Introduction to High Energy Physics * Software

Pere Mato
(CERN/ PH-SFT)

DTI International Technology Service-GlobalWatch Mission
June 30th, 2004

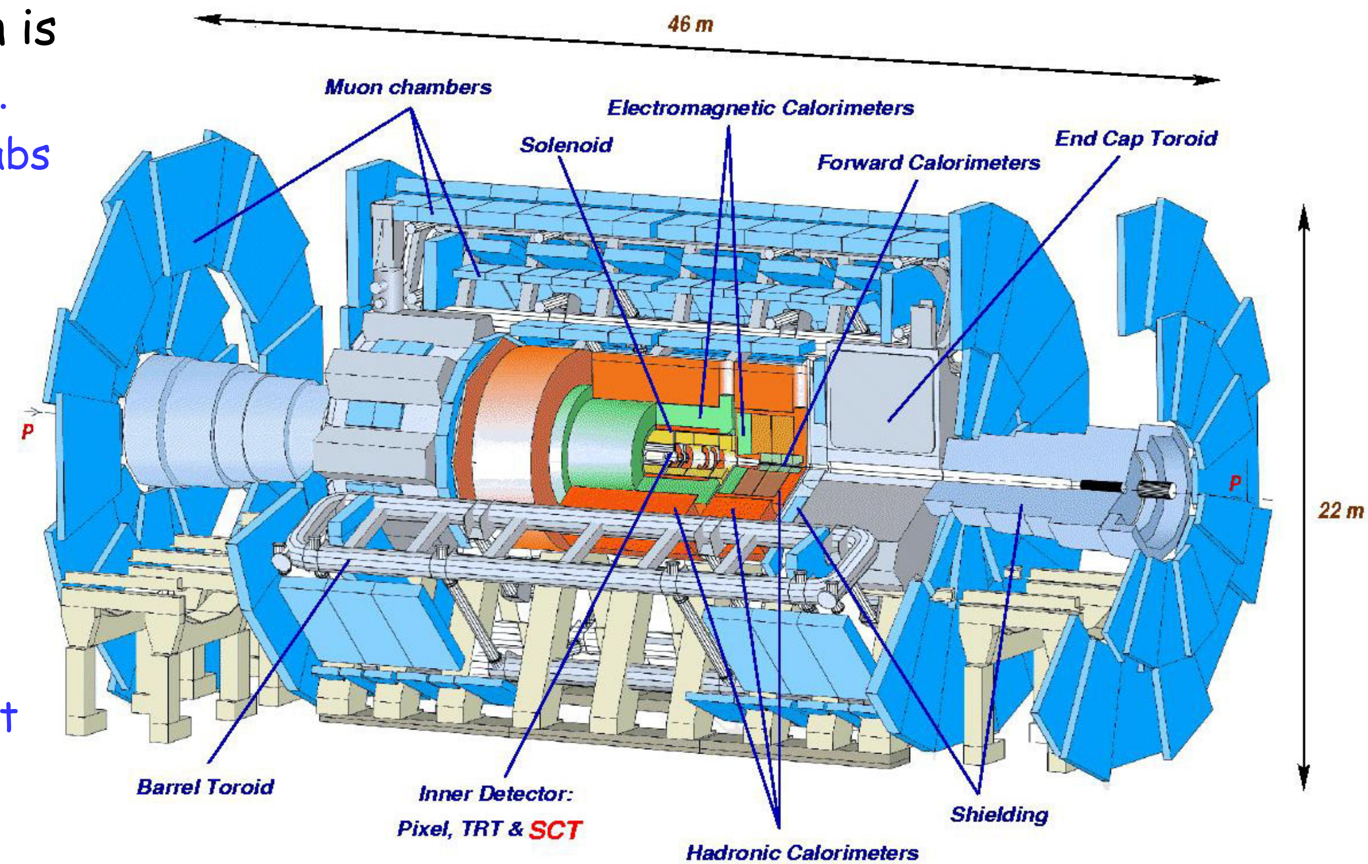
* Very much from a Large Hadron Collider (LHC) perspective

Talk Outline

- Physics signatures and rates
- Data processing and datasets
- Software structure and frameworks
- Software components and domains
- Usage of third-party software
- Summary

Example: The Atlas Detector

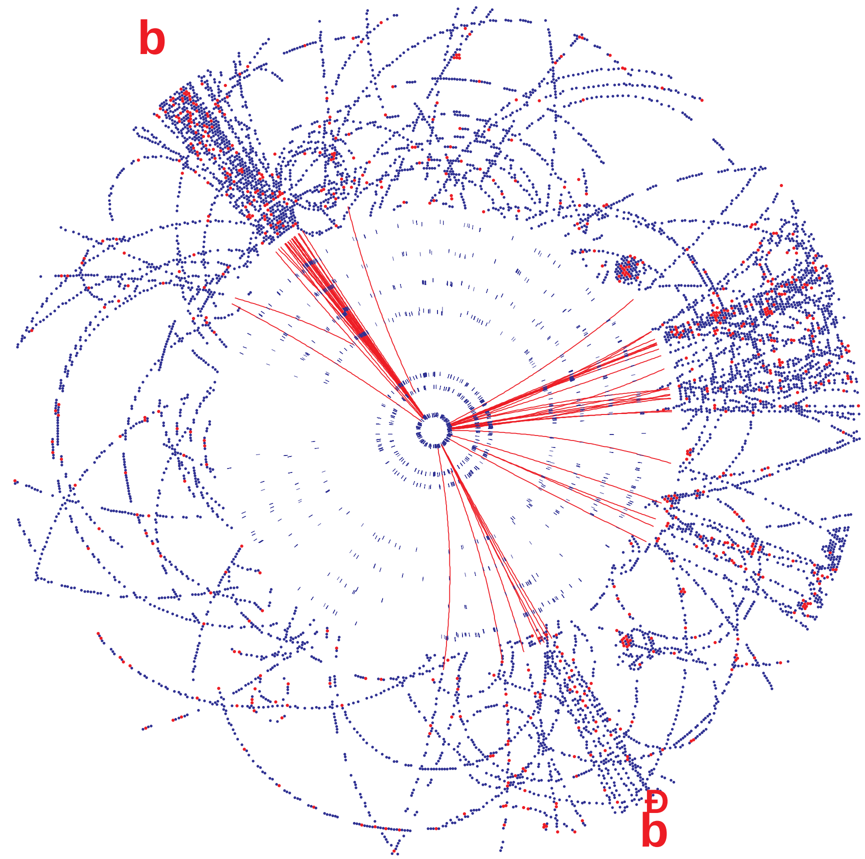
- The ATLAS collaboration is
  - ~2000 physicists from ..
  - ~ 150 universities and labs
  - from ~ 34 countries
  - distributed resources
  - remote development
- The ATLAS detector is
  - 26m long,
  - stands 20m high,
  - weighs 7000 tons
  - has 200 million read-out channels



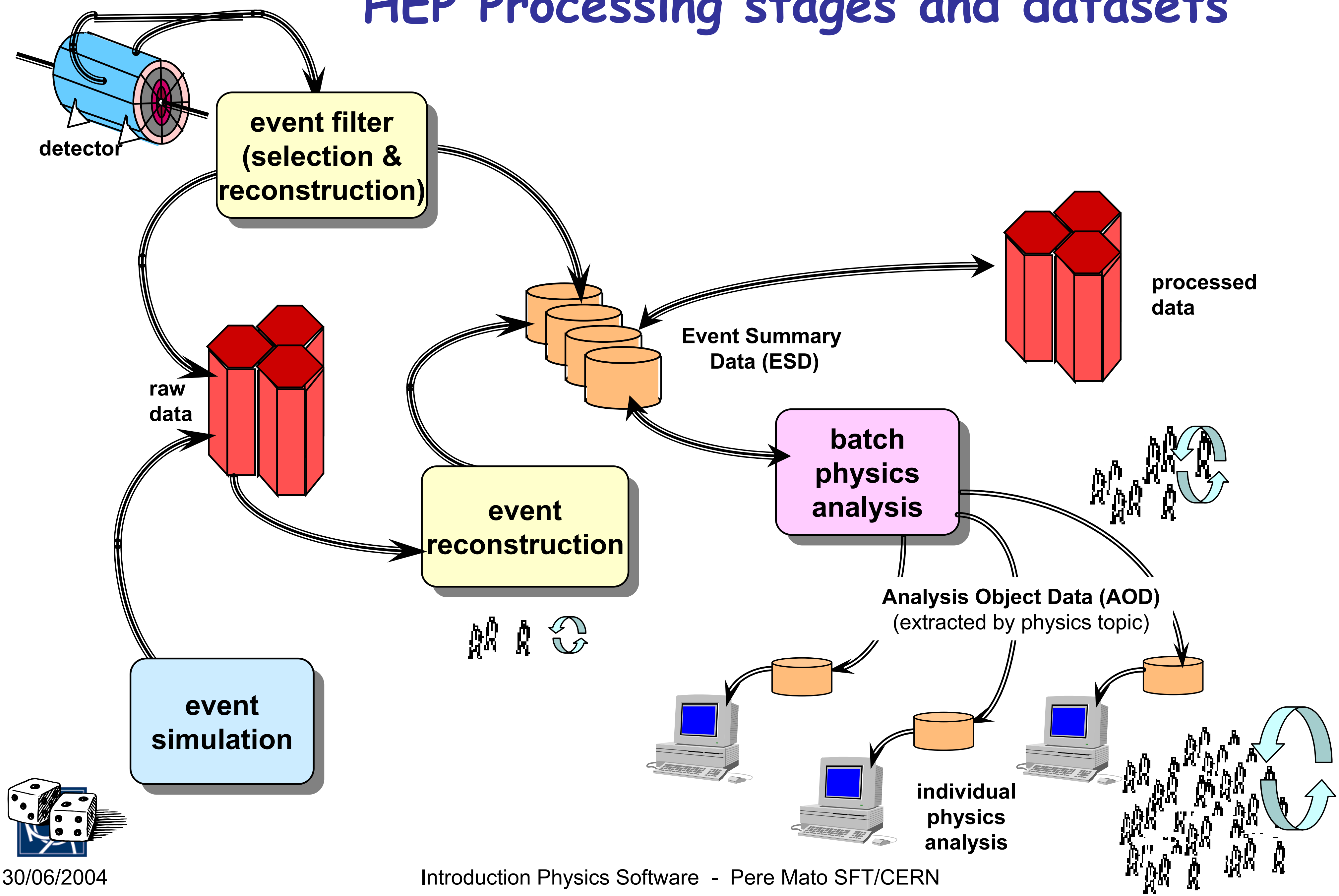
Atlas : Physics Signatures and Event Rates

- LHC: pp collisions at $\sqrt{s}$ = 14 TeV
- Bunches cross at 40 MHz
- $\sigma_{inelastic}$ = 80 mb
  - at high L >> 1 pp collision/crossing
  - 10^9 collisions per second
- Study different physics channels each with their own signature e.g.
  - Higgs
  - Supersymmetry
  - B physics
- Interesting physics events are buried in backgrounds of uninteresting physics events (~ 1 in 10^5 - 10^9 of recorded events)

ATLAS Barrel Inner Detector
$H \rightarrow b\bar{b}$



HEP Processing stages and datasets



Data and Algorithms

- HEP main data is organized in *Events* (particle collisions)
- Simulation, Reconstruction and Analysis programs process "one Event at the time"
  - Events are fairly independent to each other
  - Trivial parallel processing
- Event processing programs are composed of a number of Algorithms selecting and transforming "raw" Event data into new "processed" Event data and statistics
  - Algorithms are mainly developed by "Physicists"
  - Algorithms may require additional "detector conditions" data (e.g. calibrations, geometry, environmental parameters, etc.)
  - Statistical data (histograms, distributions, etc.) are typically the final data processing results

Data Hierarchy

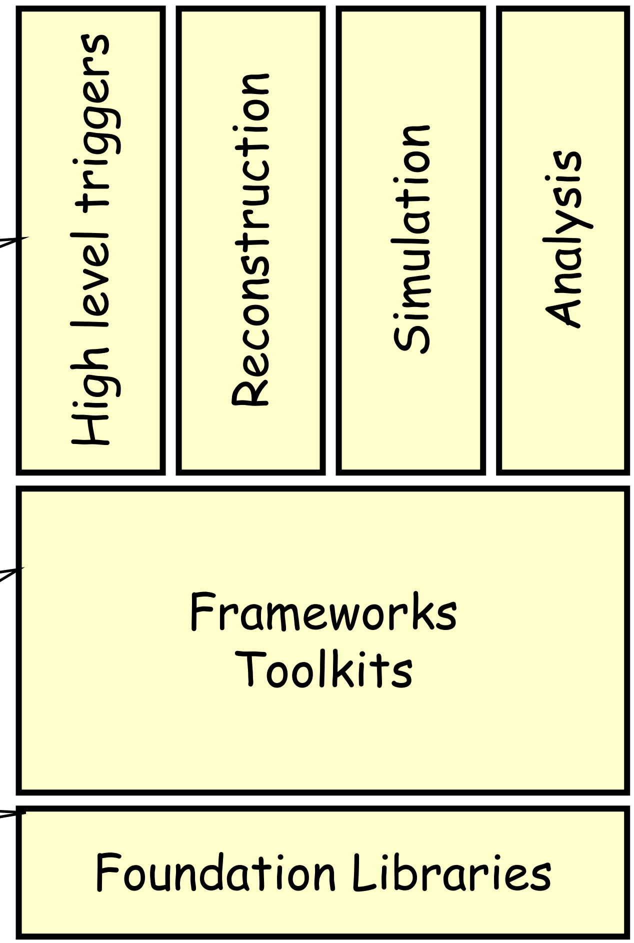
~2 MB/event	RAW	Triggered events recorded by DAQ	Detector digitisation 10^9 events/yr * 2 MB =2 PB/yr
~100 kB/event	ESD	Reconstructed information	Pattern recognition information: Clusters, track candidates
~10 kB/event	AOD	Analysis information	Physical information: Transverse momentum, Association of particles, jets, (best) id of particles,
~1 kB/event	TAG	Classification information	Relevant information for fast event selection

Software Organization

Applications built on top of frameworks and implementing the required algorithms.

One framework for basic services + various specialized frameworks: detector description, visualization, persistency, interactivity, simulation, etc.

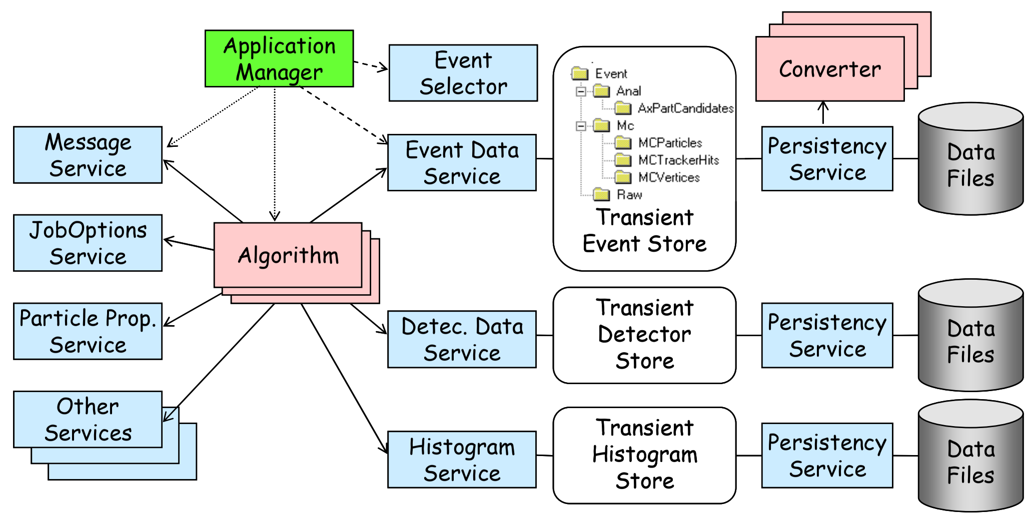
A series of basic libraries widely used: STL, CLHEP, GSL etc.



Software Frameworks

- Experiments develop Software Frameworks
  - General Architecture of the Event processing applications
  - To achieve coherency and to facilitate software re-use
  - Hide technical details to the end-user Physicists (providers of the Algorithms)
- Applications are developed by customizing the Framework
  - By the "composition" of elemental Algorithms to form complete applications
  - Using third-party components wherever possible and configuring them

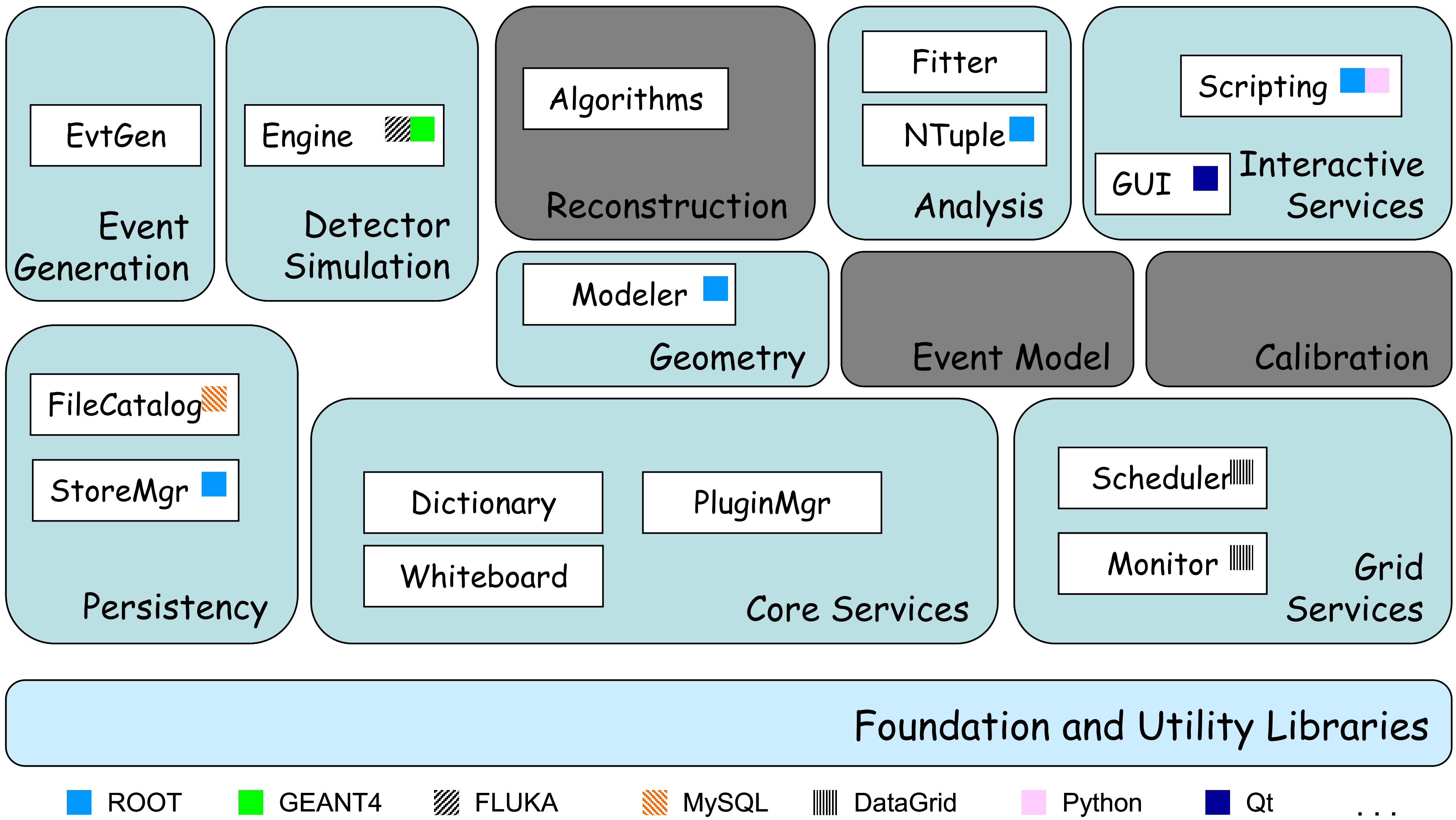
Example: Gaudi framework (C++) in use by LHCb and ATLAS



Software Components

- Foundation Libraries
  - Basic types
  - Utility libraries
  - System isolation libraries
- Mathematical Libraries
  - Special functions
  - Minimization, Random Numbers
- Data Organization
  - Event Data
  - Event Metadata (Event collections)
  - Detector Conditions Data
- Data Management Tools
  - Object Persistency
  - Data Distribution and Replication
- Simulation Toolkits
  - Event generators
  - Detector simulation
- Statistical Analysis Tools
  - Histograms, N-tuples
  - Fitting
- Interactivity and User Interfaces
  - GUI
  - Scripting
  - Interactive analysis
- Data Visualization and Graphics
  - Event and Geometry displays
- Distributed Applications
  - Parallel processing
  - Grid computing

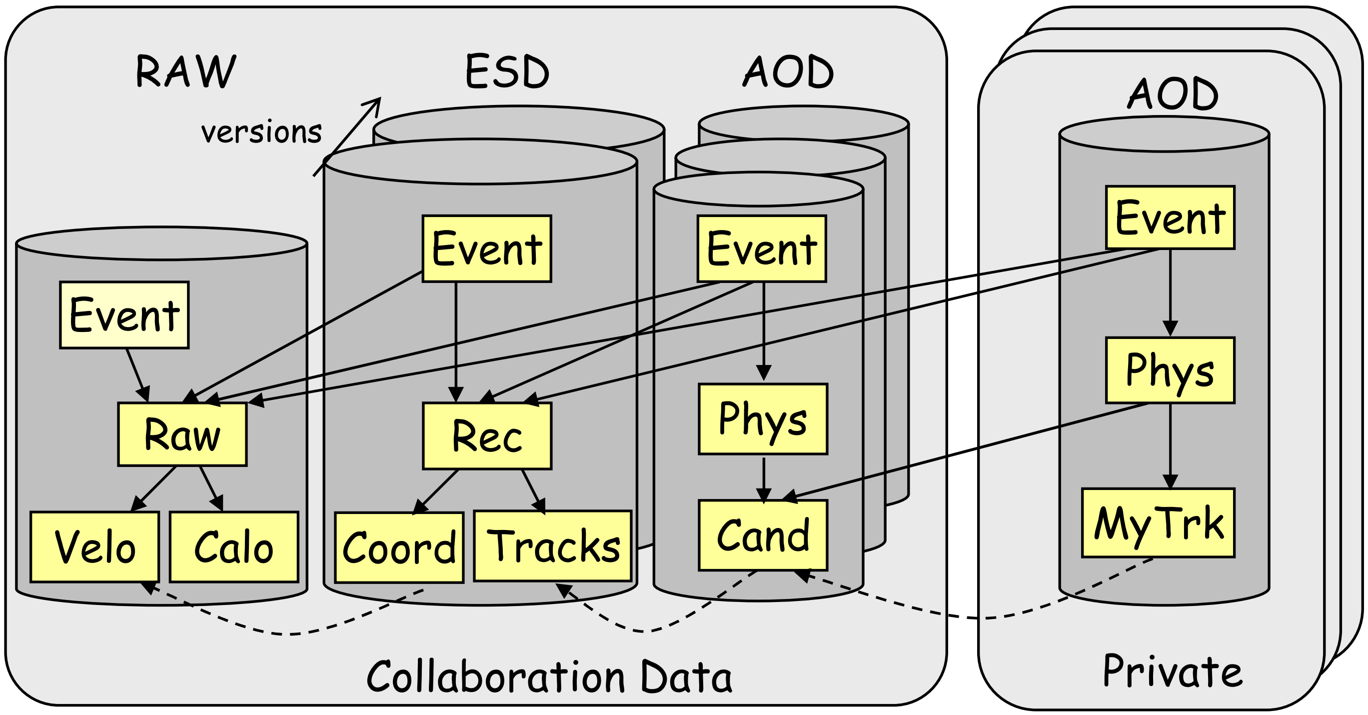
Components and Domains



Core Libraries and Services (SEAL)

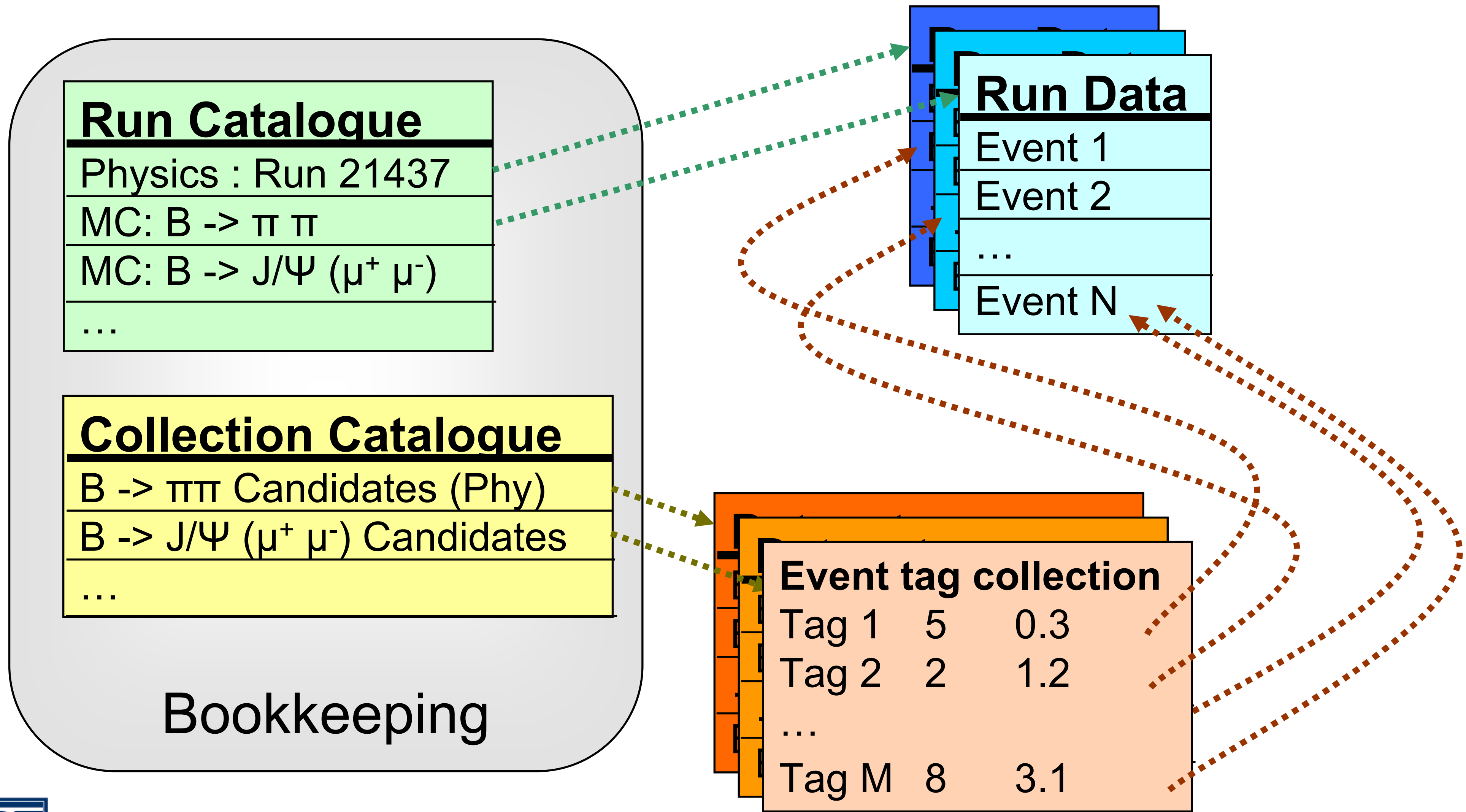
- Goals
  - Provide the software infrastructure, basic frameworks, libraries and tools that are common among the LHC experiments
  - Select, integrate, develop and support foundation and utility class libraries
  - Develop a coherent set of basic framework services to facilitate the integration of LCG and non - LCG software
- Scope
  - Foundation Class Libraries
    - Basic types (STL, Boost, CLHEP, …), utility libraries, system isolation libraries, domain specific foundation libraries
  - Basic Framework Services
    - Component model, reflection, plugin management, incident (event) management, distributed computing, grid, scripting

Event Data



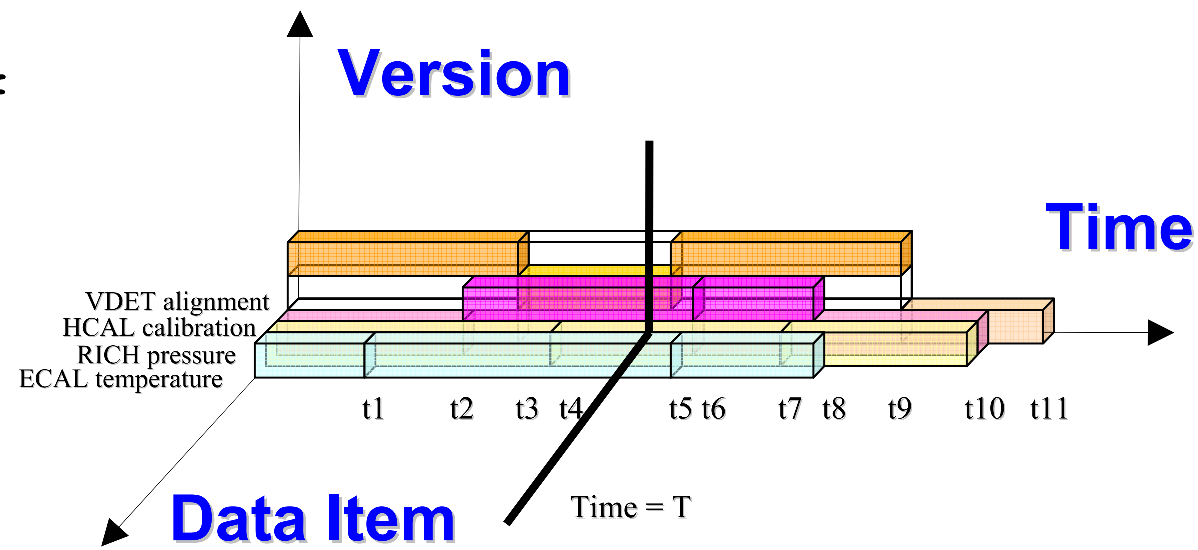
- Complex data models
  - ~500 structure types
- References to describe relationships between event objects
  - unidirectional
- Need to support transparent navigation
- Need ultimate resolution on selected events
  - need to run specialised algorithms
  - work interactively
- Not affordable if uncontrolled

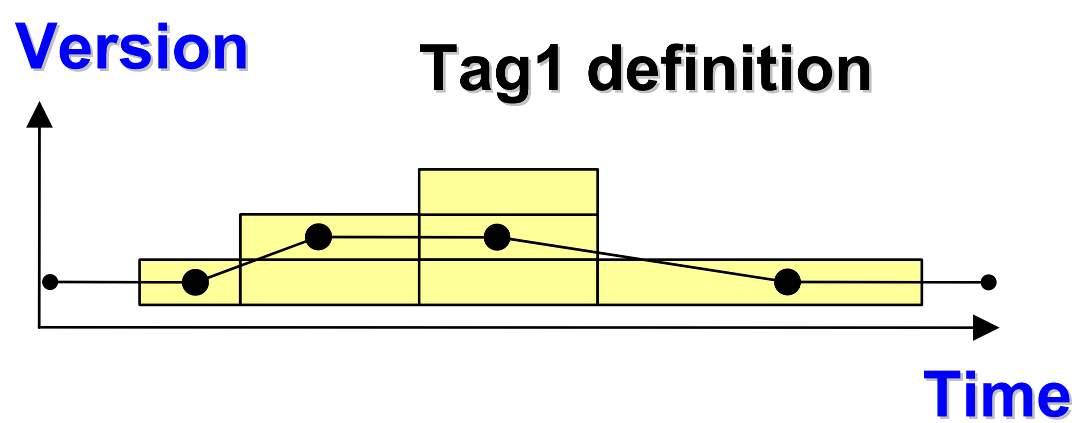
HEP Metadata - Event Collections



Detector Conditions Data

- Reflects changes in state of the detector with time
- Event Data cannot be reconstructed or analyzed without it
- Versioning
- Tagging
- Ability to extract slices of data required to run with job
- Long life-time





LHC Data Management Requirements

- Increasing focus on maintainability and change management for core software due to long LHC lifetime
  - anticipate changes in technology
  - adapt quickly to changes in environment & physics focus
- Common solutions will simplify considerably the deployment and operation of data management in centres distributed worldwide
  - Common persistency framework (POOL project)
- Strong involvement of experiments from the beginning required to provide requirements
  - some experimentalists participate directly in POOL
  - some work with software providers on integration in experiment frameworks

Common Persistency Framework (POOL)

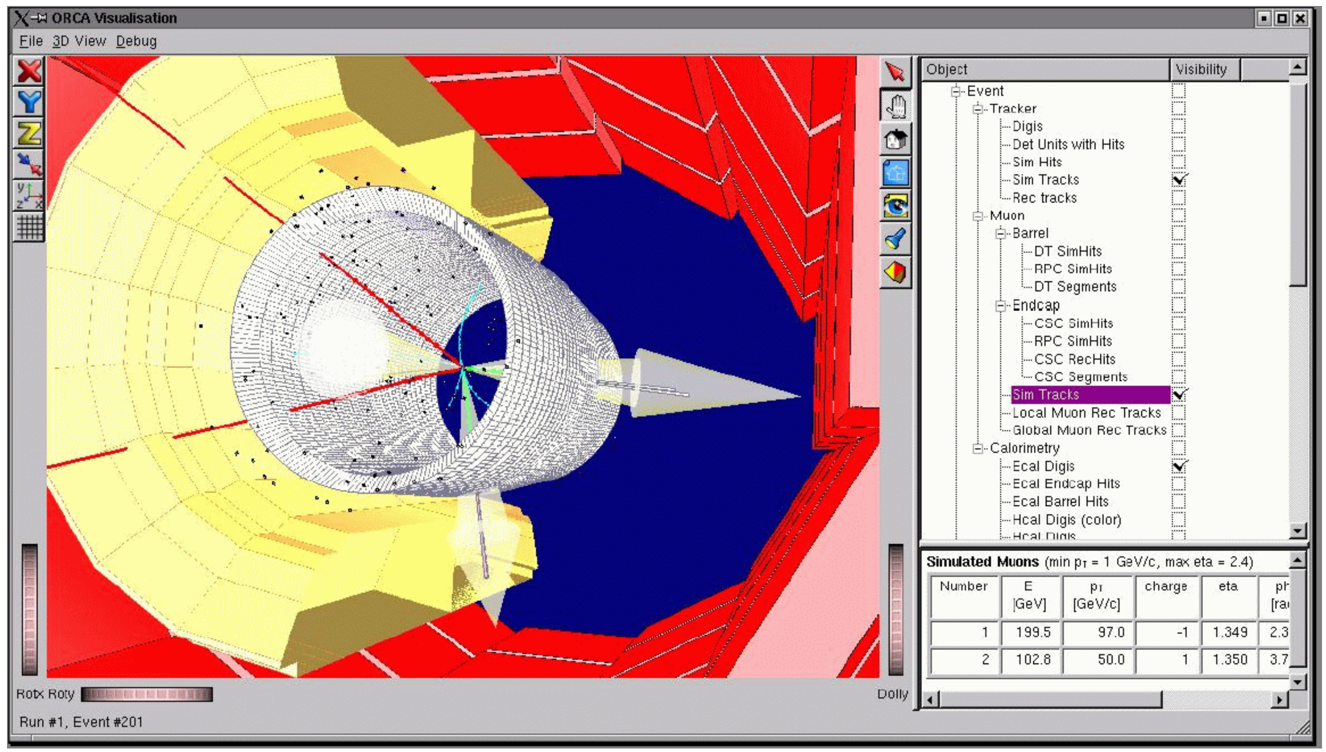
- Provides persistency for C++ transient objects
- Supports transparent navigation between objects across file and technology boundaries
  - without requiring user to explicitly open files or database connections
- Follows a technology neutral approach
  - Abstract component C++ interfaces
  - Insulates experiment software from concrete implementations and technologies
- Hybrid technology approach combining
  - Streaming technology for complex C++ objects (event data)
    - event data - typically write once, read many (concurrent access simple)
  - Transaction-safe Relational Database (RDBMS) services,
    - for catalogs, collections and other metadata
- Allows data to be stored in a distributed and grid-enabled fashion
  - Integrated with an external File Catalog to keep track of the file physical location, allowing files to be moved or replicated

Simulation

- Event Generators
  - Programs to generate high-energy physics events following the theory and models for a number of physics aspects
- Specialized Particle Decay Packages
  - Simulation of particle decays using latest experimental data
- Detector Simulation
  - Simulation of the passage of particles through matter and electromagnetic fields
  - Detailed geometry and material descriptions
  - Extensive list of physics processes based on theory, data or parameterization
- Detector responses
  - Simulation of the detecting devices and corresponding electronics

Data Visualization

- Experiments develop interactive Event and Geometry display programs
  - Help to develop detector geometries
  - Help to develop pattern recognition algorithms
  - Interactive data analysis and data presentation
- Ingredients
  - GUI
  - 3-D graphics
  - Scripting



Example: ORCA visualization in CMS (IGUANA)

Distributed Analysis

- Analysis will be performed with a mix of "official" experiment software and private user code
  - How can we make sure that the user code can execute and provide a correct result wherever it "lands"?
- Input datasets not necessarily known a-priori
- Possibly very sparse data access pattern when only a very few events match the query
- Large number of people submitting jobs concurrently and in an uncoordinated fashion resulting into a chaotic workload
- Wide range of user expertise
- Need for interactivity - requirements on system response time rather than throughput
- Ability to "suspend" an interactive session and resume it later, in a different location

Used External Products

- Foundation Libraries
  - STL, Boost, CLHEP, Zlib,…
- Mathematical Libraries
  - NagC, GSL, CLHEP,…
- Data Organization
  - Oracle, MySQL, XercesC,…
- Data Management Tools
  - ROOT, Oracle, MySQL, EDG,…
- Simulation Toolkits
  - Pythia, Herwig, Geant4, Fluka,…
- Statistical Analysis Tools
  - ROOT, GSL,…
- Interactivity and User Interfaces
  - Qt, Python, ROOT,…
- Data Visualization and Graphics
  - Coin, OpenGL,…
- Distributed Applications
  - PROOF, Globus, EDG,…

Summary

- HEP applications are characterized by :
  - Amounts and complexity of the data
  - Large size and geographically dispersed nature of the collaborations
  - Most of the algorithmic software written by Physicists
  - Expected long lifetimes
- Development of Software Frameworks
  - Ensure coherency in the Event data processing applications
  - Make the life of Physicists easier by hiding most of the technicalities
  - Withstand technology changes
- A variety of different software domains and expertise required
  - Data Management, Simulation, Interactive Visualization, Distributed Computing, etc.
- Extensive use of third-party generic software
  - Open-source products favored

The LHC Detectors