Online processing in the ALICE DAQ
The Detector Algorithms

Sylvain Chapeland

Vasco Barroso

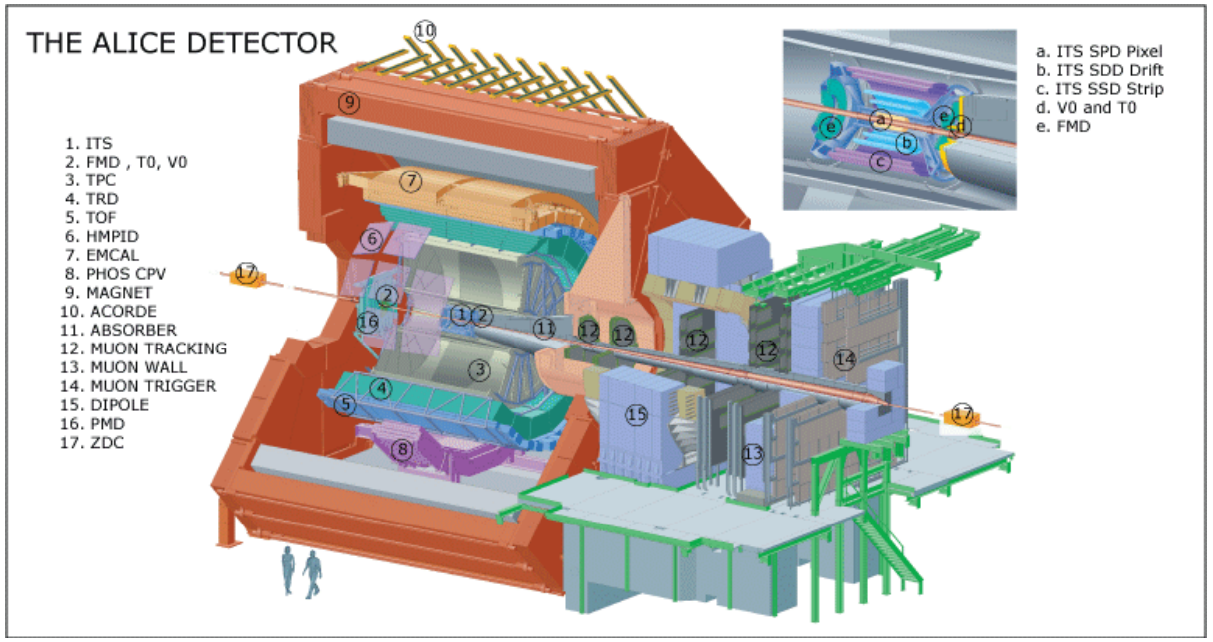
ALICE DAQ

Outline

- Context
- Detector Algorithms framework
  - Architecture
  - Interfaces
  - Databases
  - Libraries
- Deployment
- Current Status
- Performance considerations
- Perspectives

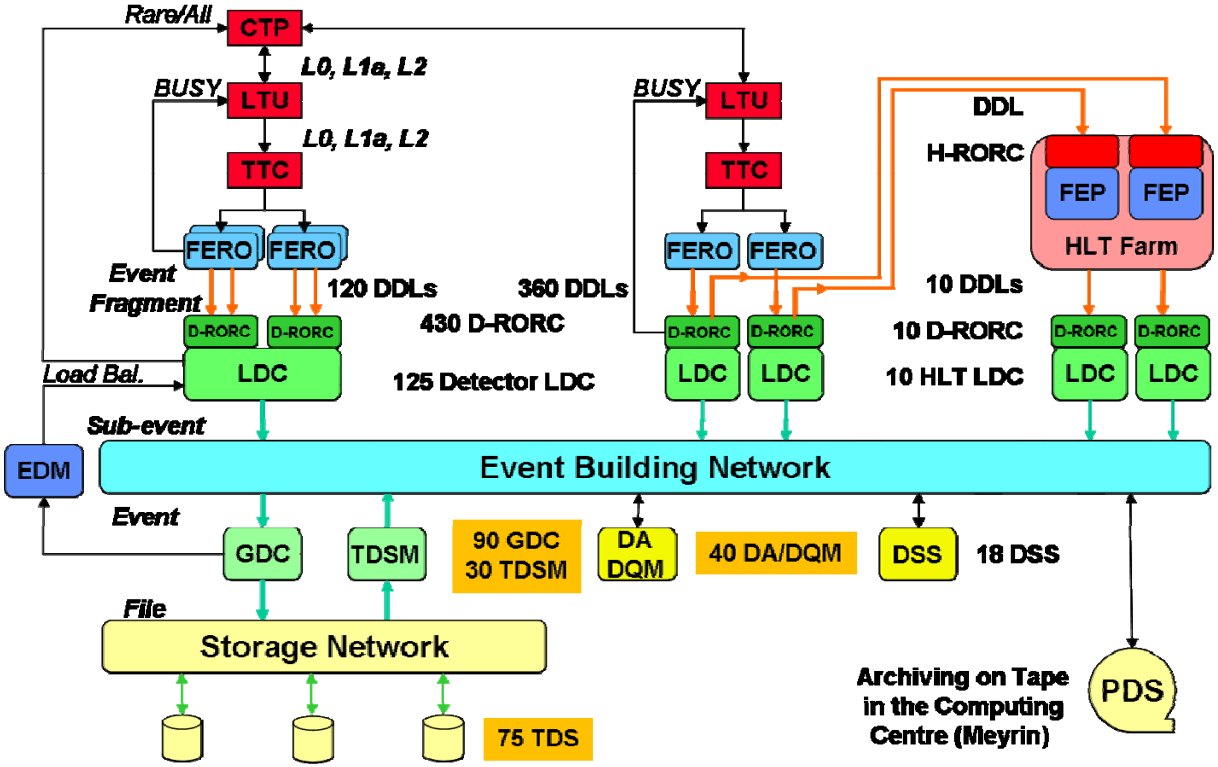
Context: ALICE

- 18 sub-detectors
- Need calibration
  - configuration
  - components vary with time
  - conditions change



- Large data volumes / statistics sets involved
- Should be done online
  - needed to configure some detectors electronics
  - results used in reconstruction

Context: ALICE DAQ



"Commissioning the ALICE Experiment "

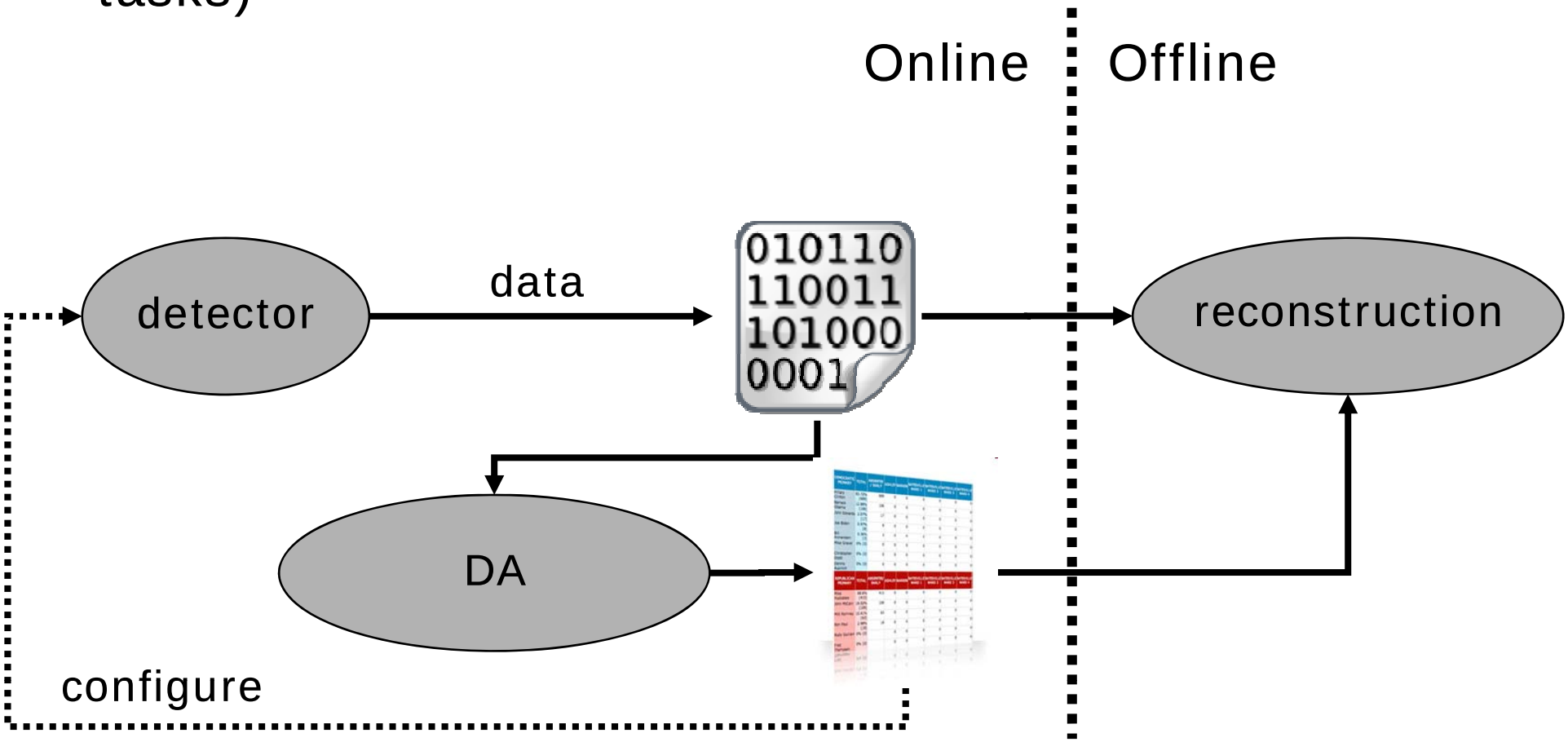
Pierre Vande Vyvre

Thursday, 16:30, Club D

- Full-duplex link to sub-detectors Front End Electronics
- Links/nodes parallelism
- Data-grabbing API
- Standalone / global runs

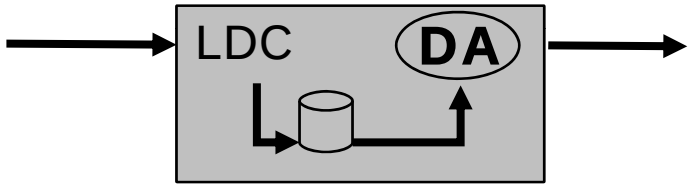
Context: Detector Algorithms (DA)

- DA = Calibration task running online provided by the sub-detectors teams (exclusive tasks or background tasks)

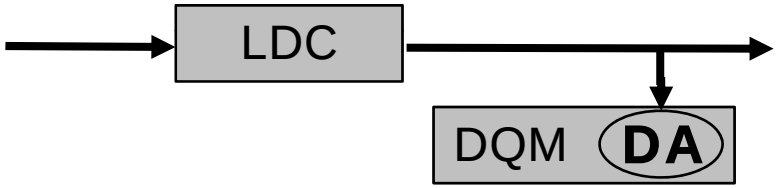


DA framework: Architecture

EXCLUSIVE

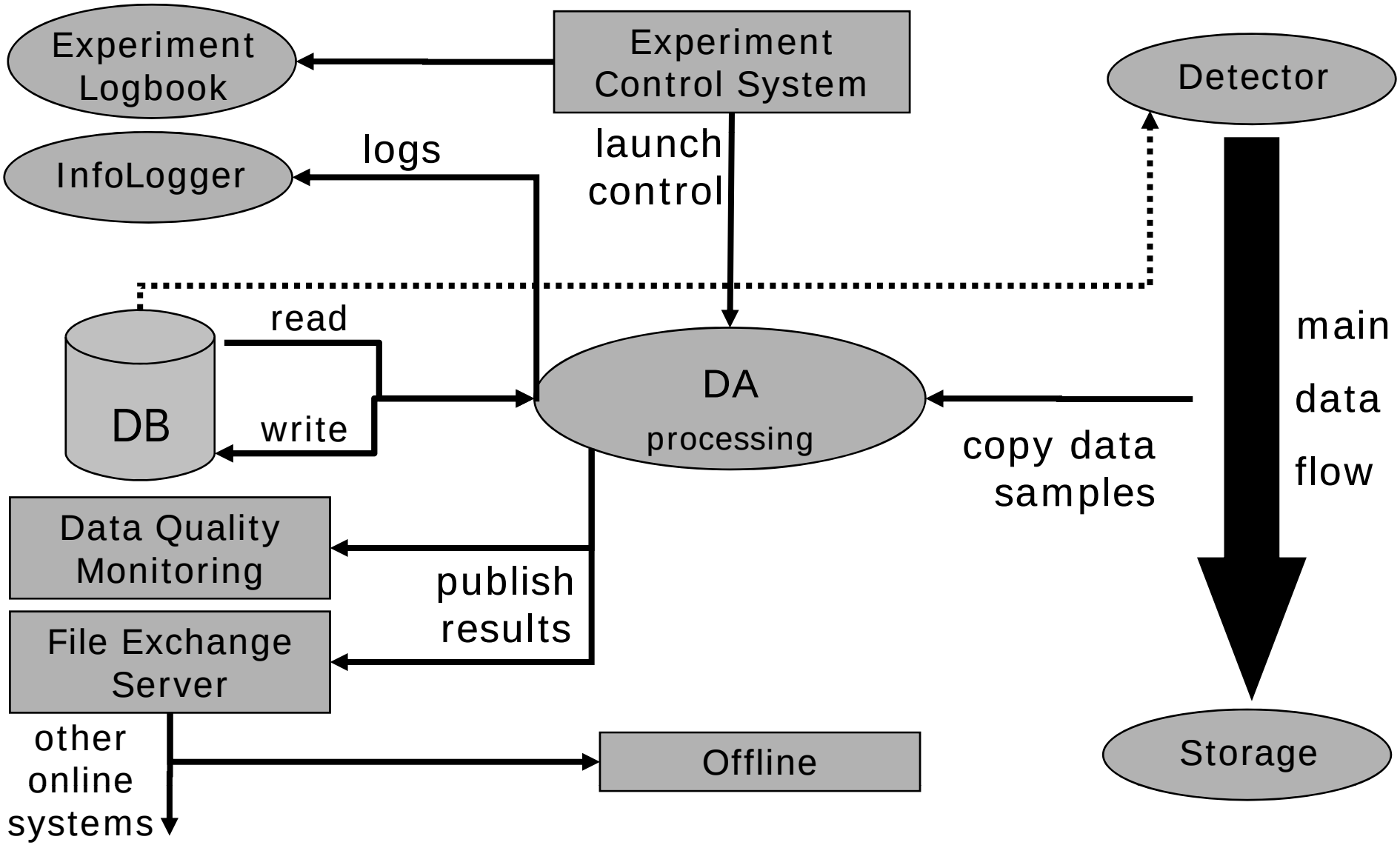


BACKGROUND



- **'LDC DA':**
  - dedicated (short) standalone run
  - data recorded locally (in parallel)
  - DA launched at end of run
  - data processed directly on the LDC (easily exportable to FEE)
  - e.g. pedestal
- **'MON DA':**
  - dedicated standalone or global physics runs
  - data samples picked-up from the normal data flow (non-intrusive)
  - processed on dedicated machines
  - e.g. dead channel mapping
- Events used: depends on DA
  - Physics triggers
  - Calibration triggers (laser, pulser, ...)

DA framework: Architecture



DA framework: Interfaces

- Experiment Control System:
  - control/launch, define runtime environment (e.g. data source)
  - Logging + ALICE Electronic Logbook
- File Exchange Server:
  - export to Offline Shuttle
    - Post processing & archiving
  - export to other online systems

"The ALICE Online Offline Framework for the Extraction of Conditions Data"

Chiara Zampolli

- Data Quality Monitoring
  - AMORE library
  - export to physics monitoring facility
    - display, check, reuse results

"The ALICE data quality monitoring"

Barthélémy von Haller

DA framework: Interfaces

- DAQ detector DB
  - input parameters
  - results (e.g. to be reused by other DAs, FEE scripts)
- Local storage:
  - working directory: temporary files
  - persistent directory: files saved for later local usage
- DAQ Data flow:
  - LDC DA: local recording
  - MON DA: data-grabbing per role, per detector, per trigger type

DA framework: Databases

- daqDetDB
  - MySQL DB
  - Tcl/Tk GUI browser
  - Command line + C API to get/store/list files
- File Exchange Server (FXS):
  - MySQL DB for indexation
  - Files on disk, sFTP/rssh for file access
  - Command line + C API to store files
- Logs (Tcl/Tk GUI browser) + ALICE Electronic Logbook (Web Interface)

Deployment

- Many sub-detectors teams: single deployment procedure needed

1. Appropriate run types defined on ECS
2. Detectors provide DAs
3. DA validated on test platform (same than at runtime)
   - Automatically checkout, build, validate and report
   - Data file as input (real data / simulation)
   - Check I/O interface compliance
   - Benchmark
4. Deployed at experimental area and tested standalone
5. Put into production

Deployment: packaging

- DA code is stored in ALIROOT (Offline Software)
- Static executable is built => no runtime dependency
- Self documented RPM:
  - Contact
  - Link to documentation
  - Reference runs
  - DA type
  - Number of needed events
  - Input and output files
  - Used Trigger types

Current Status

- In production at experimental area since Dec 2007
- Collection and integration of the numerous DAs during the 2008 ALICE cosmic runs
- 30 DAs currently in production
- 1 DA per CPU core to guarantee resources
- 11 000 runs, 33 000 files, 80 GB of calibration data exported

Performance considerations

- DAs optimization whenever possible:
  - better performance => more stats, shorter dead time
- However, calibration tasks are often CPU demanding
- How to optimize:
  - DAs code
  - Compilers
  - Hardware

Performance considerations

- Compilers: ICC provides 10-20% faster DA execution than GCC

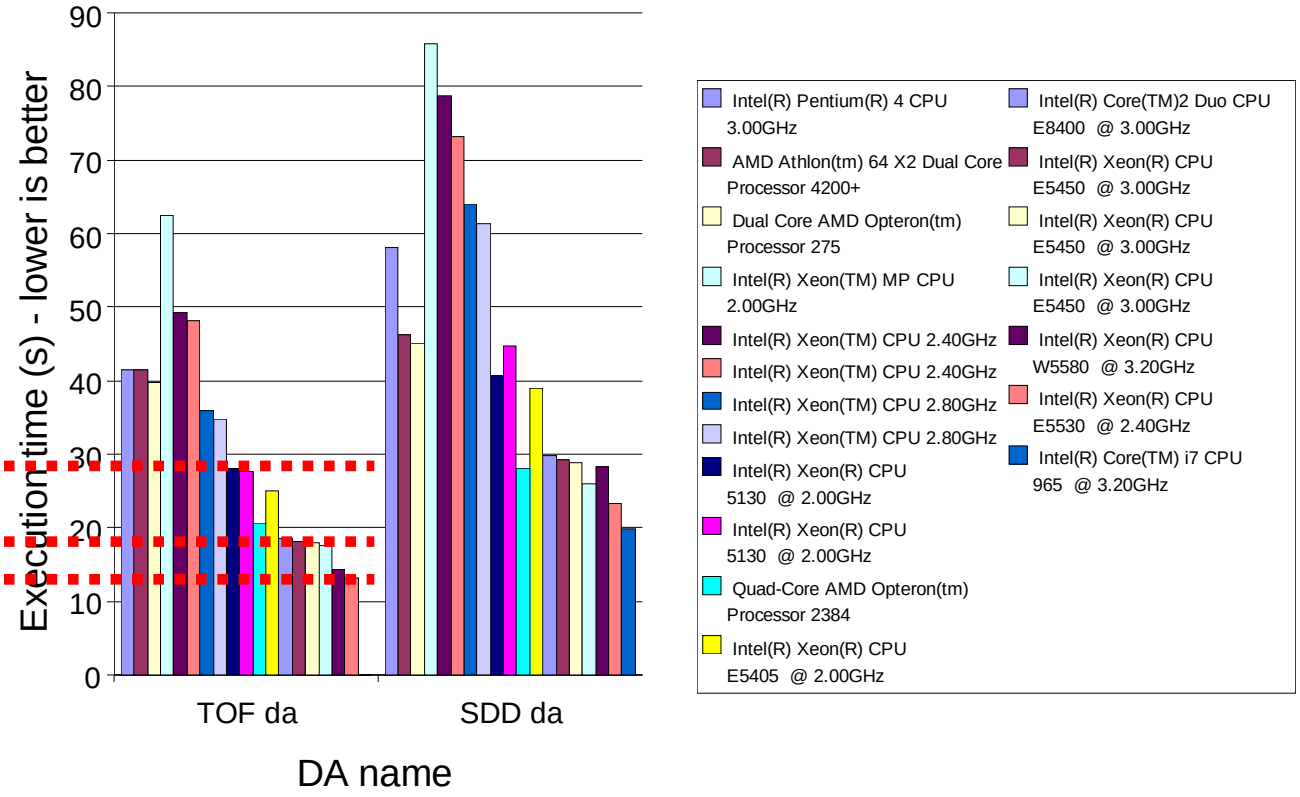
- Hardware:

Latest CPU generation will provide big performance gain

P2 machines

typical 2008 server

Nehalem



Perspectives

- Hardware expansion for 2009 beam
- Distribution of events across processes for parallelism
- Launching mechanisms
- Dynamic libraries
- More DAs

Optimizations

Perspectives: conclusion

- DA framework in place and proved to work in production
  - common interface was worth the effort
  - all calibration requests so far fit in the architecture
  - mature implementation, can now deal with fine tuning and optimizations
- System ready, will scale up with needs
  - Sub-detectors are working on more DAs
  - Higher calibration demand expected with LHC startup