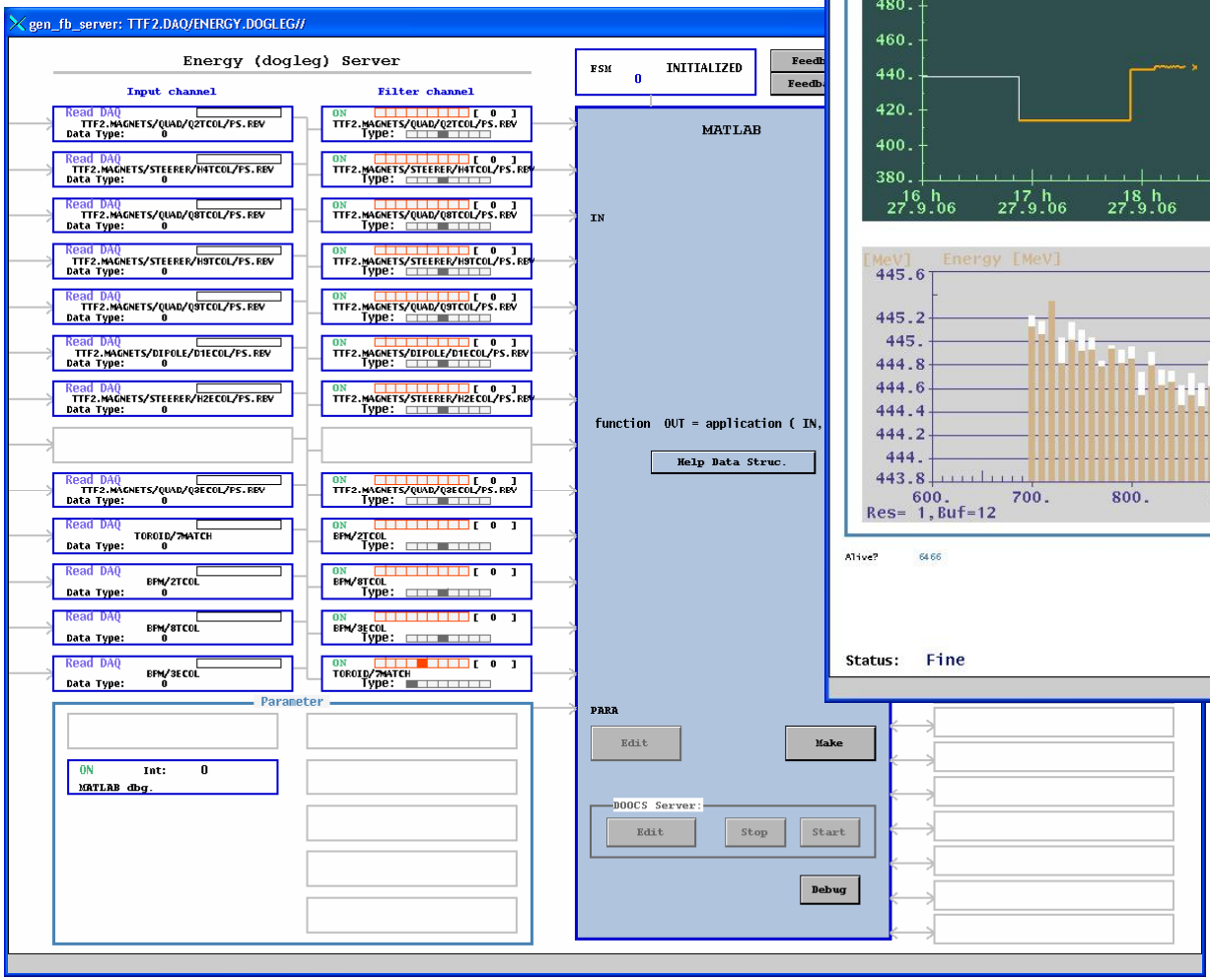
An example: the energy server

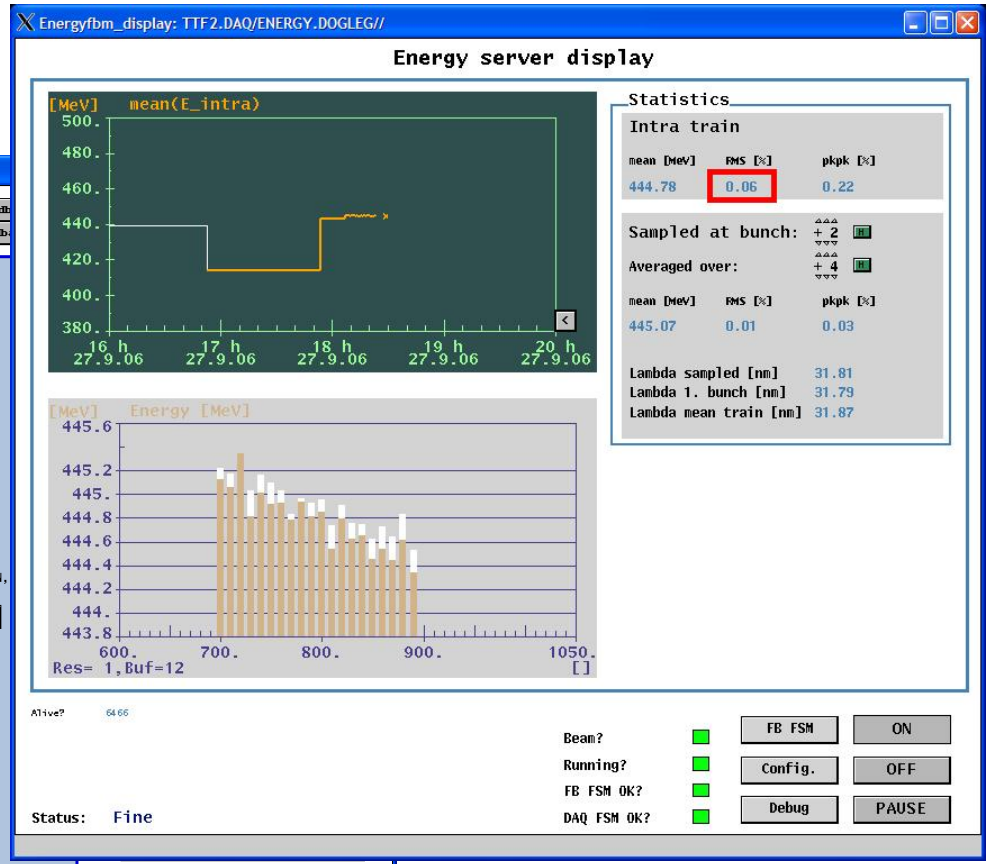
Logical flow



An example: the energy server

Configuration panel





Energy server display

Summary and conclusions

Middle layer server using the FBM API:

- *Acqiris* data compression (E)
- Calibration server for FEL experiments (E)
- Charge FB (O)
- Energy monitor (O)
- Gas Monitor Detector (OE)
- LLRF monitor (D)
- Orbit FB (D)
- Photon energy monitor (E)

E = used for experiments
O = used for standard operation
D = Special diagnostics / partly used for operation

Summary and conclusions

Some statistics:

- Several billion calls to MATLAB application

  → API is stable

- missed events < **1%** (macro pulses)

  → Data quality if sufficient

- CPU load ~ **60%** (off one SPARC CPU)

  → 16 CPUs we can run dozens of servers

- Roundtrip (appl.) ~ **100 ms** (MATLAB)

  → 5 Hz – no problem

Summary and conclusions

+ FBM API offers highly configurable interface
+ Good reliability of the interface
+ Approach applicable for many purposes
+ Very complex calculations possible

– Configuration is tricky (→Provide Wizard)
– Currently 10 Hz with complex MATLAB application can not be ensured

The End

Thank you for your attention!

References:

FLASH: http://flash.desy.de

DOOCS: http://doocs.desy.de

DAQ: http://gan.desy.de

FBM: https://ttfinfo.desy.de/TTFelog (restricted)
→ doc → Subsystems → Feedbacks