Data Acquisition Backbone Core library

Jörn Adamczewski-Musch, Hans G.Essel, Nikolaus Kurz, Sergey Linev
GSI, Experiment Electronics: Data Processing group

Release V1.0
Use cases
Readout controller board

Work supported by EU RP6 project JRA1 FutureDAQ RII3-CT-2004-506078

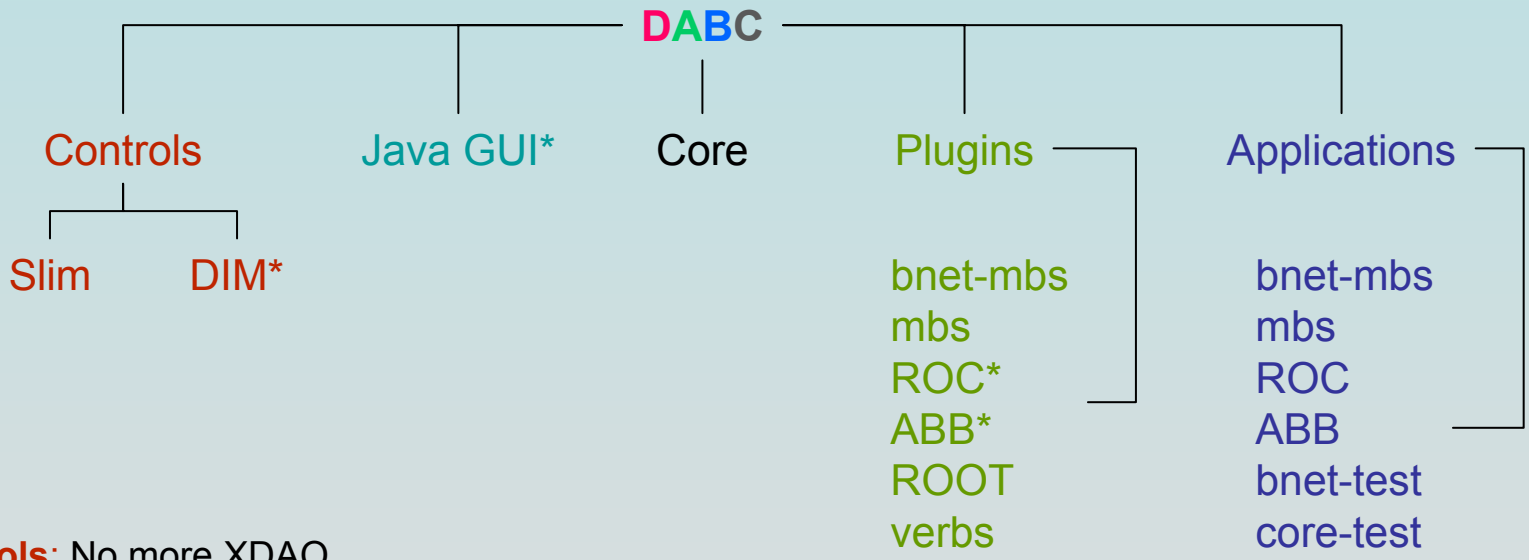
Software packages developed:

1. 2005 Simulation with SystemC (flow control, scheduling)
   - Meta data on data network

2. 2006 Real dataflow core (round robin, with/without sychronization)
   - Linux, InfiniBand, GB Ethernet
   - Simulates data sources

3. **2007/8 Data Acquisition Backbone Core DABC (includes dataflow core)**
   - **Controls, Configuration, Monitoring, GUI ...**
   - **Real data sources**
   - **General purpose DAQ framework**

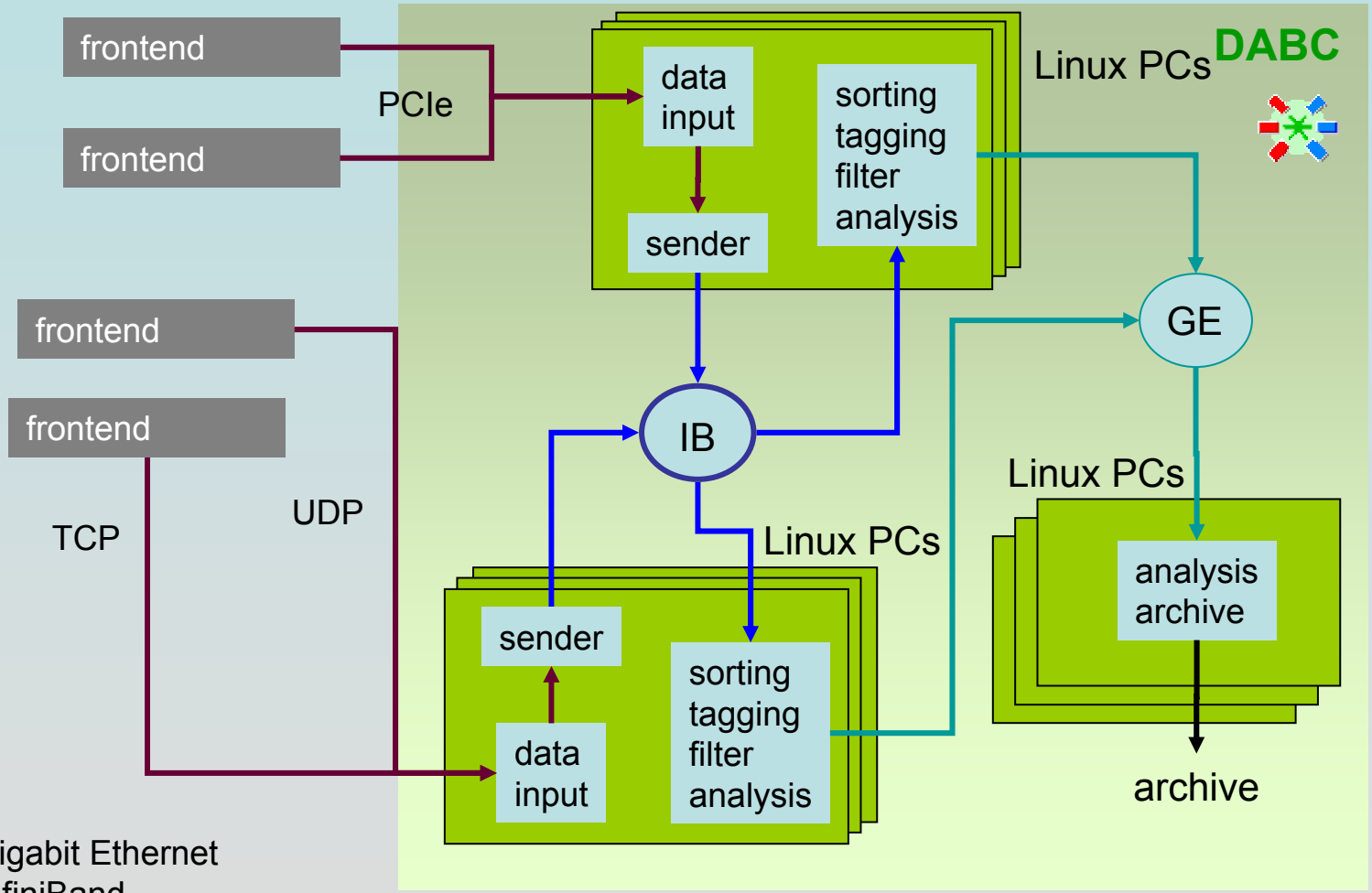
Download via dabc.gsi.de



Controls: No more XDAQ
Plugins: Implementation of applications (programmers)
Applications: Mainly setup or testing programs (users)

* external packages needed

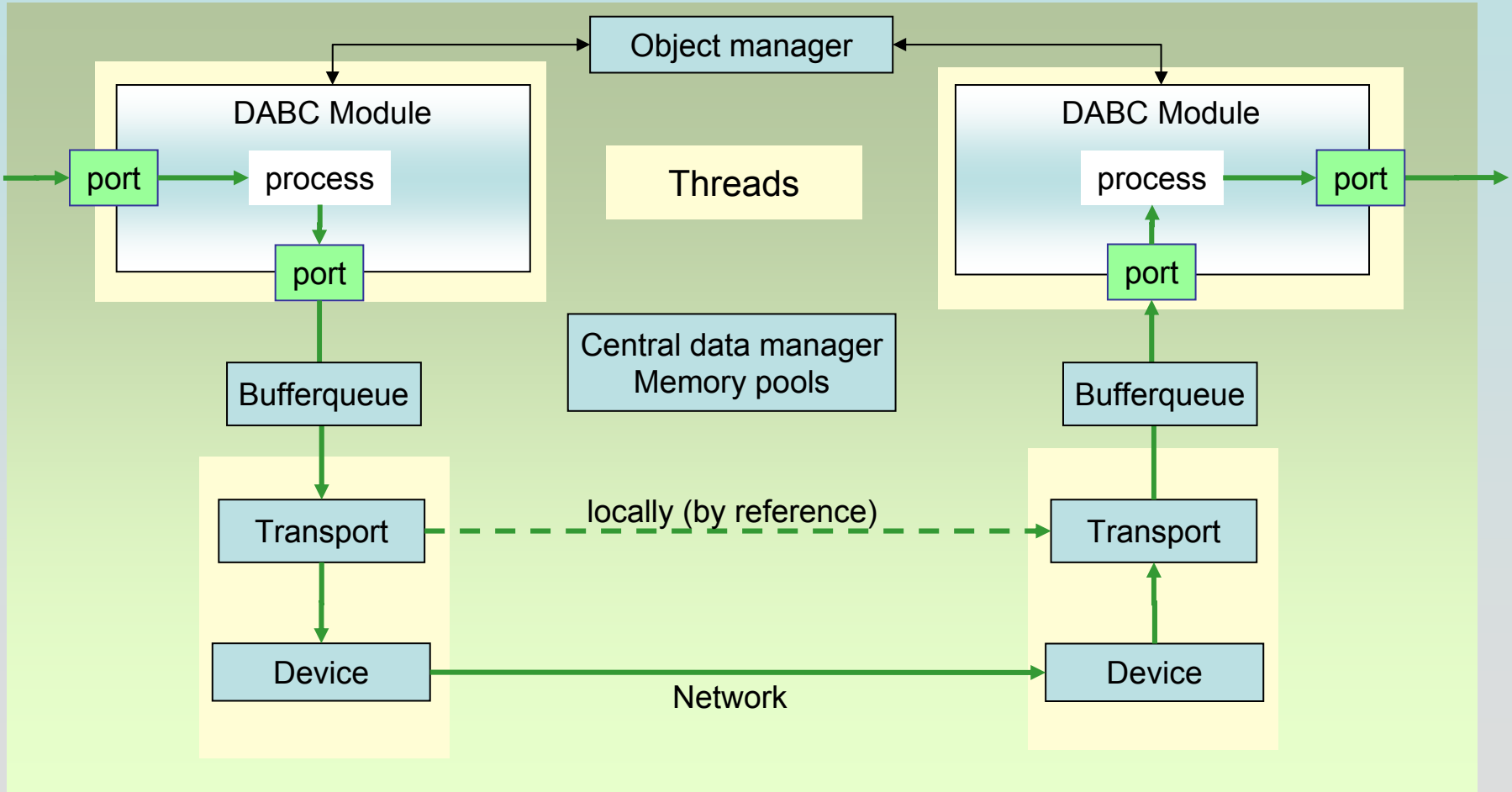
DABC design: global overview



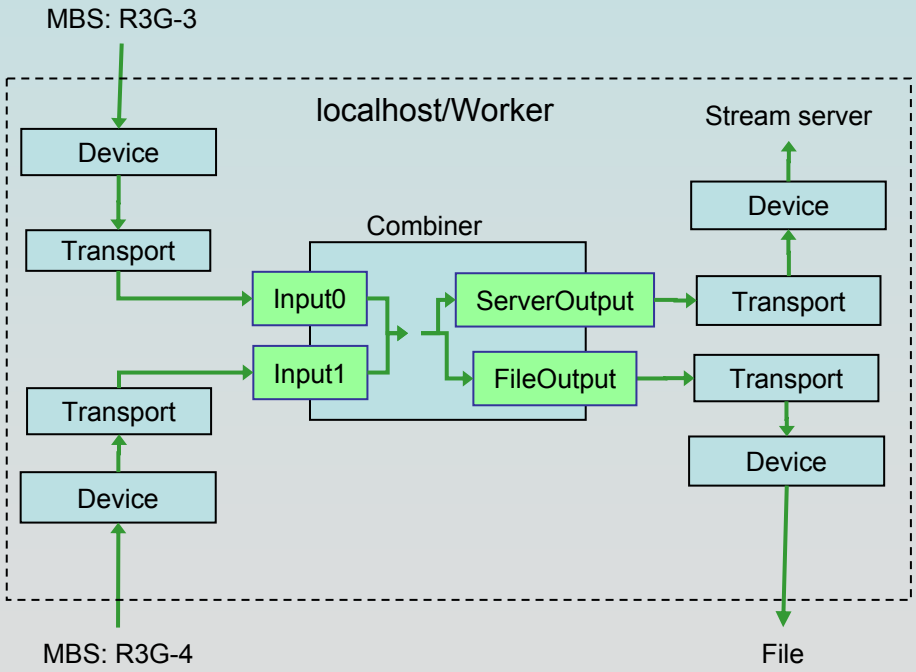
GE: Gigabit Ethernet
IB: InfiniBand

DABC design: Software objects of data flow engine

A *module* processes data of one or several data streams.
Data streams propagate through ***ports***, which are connected by ***transports*** and ***devices***

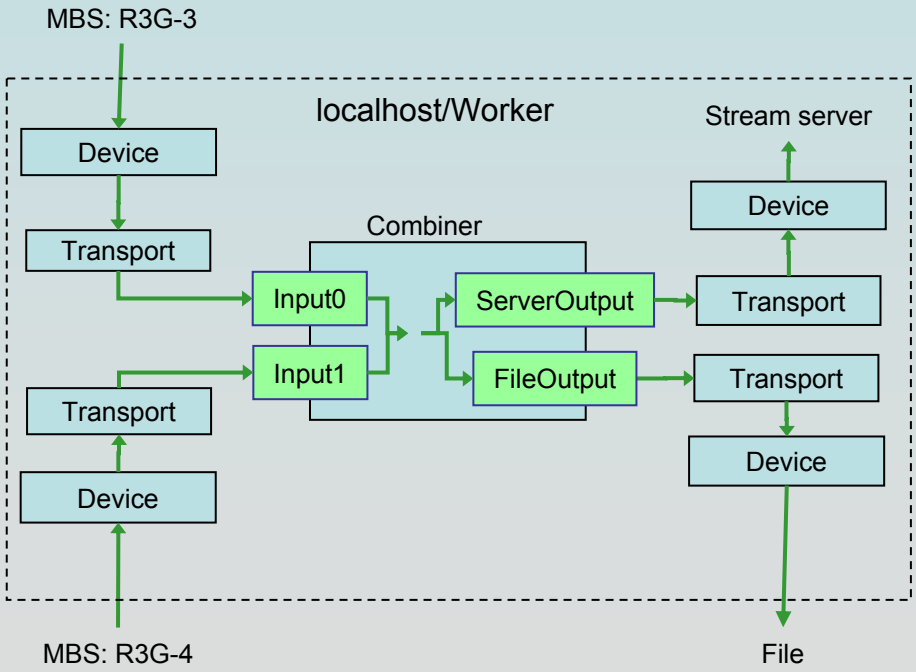


Simple MBS combiner setup



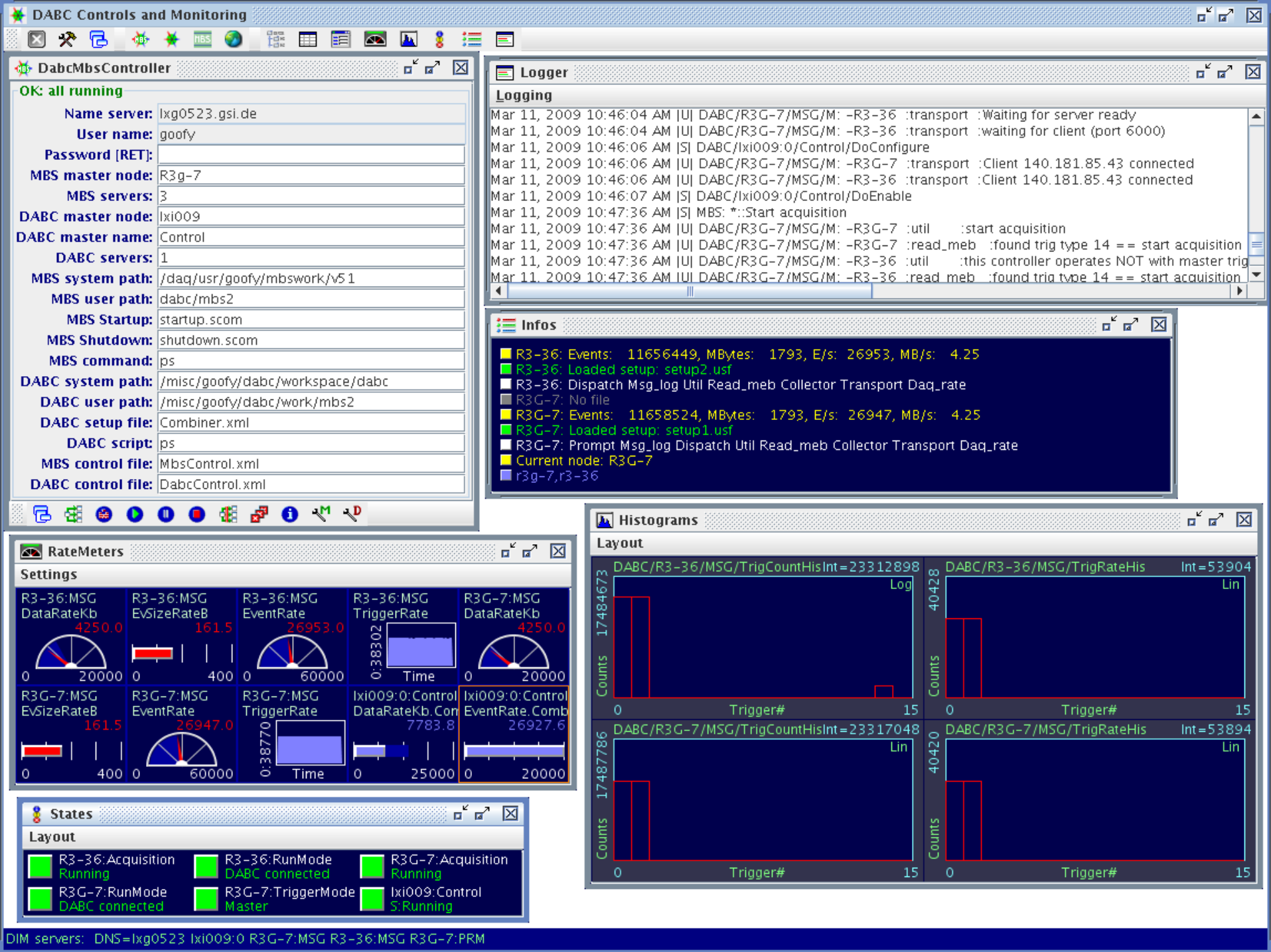
Simple MBS combiner setup

- Context describes executable
- Run specifies libraries and functions
- Application has here no parameters
- Standard device (socket based)
- Module and Port parameters

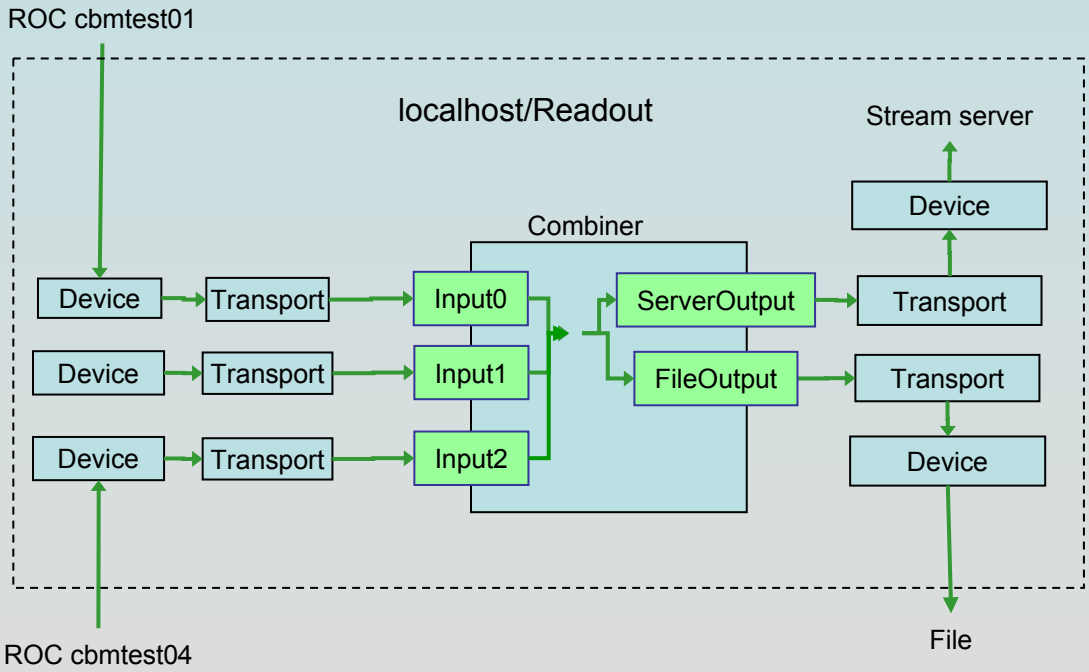



```
<?xml version="1.0"?>
<dabc version="1">
  <Context host="localhost" name="Worker">
    <Run>
      <lib value="libDabcMbs.so"/>
      <func value="StartMbsCombiner"/>
    </Run>
    <Module name="Combiner">
      <NumInputs value="2"/>
      <DoFile value="false"/>
      <DoServer value="true"/>
      <BufferSize value="16384"/>
      <Port name="Input0">
        <InputQueueSize value="5"/>
        <MbsServerKind value="Transport"/>
        <MbsServerName value="R3G-3"/>
        <MbsServerPort value="6000"/>
      </Port>
      <Port name="Input1">
      <Port name="FileOutput">
      <Port name="ServerOutput">
        <MbsServerKind value="Stream"/>
      </Port>
    </Module>
  </Context>
</dabc>
```

MBS

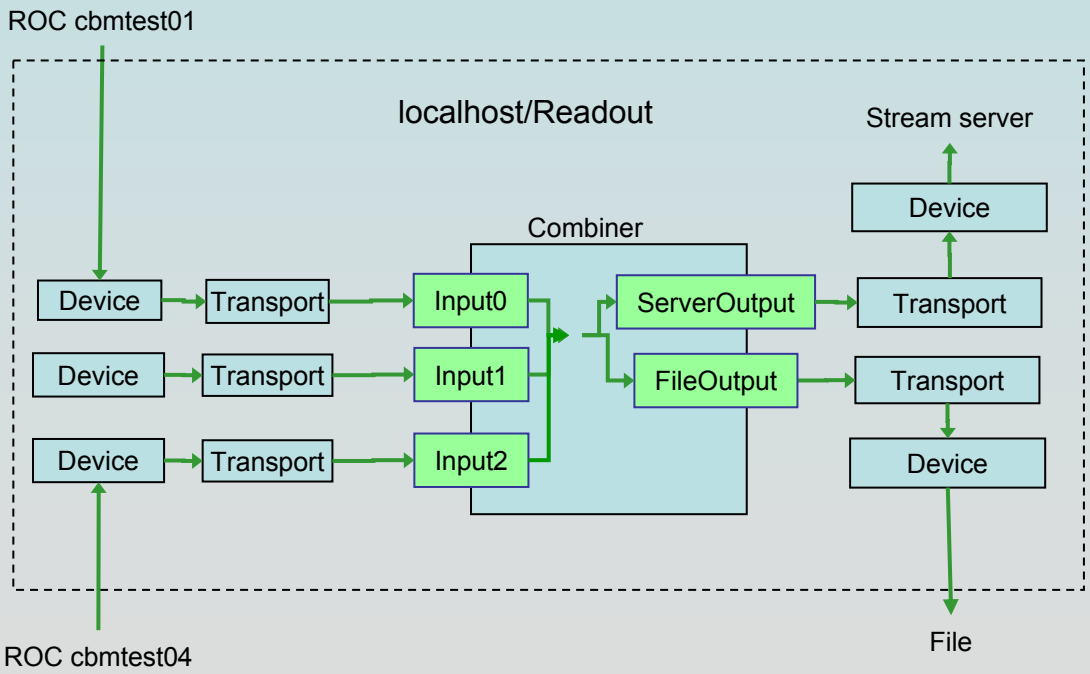

DABC Controls and Monitoring
DabcMbsController
OK: all running
Name server: lxg0523.gsi.de
User name: goofy
Password [RET]:
MBS master node: R3g-7
MBS servers: 3
DABC master node: lxi009
DABC master name: Control
DABC servers: 1
MBS system path: /daq/usr/goofy/mbswork/v5 1
MBS user path: dabc/mbs2
MBS Startup: startup.scom
MBS Shutdown: shutdown.scom
MBS command: ps
DABC system path: /misc/goofy/dabc/workspace/dabc
DABC user path: /misc/goofy/dabc/work/mbs2
DABC setup file: Combiner.xml
DABC script: ps
MBS control file: MbsControl.xml
DABC control file: DabcControl.xml
Logger
Logging
Mar 11, 2009 10:46:04 AM |U| DABC/R3G-7/MSG/M: -R3-36 :transport :Waiting for server ready
Mar 11, 2009 10:46:04 AM |U| DABC/R3G-7/MSG/M: -R3-36 :transport :waiting for client (port 6000)
Mar 11, 2009 10:46:06 AM |S| DABC/lxi009:0/Control/DoConfigure
Mar 11, 2009 10:46:06 AM |U| DABC/R3G-7/MSG/M: -R3G-7 :transport :Client 140.181.85.43 connected
Mar 11, 2009 10:46:06 AM |U| DABC/R3G-7/MSG/M: -R3-36 :transport :Client 140.181.85.43 connected
Mar 11, 2009 10:46:07 AM |S| DABC/lxi009:0/Control/DoEnable
Mar 11, 2009 10:47:36 AM |S| MBS: *::Start acquisition
Mar 11, 2009 10:47:36 AM |U| DABC/R3G-7/MSG/M: -R3G-7 :util :start acquisition
Mar 11, 2009 10:47:36 AM |U| DABC/R3G-7/MSG/M: -R3G-7 :read_meb :found trig type 14 == start acquisition
Mar 11, 2009 10:47:36 AM |U| DABC/R3G-7/MSG/M: -R3-36 :util :this controller operates NOT with master trig
Mar 11, 2009 10:47:36 AM |U| DABC/R3G-7/MSG/M: -R3-36 :read_meb :found trig type 14 == start acquisition
Infos
R3-36: Events: 11656449, MBytes: 1793, E/s: 26953, MB/s: 4.25
R3-36: Loaded setup: setup2.usf
R3-36: Dispatch Msg_log Util Read_meb Collector Transport Daq_rate
R3G-7: No file
R3G-7: Events: 11658524, MBytes: 1793, E/s: 26947, MB/s: 4.25
R3G-7: Loaded setup: setup1.usf
R3G-7: Prompt Msg_log Dispatch Util Read_meb Collector Transport Daq_rate
Current node: R3G-7
r3g-7,r3-36
RateMeters
Settings
R3-36:MSG DataRateKb 4250.0 0 20000
R3-36:MSG EvSizeRateB 161.5 0 400
R3-36:MSG EventRate 26953.0 0 60000
R3-36:MSG TriggerRate 0:38302 Time
R3G-7:MSG DataRateKb 4250.0 0 20000
R3G-7:MSG EvSizeRateB 161.5 0 400
R3G-7:MSG EventRate 26947.0 0 60000
R3G-7:MSG TriggerRate 0:38770 Time
lxi009:0:Control DataRateKb.Con 7783.8 0 25000
lxi009:0:Control EventRate.Comb 26927.6 0 20000
Histograms
Layout
DABC/R3-36/MSG/TrigCountHisInt=23312898 Log 17484673 Counts 0 Trigger# 15
DABC/R3-36/MSG/TrigRateHis Int=53904 Lin 40428 Counts 0 Trigger# 15
DABC/R3G-7/MSG/TrigCountHisInt=23317048 Lin 17487786 Counts 0 Trigger# 15
DABC/R3G-7/MSG/TrigRateHis Int=53894 Lin 40420 Counts 0 Trigger# 15
States
Layout
R3-36:Acquisition Running
R3-36:RunMode DABC connected
R3G-7:Acquisition Running
R3G-7:RunMode DABC connected
R3G-7:TriggerMode Master
lxi009:Control S:Running
DIM servers: DNS=lxg0523 lxi009:0 R3G-7:MSG R3-36:MSG R3G-7:PRM

ROC: Three ROCs to raw file and stream server



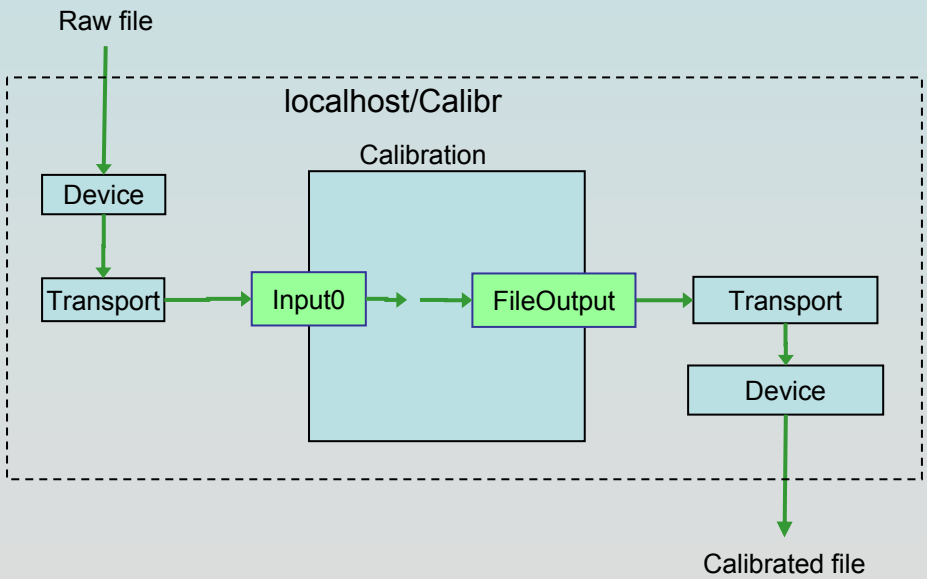
ROC: Three ROCs to raw file and stream server

- Class *Readout* implements *CreateAppModules*
- Application parameters known to all modules



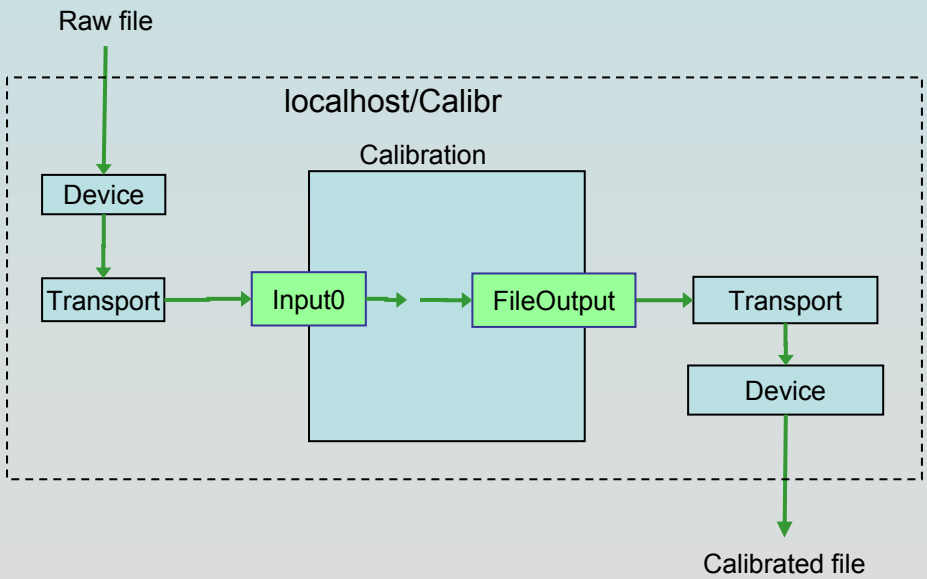

```
<?xml version="1.0"?>
<dabc version="1">
<Context name="Readout">
  <Run>
    <lib value="libDabcMbs.so"/>
    <lib value="libDabcKnut.so"/>
  </Run>
  <Application class="roc::Readout">
    <DoCalibr value="0"/>
    <NumRocs value="3"/>
    <BufferSize value="65536"/>
    <NumBuffers value="100"/>
    <RawFile value="run090.lmd"/>
    <MbsFileSizeLimit value="110"/>
    <RocIp0 value="cbmtest01"/>
    <RocIp1 value="cbmtest02"/>
    <RocIp2 value="cbmtest04"/>
    <TransportWindow value="30"/>
    <MbsServerKind value="Stream"/>
  </Application>
 </Context>
</dabc>
```

ROC: raw input file to calibrated output file



ROC: raw input file to calibrated output file

- Class *Readout* implements *CreateAppModules*
- Application parameters known to all modules




```
<?xml version="1.0"?>
<dabc version="1">
<Context name="Calibr">
  <Run>
    <lib value="libDabcMbs.so"/>
    <lib value="libDabcKnut.so"/>
  </Run>
  <Application class="roc::Readout">
    <DoCalibr value="2"/>
    <NumRocs value="3"/>
    <BufferSize value="65536"/>
    <NumBuffers value="100"/>
    <RawFile value="run028*.lmd"/>
    <MbsFileSizeLimit value="110"/>
    <CalibrFile value="testcal.lmd"/>
  </Application>
 </Context>
</dabc>
```

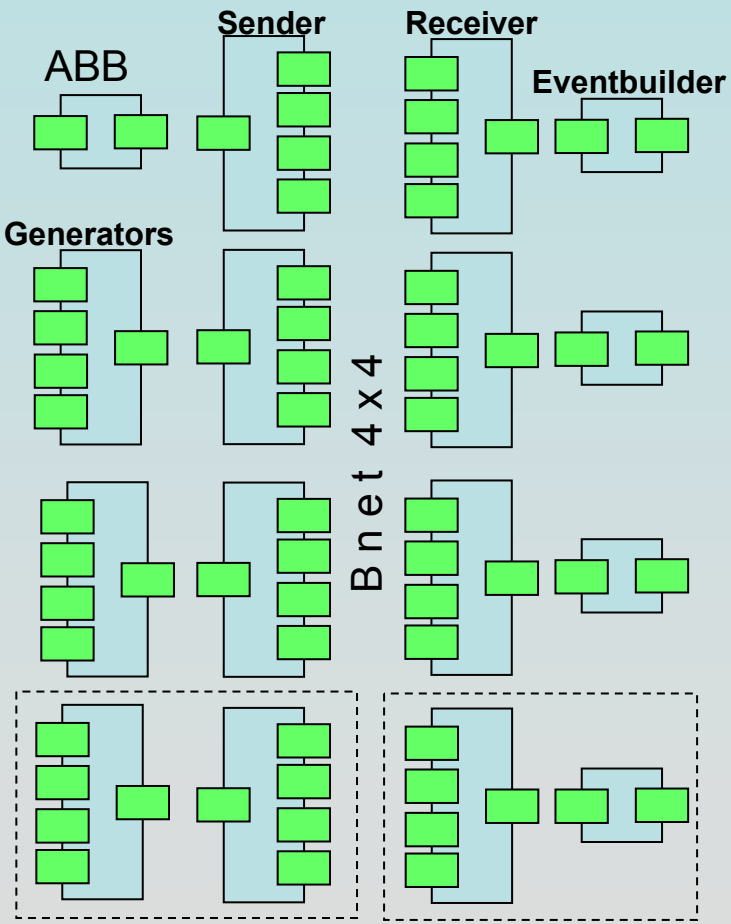
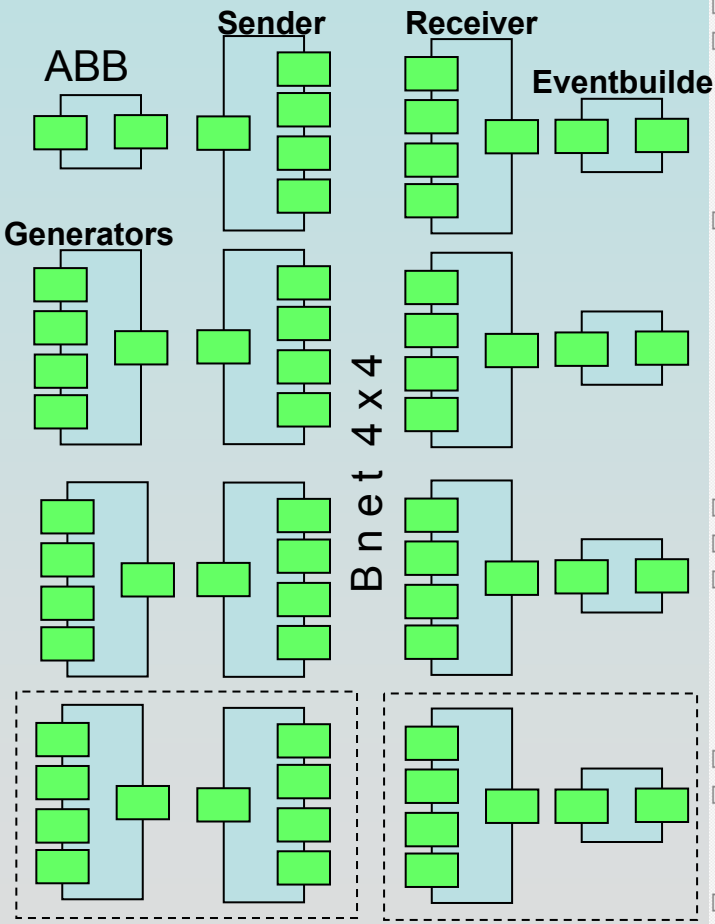


ABB
Sender
Receiver
Eventbuilder
Generators
Bnet 4x4




```
<Context host="master" name="Controller">
  <Run>
    <runfunc value="RunTestBnet"/>
  </Run>
  <Application class="bnet::Cluster">
     <NetDevice value="verbs::Device"/>
  </Application>
</Context>
<Context host="master" name="Worker1">
    <Run>
      <lib value="${DABCSYS}/lib/libpcidriver.so"/>
      <lib value="${DABCSYS}/lib/libmprace.so"/>
      <lib value="${DABCSYS}/lib/libDabcAbb.so"/>
    </Run>
  <Application class="bnet::TestWorker">
    <NumReadouts value="1"/>
    <InputOCfg value="ABB"/>
  </Application>
</Context>
<Context host="node01" name="Worker2"/>
<Context host="node02" name="Worker3"/>
<Context host="node03" name="Worker4"/>
<Defaults>
  <Context name="*">
    <Run>
      <lib value="libDabcVerbs.so"/>
      <lib value="libDabcBnet.so"/>
    </Run>
  </Context>
  <Context name="Worker*">
    <Run>
      <lib value="${DABCSYS}/applications/bnet-test/libBnetTest.so"/>
    </Run>
    <Application class="bnet::TestWorker">
      <IsSender value="true"/>
      <IsReceiver value="true"/>
      <NumReadouts value="4"/>
    </Application>
  </Context>
</Defaults>
```

Only one ABB

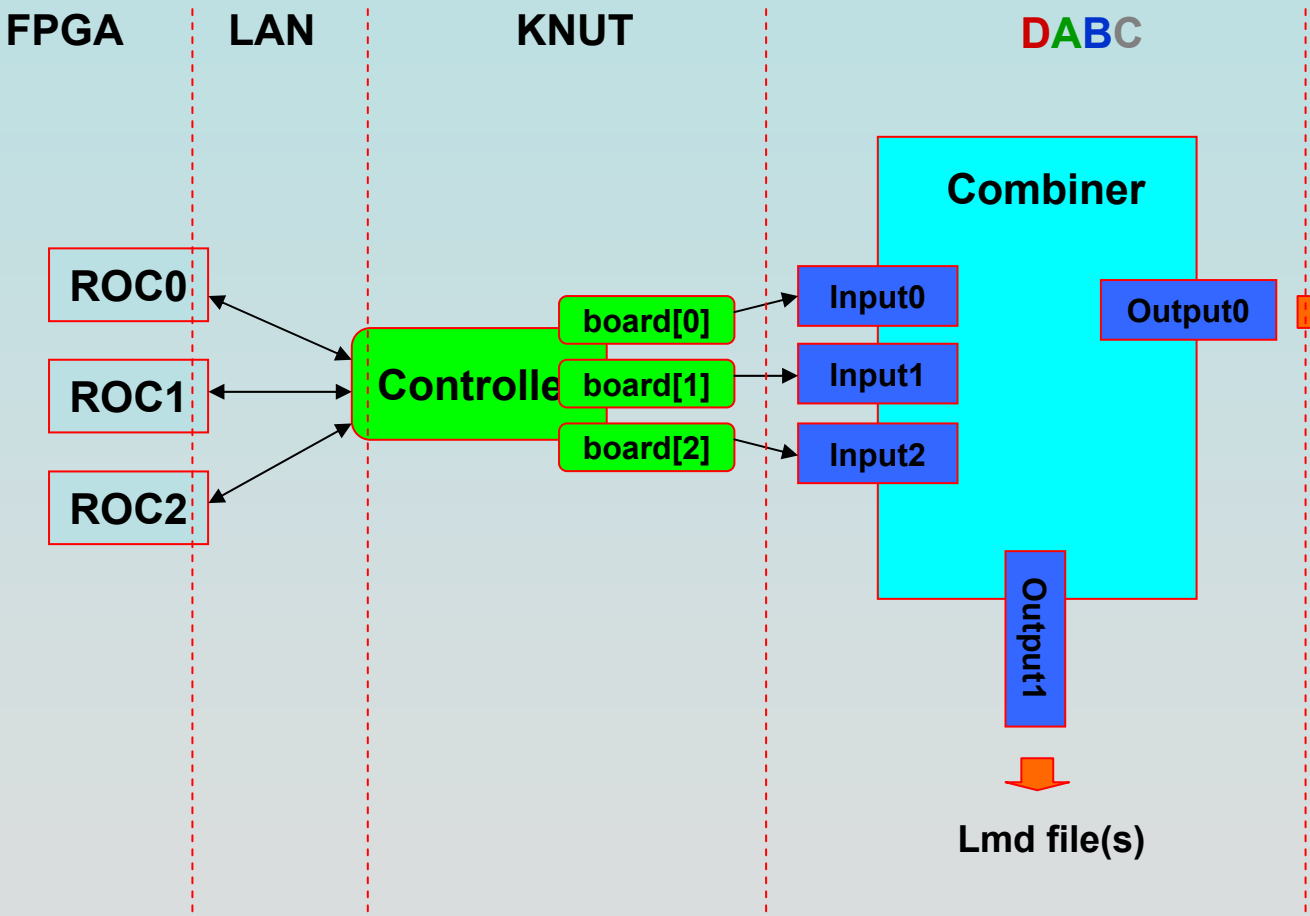
Values for all nodes

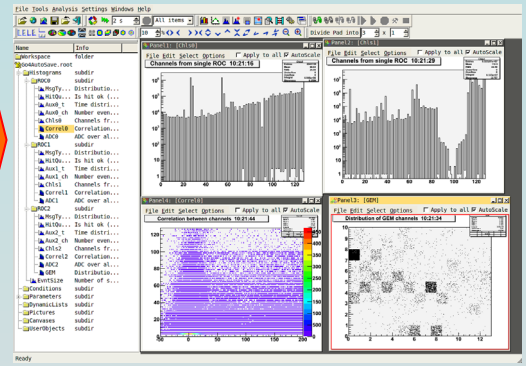
Values for all workers

Three generators

Four readouts each

DABC in September 08 beamtime





Observations during beamtime

- This was the first usage of DABC (beta) in real work.
- First run of DAQ with 3 ROCs and sync messages half an hour before beam.
- 24 GByte of data stored in LMD files.

- Of course, lot of problems:
  - lost of sync markers;
  - lost of epoch markers;
  - only ~1 MB/s data rate per ROC;
  - unsynchronized drop of data at high beam intensity;
  - influence of nXYTER control gui on DAQ.

- Most problems now identified and solved

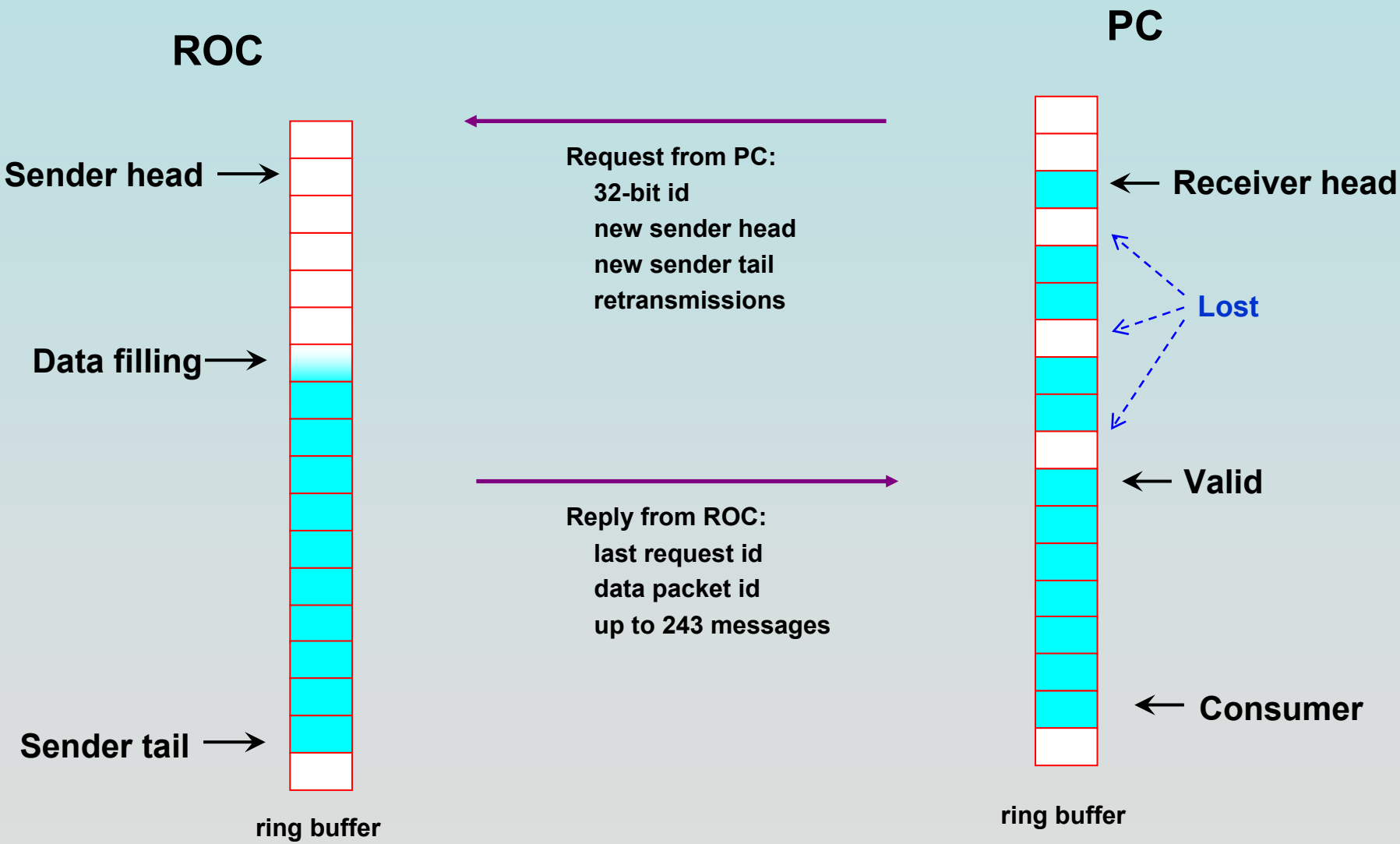
Redesign of ROC and PC software

- On PowerPC
  - decoupling of three major tasks:
    - data taking
    - data sending
    - serving control requests
  - use ~110 MB for buffering data
  - fix half-duplex problem with Ethernet driver

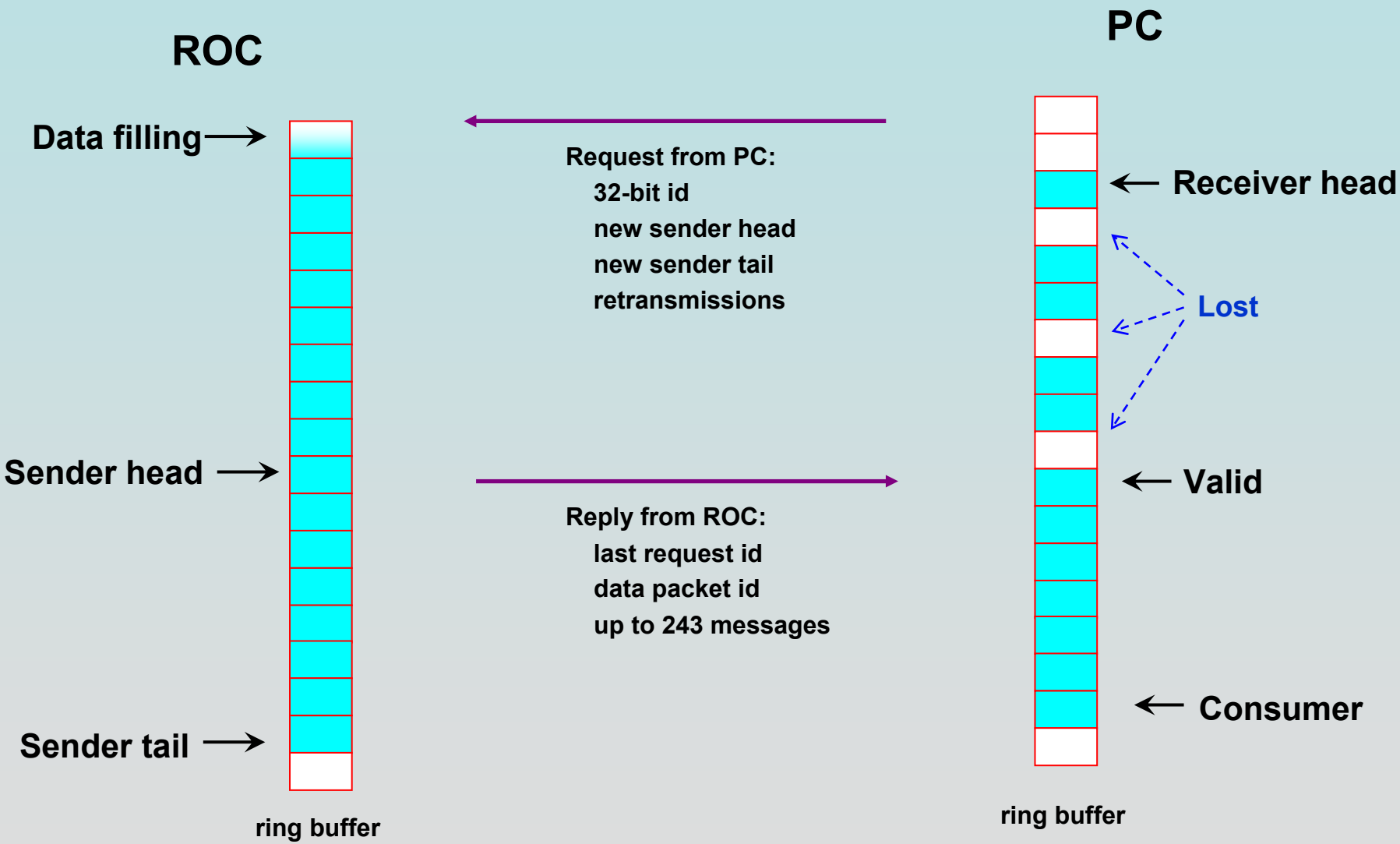
- Fully redesigned UDP-based data transfer protocol

- On host PC:
  - fault tolerant, blocking peek/poke control interface
  - blocking and non-blocking mode for data taking
  - daq suspend logic
  - first approach for master/observer modes of operations

New request-reply transport protocol



When data taking faster than sending



Integration with DABC

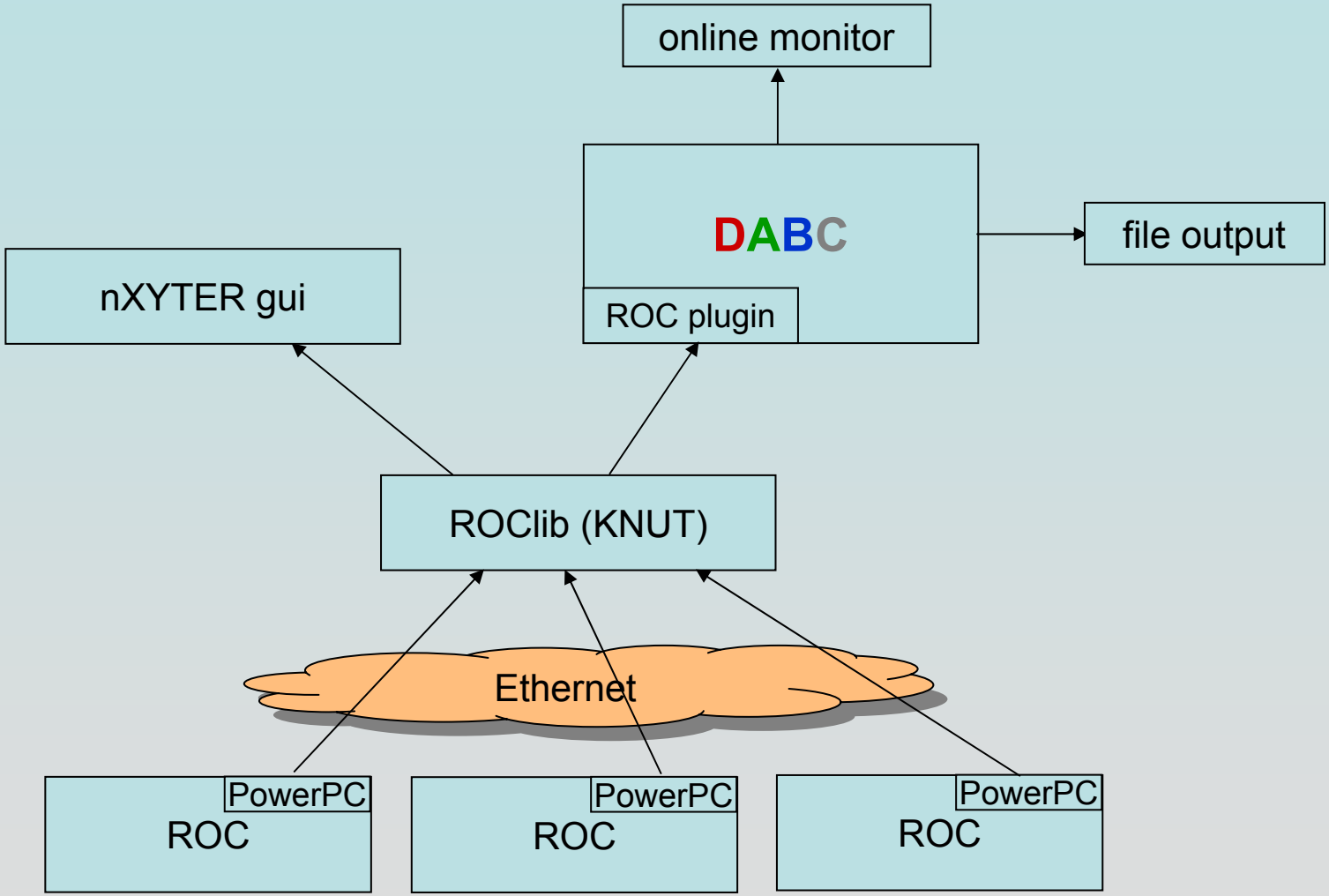
- roc::Transport and roc::Device classes to encapsulate functionality of KNUT SysCoreBoard and SysCoreControl classes
- SysCoreSorter class (later was moved into KNUT) to perform time sorting (and errors corrections) of data stream from ROC
- roc::CombinerModule for packing of data from several ROCs into mbs subevents – was used to produce raw data in beamtime
- roc::CalibrationModule to perform time sorting and merging of data streams from several ROCs
- roc::ReadoutApplication for configuring and instantiation of all necessary components to readout data from ROCs

Further developments

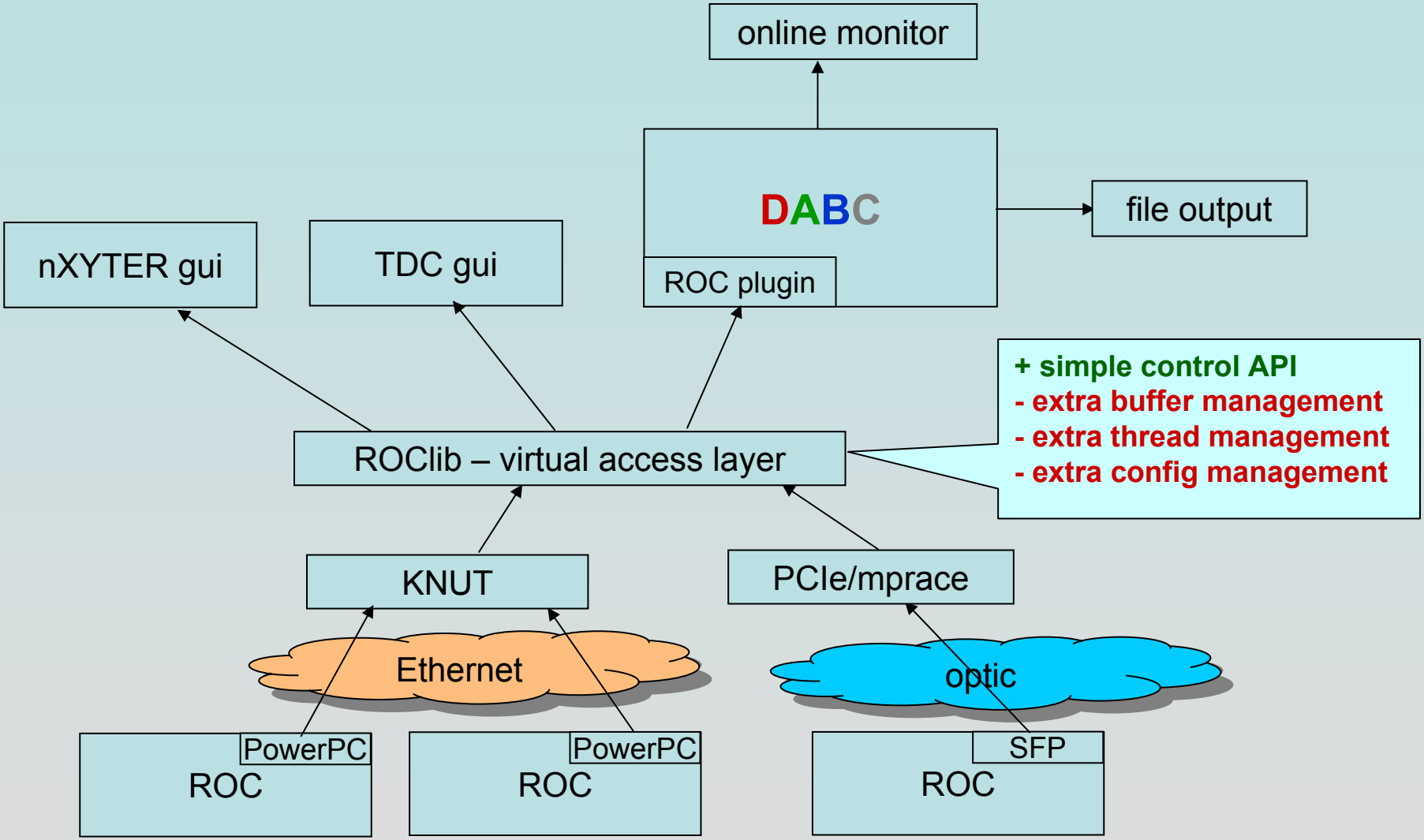
- Shot-term tasks
  - Fix problems with faster data readout from FPGA
  - Use DMA mode of Ethernet driver
  - gain performance up to 12 MB/s sustained
  - modify peek/poke logic that only register-like operations are performed

- Mid-term tasks
  - multiple controlling, single readout channels (several applications)
  - control and readout via PCIe board
  - common access layer for Ethernet and optic channel
  - application layer (transactions) for specific front-ends

Access layer – current view



Common access layer (variant)



DABC as common access layer