Clear Requirements and Goals

Robust Architecture

Effective Project Management

Strong [Coding] Practices

Continuous Integration and Continuous Deployment (CI/CD)

Documentation

Security (important for GRID comp.)

Performance Optimization

User-Centric Design

Collaboration and Communication

Adaptability and Flexibility

Sustainable Development

Exp - Collider (RHIC, LHC, EIC)

SANPC2024 Workshop

Key topics

- Sustainability of the software development and maintenance model
  - *Core team - contributor in the cloud* paradigm
  - Dependencies - in and out of control of the project/experiment
- Role of CS/IT divisions
  - “IT as a service” vs “IT research focus”
  - Spotlight on host labs
- Role of funding agencies
  - Distributed computing - distributed software development vs. core team funding
  - Accounting - direct vs. indirect funding for software development
- Development and retention of the workforce
  - Key drivers for computing scientists (often physicist-turned-cs)
  - Career development - paths
- New directions - new methods
  - Novel tools enabled by ML - applications to optimizations of detector simulations and data analysis
- Lessons for the future
  - Recap where to improve - clarity in spending and efficiency of $
  - Community engagement, cross-talk, common cross-projects (including theory-experiment)

“Typical” collider experiment - computing infrastructure

“Typical” ⇔ largely common approaches

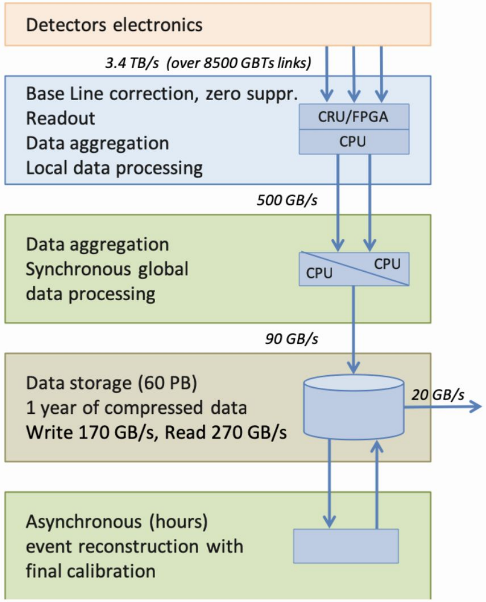
- **Data pipeline** - a generalization based on ALICE example

  (Time / Decreasing size ↓)
  - Detector FEE (computing starts already here!)
  - First level data reduction (e.g., ZS)
  - *Local* event reconstruction (hit/cluster level)
  - *Global* event reconstruction (track level)
  - Additional data reduction (if needed/lossy)
  - PID (particle hypothesis level)
  - Storage (different formats/skims) - improved reco/calib iterations - physics analysis
- Important improvements/variation: **continuous/streaming readout, iterative approach to event reconstruction**

Offline-online integration: same algos online and offline

- **Collider experiment - software project(s)**
- Collaborations O(100) - O(1k)
  - Efficient management structure essential (computing a separate sector within these structures)
- Example: Sheer lines of code (ALICE example / no deps)
  - Online-Offline (reconstruction): > 1.1M lines
  - O2 Physics (data analysis): > 650k lines
- Workforce - example:
  - Core team of O(10) (may include IT FTE)
  - Contributors distributed @ various expertise levels >100
  - Physics analysis contributors (global) O(100)
- Self-built tools for distribution / installation / running
  - Scripts / tools (e.g., aliBuild), conternerization
- Use of mainstream development tools (e.g., GitHub/Lab)
- Number of common tools used but often different approaches determine different direction (also historical aspects play a role)

‘Example’ - ALICE



Not discussed: distributed offline computing - GRID; tier structure; etc

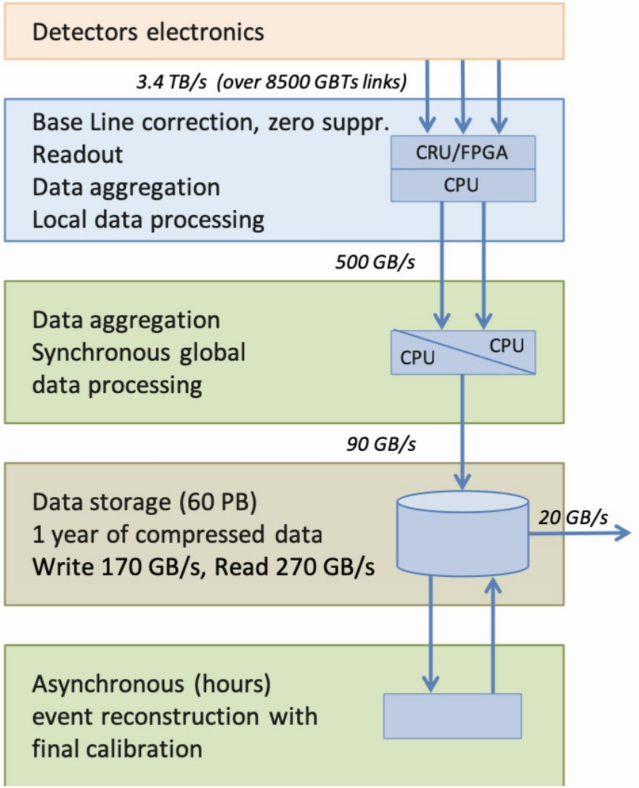
"Typical" collider experiment - computing infrastructure

Main challenges - broad strokes:

"Typical" ⇔ largely common approaches

- Most steps alignment and calibration sensitive
  - Needs a dedicated team / skilled / in training WF
- Lossy vs. lossless compression
  - Needs experts (also knowledge of the detector)
  - No rapid changes, needs R&D, maintenance
- Time to data analysis
  - Could be O(Year) - EIC aim: 2-3 weeks ⇔ improved automation
  - Scales with complexity of the detector system
  - Efficient ops need dedicated team of calibration experts
- MC generation / simulations
  - Logistics, expert team, monitoring
- Reprocessing vs. resources vs. new data
  - Logistics, storage, CPU (reuse GPU)
- Management of online and offline software
  - Strong integration of online-offline processing (same algos; aim for seamless DAQ-to-analysis workflow)
  - Needs skilled FTE operations
  - Knowledge of the dependencies
  - R&D - continuous evolution / adaptation to hardware
  - Packaging (script, conteneralization, OS compatibility etc)

'Example' - ALICE



"Typical" collider experiment - computing infrastructure

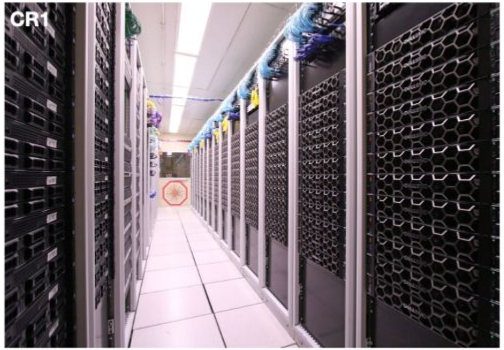
"Typical" ⇔ largely common approaches

Small revolution: continuous readout (aka streaming)
=> new data challenge and new software paradigm

Setting up the stage: 'Example' - ALICE

20 out of 100 PB deployed
joint effort with the ALICE IT team

200 First Level Processors (FLP) in CR1



1) Readout of detectors and raw data processing
(e.g. TPC baseline corrections, ZS)
Data compression
3.5 TB/s → 600 GB/s

250 Event Processing Nodes (EPN) with 2000 GPUs

⟺ **Disk Buffer**

2) Synchronous processing
⟹ Event/time frame building
⟹ **Online** reconstruction and calibration

99% on GPUs

3) Asynchronous reprocessing
⟹ **Final** calibration and full reconstruction

80% on GPUs

4) Permanent storage
Data compression
600 GB/s → < 100 GB/s

Not discussed: distributed offline computing - GRID; tier structure; etc

Sustainability of software development and maintenance

- Collider experiments are multi-year/decade long enterprises
  - Need for sustained support of overall computing infrastructure (see WLCG, MoUs FAs-CERN etc)
  - Even after the experiments mission complete need for data preservation plans & resources
  - Typically driven/covered by the host lab(s) - especially after collaboration dissolved
- Needs a continuous influx of juniors
  - Rotation, training of new experts - competition with industry…
- Requires clear career progression for physicists-turned-CS
  - Important for retention of skilled FTE / know-how - vital need
  - Can be critical especially for online systems software development/maintenance
  - Common standards / best practices enable flow of juniors between experiments
- Concern: reliance of single-point-of-know-how (non-replaceable experts)
  - Can be overbearing for the personnel
  - Needs long term solutions / planning rather than ad-hoc injection of funds
- Dependencies vs. re-inventing the wheel
  - Obvious but important: experiment dedicated software relies on GNU/GPL external packages, OS and its evolution ⇔ there is an assumption that these will be always available and will evolve favorably
  - Important aspect: a critical decision - do NOT re-invent software that already exists (easy for generic/well designed pgks)
- Building software for Tomorrow’s experiment with Today’s tools inadequate
  - ePIC in ideal situation
  - Ongoing experiments: incorporation of “new and better” requires study and evaluations - in time critical periods this is often impossible / dropped in favor of custom quick fixes
  - Essential to value basic work on software development as early as possible - in particular: strong positive long-term net benefit for the experiment when building up the expertise within juniors

Organizational aspects - Role of CS/IT departments

- CS/IT departments carry a reservoir of expertise / knowledge / applications
- CS/IT departments carry a capacity for long-term career development
  - Retention of talent (otherwise too heavy-$ on experiments)
- Good alternative to industry for continuation of critical expertise
  - Still attracts science enthusiasts (compensation competitiveness regional)
- R&D towards advancement of CS not always aligned with experiments priorities / needs
  - Unexercised opportunities (@host lab IT departments: disconnect with the primary [science] mission)
  - Examples of failed “frameworks of everything” vs. success of ROOT (experiment supported; only later to evolve into / accepted as the CERN supported team - community has chosen functionality over complexity/abstraction)
  - Good examples of experiment driven frameworks ⇔ success because of the strong line to operations
  - Typically: experiment internal physicists-turned-CS take the main dev. tasks but also those that need less domain knowledge (input from CS needed but not always possible [time/cost issue])
- New directions (CERN ~2022)
  - “IT as a service” - understanding the main task is the support of the science program
  - Embedding of IT/CS FTE into experiment (months-years)
  - Targeted hires recognizing exp. needs - often addressing *common* needs
  - Special advisory committee - good cross-division/experiment cross-talk with IT:
    - Selected experts from experiments, accelerator, IT meet regularly / discuss / recommend to mgmt

Development and retention of workforce

- Likely one of the key issue: definition of clear career paths / progression for physicist-CS crossover
- Experiments rely on a dedicated “core teams”
    - Unique expertise in key areas such as alignment, calibration, GPU processing, software (from within and outside), MC simulation packages, distributed computing (GRID) … **with intimate knowledge of the experiment, physics program, and urgency**
    - First motivation is to support science - compensation @ the 2nd place .. but reality forces
- Support for core teams essential
    - Workforce: support for education and development
    - Stability: Good developments take time (quick fixes come haunting sooner or later)
- Generic question: How to fund projects/experiments allowing for…
    - Good cycle / overlap of junior and senior software devs.
    - Continuous R&D, testing new solutions, adaptation to new hardware
    - Enabling cross-collaborative efforts on common toolkits / approaches

Funding / Role of funding agencies

- Funding of [host;facility] labs as an efficient way to provide software dev. support
  - This includes infrastructure - local (concentrated) mostly most efficient
  - Including contributions to computing (as opposed to soft development) on par with groups participation (e.g. participation in CERN experiments requires computing or $ contribution per M&O-A member - FA level MoUs)
  - This includes resources for data preservation
  - Embedding of CS into university research group beneficial to the overall projects
- Need for allocating/allowing for additional funding within the projects and experiments to tackle software development specific tasks - including hiring of CS/IT within exp. projects
  - Currently difficult to hire skilled CS to a specific task within collaboration
  - This is not always practical - funding necessity for domain expertise still needed (e.g. data and analysis preservation is a requires domain knowledge - not solely tools and workflows)
- Clarity on *innovation* vs. *applications of innovation(s)*
  - Support for development of applications important (as opposed to advancement of CS) - especially true for ML-area
  - Even if innovative solution at hand still significant effort / time needed for production quality
- Enabling cross-talk, cross-pollination
  - Many elements/methods of the toolkits are common between experiments
  - Enable common/joint multi-exp software oriented projects (?)
  - Enable theory-experiment collaborations (what’s the plan?) on software development

New directions

- ML - Cannot afford to ignore ML and its rapid growth
  - Efficient detector response simulations
  - Unbinned observable analysis
  - Improved inference - wholistic approach to event reconstruction and data analysis
- Theory-experiment cross-developments
  - Experimentalists good at efficient framework build up - know what’s practical and useful
  - New culture - new paradigm of physics extraction from data (e.g. Bayesian analysis)
- Software-hardware infrastructure optimizations
  - Homo- vs. heterogeneous computing (off load to GPU certain processing)
  - Software development evolves given new hardware capabilities - this needs R&D, extensive testing/commissioning before production rollout
- What’s the best funding model to support the *new directions*?

Lessons for the future

- Funding of dedicated workforce - what‘s optimal? - a good mix of the two
    - Experiment specific (a line in the funding request an important item)
    - Via CS/IT departments (long-term strategy with IT as a service - experiment specific and general ’framework’ type support)
- Need: systemic handshake on priorities of institutional CS/IT and experiments
    - CS embedding into experiments - sync. on priorities - effectiveness
    - Long-term projection for career development - isolate career path development / uncertainty from project engagement (long vs. short term) - retention of skilled workforce
- Enable cross-talk / cross-pollination
    - Between experiments (e.g., CHEP, ACAT not sufficient - smaller targeted workshops / travel)
    - Between theory and experiments - focus on targeted collaborations / generic frameworks
    - Enable application of new methods (ML) for exp. applications (costly detector sim., inference, data recasting) - funding for CS research => practical domain applications

Thank you: Markus Diefenthaler, Jerome Lauret, Stefano Piano, Irakli Chakaberia