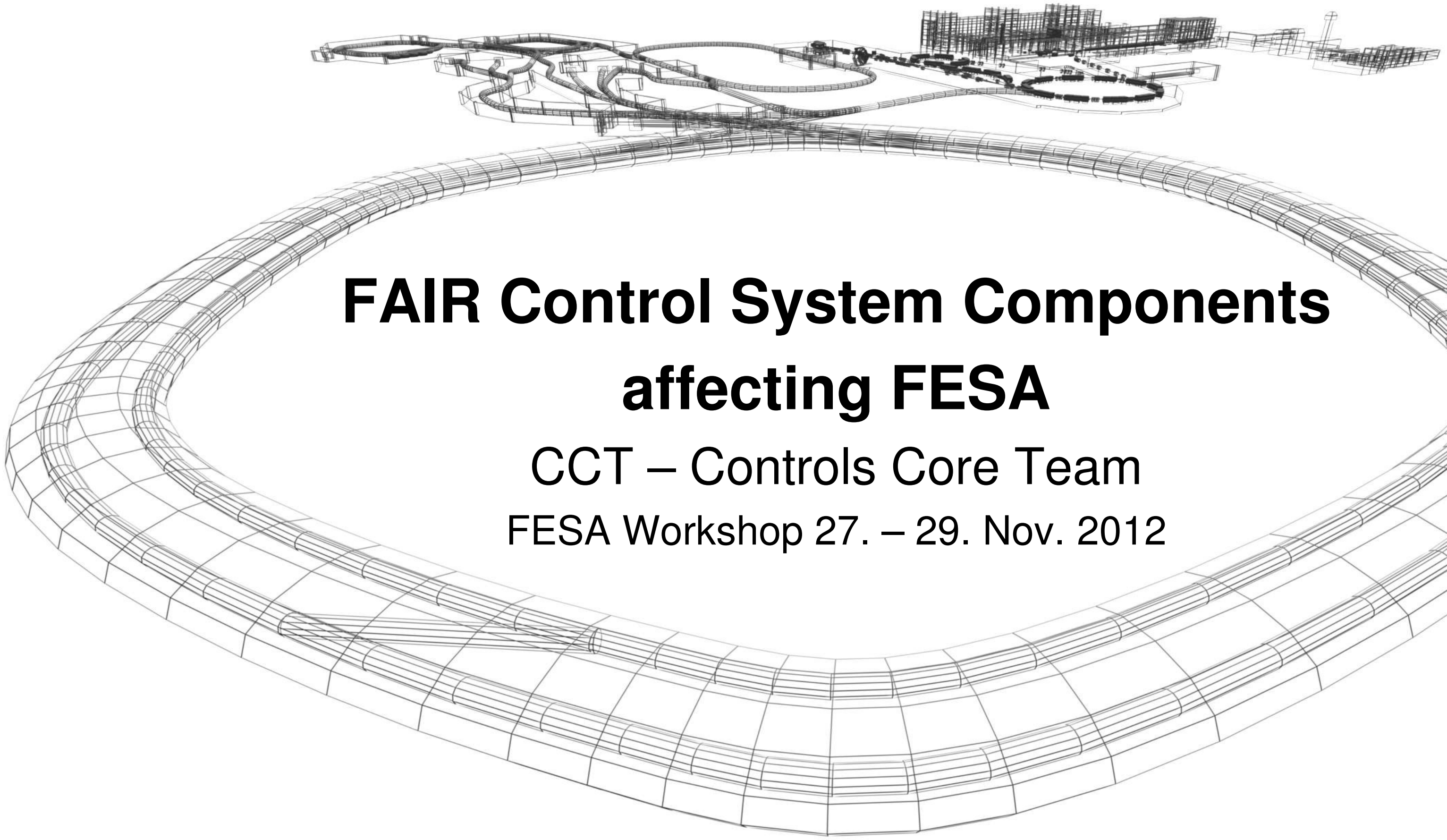


FAIR Control System Components affecting FESA

CCT – Controls Core Team

FESA Workshop 27. – 29. Nov. 2012

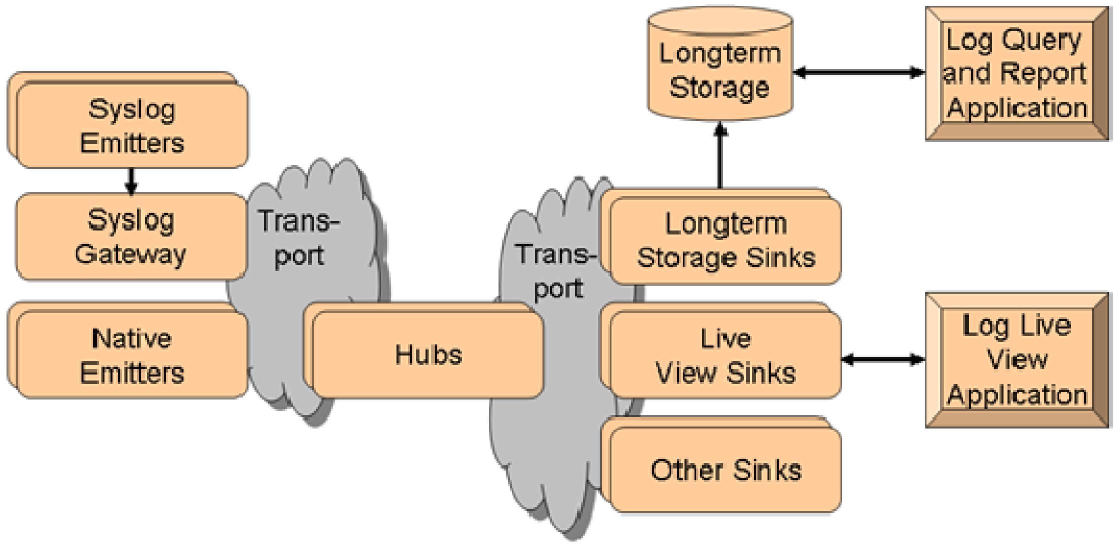
FAIR control system components

- FAIR control system components which affect FESA core or used by FESA class code
  - Diagnostic Logging System (FESA core, FESA class code)
  - Alarm System (FESA class code)
  - Timing System (FESA core)
  - Post Mortem System (FESA core, FESA class code)
  - Access Rights System (FESA core)

- Common components at CERN and FAIR
  - Device Access Middleware

- How to build lab-specific FESA?

Diagnostic Logging System



Schematic layout of the Diagnostic Logging System components

Diagnostic Logging System

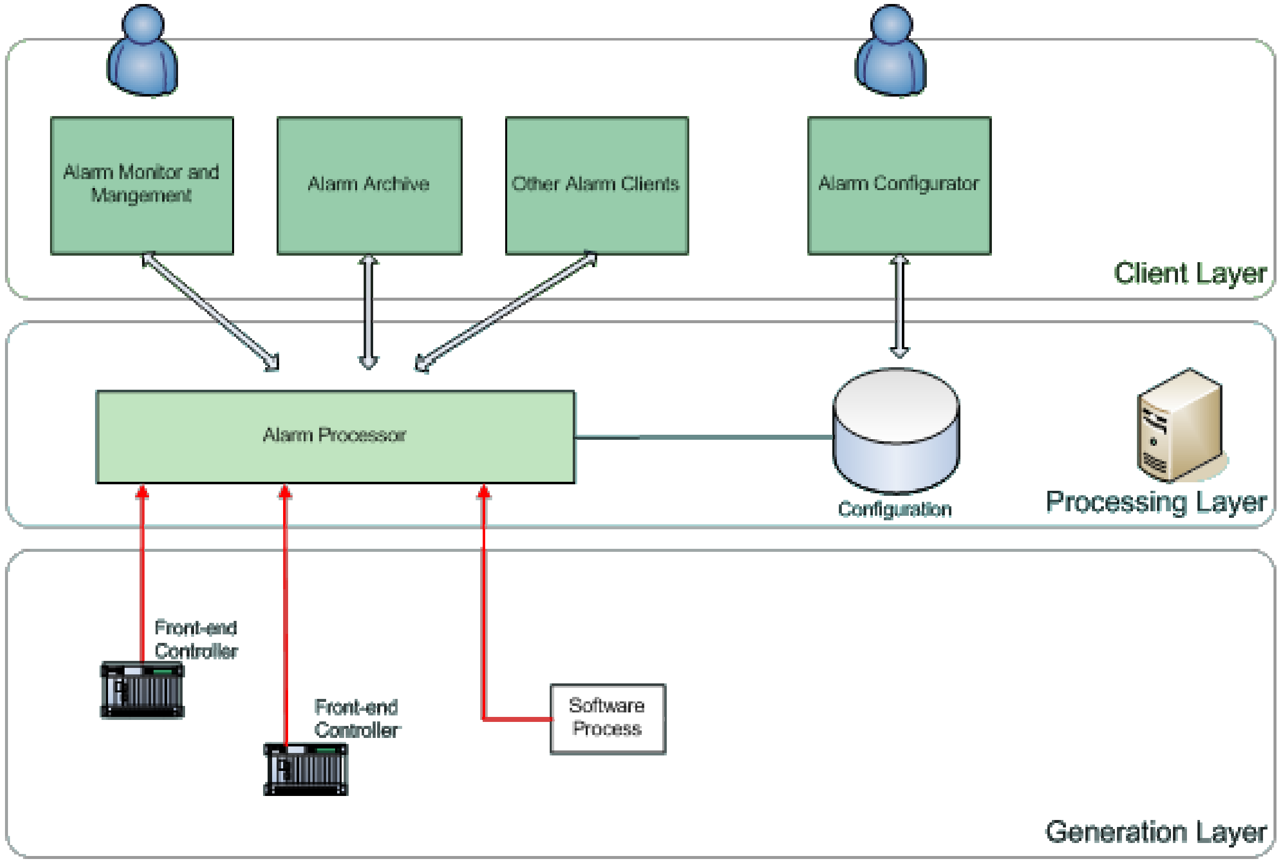
- Produces human readable text
- For diagnostic and debugging usage
- Used by developers and technical maintainers

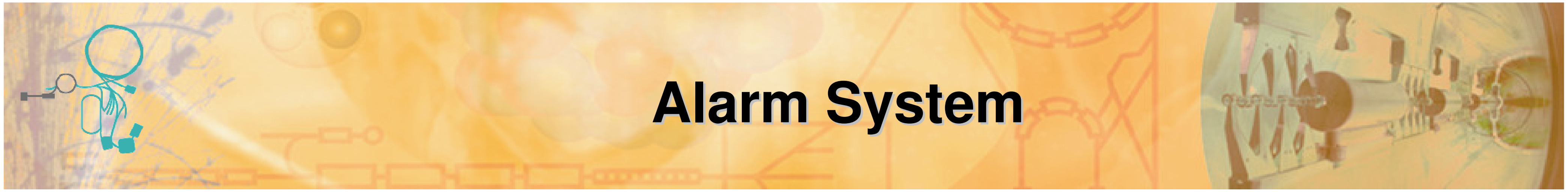
- Used by any control system component
- APIs for C, C++, Java, Python, FORTRAN
- Based on STOMP and/or ActiveMQ
- Provides a syslog gateway
- If possible, to be used (as base) for operational logging, too
  (e.g. “user U has set value V of device D at time T using tool O”)

- FESA class code uses C++ API
- FESA core ???

Alarm System

... enables hardware and software components to indicate malfunctioning





Alarm System

- is based on a message system (JMS, ActiveMQ, AMQP?, ...)
- has means to filter alarms and to reduce alarm cascades
- provides alarm generation API for C++ used by FESA class

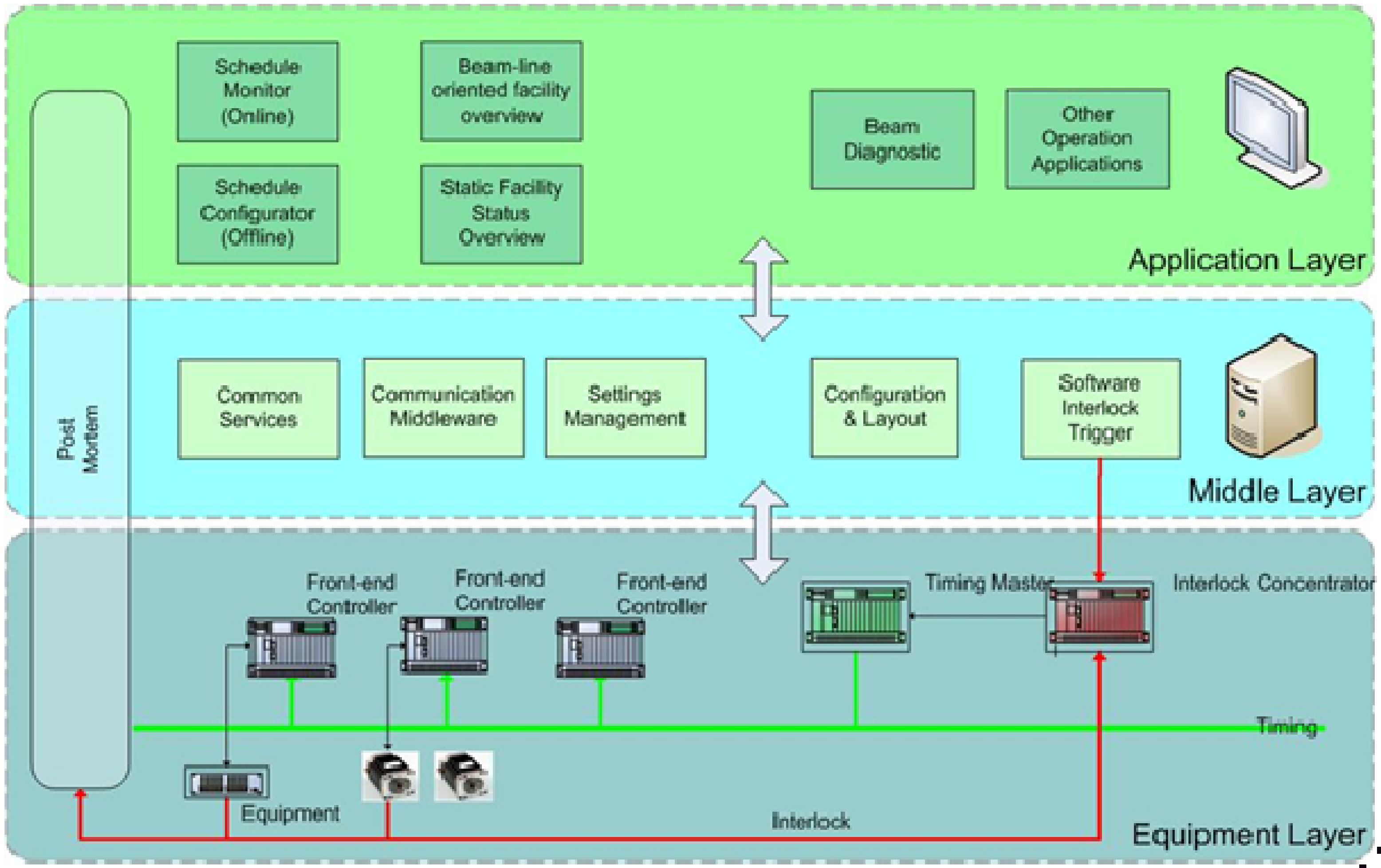
An alarm

- is stateful, i.e. active or inactive (supplied in FESA class code at runtime)
- has autonomous information like source ID, time stamp, error description, etc. (supplied in FESA class code at runtime)
- has non-inherent information like whether the alarm must be acknowledged, under which condition it may be masked, what action is to be performed, etc. which is supplied by the alarm processing tier

FESA class code must supply functionality of “When and Why” an alarm is sent. Configuration can be done via FESA properties, e.g. upper-lower limits.

Timing System

... synchronizes device actions using timing events



Timing System

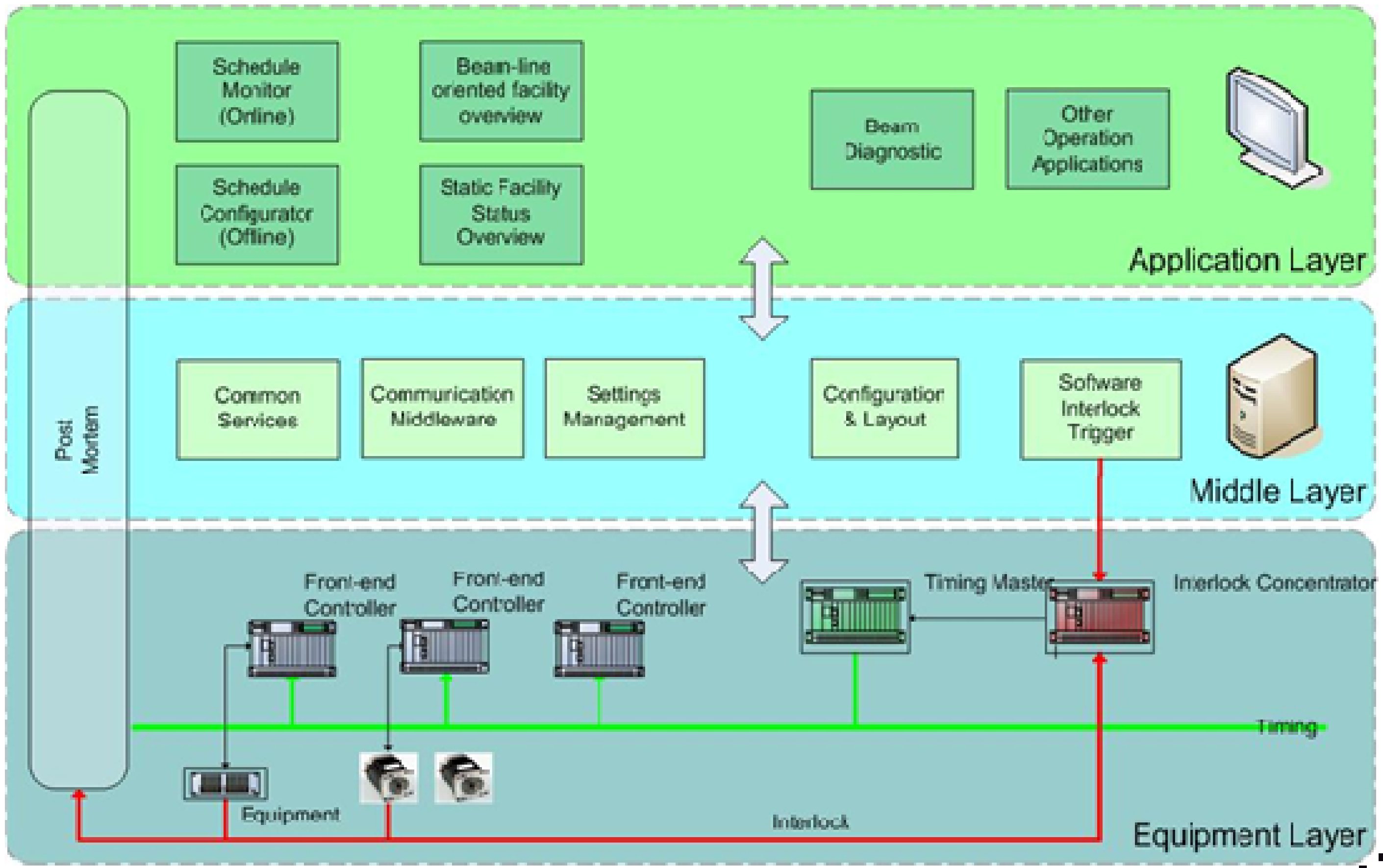
- FAIR timing based on White Rabbit
- No other (older) timing systems for FESA devices at GSI/FAIR
- One timing master for all machines

FAIR-specifics:

- Timing event payload
- Multiplexing Context: FESA classes (design?) and timing library affected; FESA Core too?
- FESA support (?) for machine-specific Commit events for transactional settings
- FESA support (?) for machine-specific Post Mortem events
- Timing Receivers for different FECs (SCU, MEN, Libera, uIOC(?), ...) in different form factors (uTCA, VME, PCIe, PMC, standalone)
- FEC- and form factor-specific drivers needed

Post Mortem

... provides a complete snapshot of the machine for further analysis in case of a major malfunction



Post-Mortem System

- A (machine-specific) post-mortem event freezes the post-mortem buffers
- Front-ends collect and store relevant data continuously in post-mortem buffers
- Buffer creation, access and freeze/unfreeze as well as post-mortem event handling must be, as far as possible, supported by the FESA framework.
- FESA class code adds data to post-mortem buffers, too

- Slowenian FAIR-Inkind workpackage will provide generic enhancements to, and/or a common toolset within the FESA framework
- FESA Core modifications necessary?

Access Rights System

- CERN
  - RBAC
- FAIR
  - To be specified

FAIR control system components

- FAIR control system components which affect FESA core or used by FESA class code
  - Diagnostic Logging System (FESA core, FESA class code)
  - Alarm System (FESA class code)
  - Timing System (FESA core)
  - Post Mortem System (FESA core, FESA class code)
  - Access Rights System (FESA core)

- Common components at CERN and FAIR
  - Device Access Middleware

- How to build lab-specific FESA?

Device Access Middleware

- CERN
  - FESA with CMW/RDA
  - based on omniORB 4.1.2 with 'CORBAthreadPriority' library extension
  - 0MQ in the future

- FAIR
  - FESA with CMW/RDA, later 0MQ
  - Existing system (GSI accelerators only):
    - based on omniORB 4.1.3
    - nevertheless runtime errors when FESA classes communicate with existing system

But: CMW/RDA at GSI

- at the moment we depend on binaries(!) copied from CERN (only C++, Java ok)
- we must be able to build the whole system at GSI

FAIR control system components

- FAIR control system components which affect FESA core or used by FESA class code
  - Diagnostic Logging System (FESA core, FESA class code)
  - Alarm System (FESA class code)
  - Timing System (FESA core)
  - Post Mortem System (FESA core, FESA class code)
  - Access Rights System (FESA core)

- Common components at CERN and FAIR
  - Device Access Middleware

- How to build lab-specific FESA?

Building lab-specific FESA

- Both labs must be able to completely build FESA including the necessary components
  - no copy of binaries from one lab to another
  - lab-specific components must be decoupled from FESA using abstraction layers
  - common but configurable build system?

- Approach
  1. Step: FESA and all necessary CERN-components can be build at GSI and at in-kind contributors (e.g. Cosylab)
  2. Step: Step by step decoupling of (at least lab-specific) components by implementing abstraction layers