



# Bringing physics data analytics software infrastructure to exascale

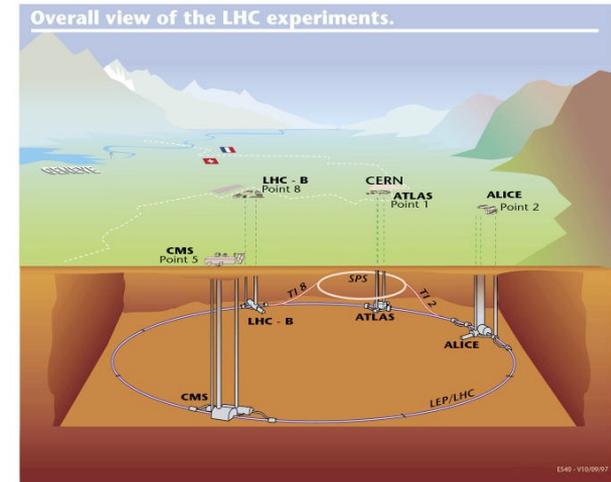
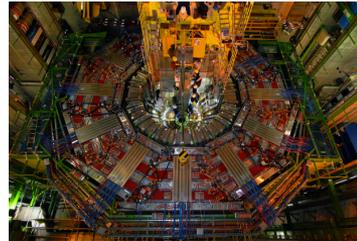
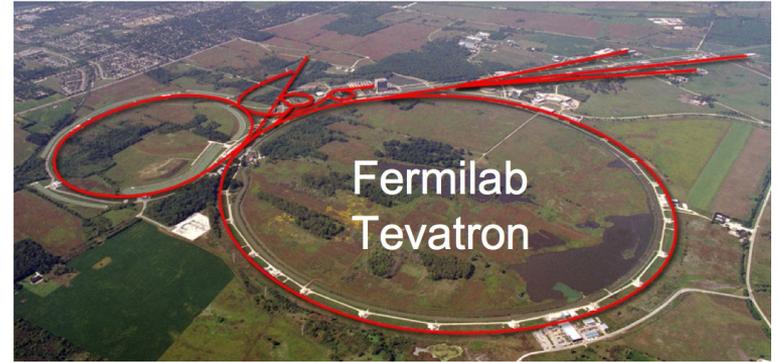
Jim Kowalkowski  
SC16 DOE Booth  
17 Nov 2016

# Abstract

High Energy Physics has had great success in defining, designing, implementing, and deploying very large-scale scientific software frameworks. These frameworks have enabled thousands of scientists to collaborate, successfully developing and constructing applications and workflows to accomplish complex analysis campaigns that operate on petabyte-scale datasets in the context of Distributed Area High Throughput Computing. This talk explores the possibilities and discusses the opportunities on how these principals, practices, and concepts can be brought to bear in the era of exascale to help effectively utilize advanced computing features while increasing participation and productivity.

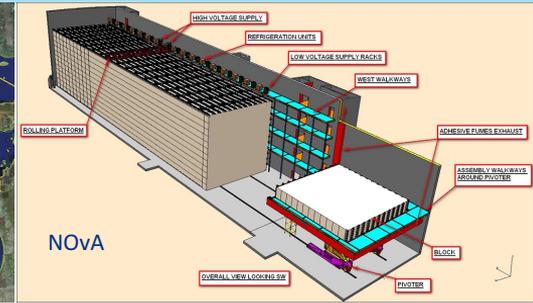
# Energy Frontier

- **Until September 2011: Tevatron at Fermilab**
  - Circumference: 3.75 miles
  - Collision energy:  $\sim 2$  TeV
  - The mass of a proton is about 1 GeV
- **Now: Large Hadron Collider (LHC) in Geneva, Switzerland**
  - Circumference 17 miles
  - Collision energy  $\sim 13$  TeV
  - Particle beam: 100 Billion protons per bunch
- **Fermilab is involved in the CMS experiment**
  - 100 million channels
  - Collisions every 25ns



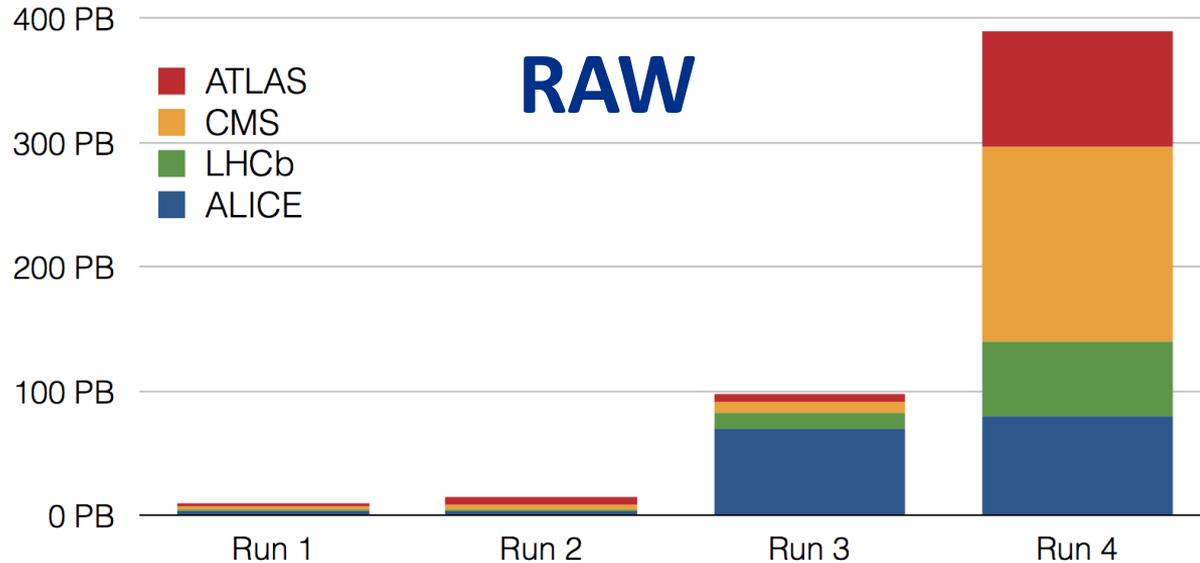
# Neutrino Experiments

- Detect neutrinos and measure their masses
- Important Standard Model measurement, candidates for dark matter
- **CP violation**
- Produce neutrino beam and direct it to far detector, compare near and far detector measurements
- **NOvA**
  - 500 miles to Ash River, Minnesota
  - Measuring neutrino oscillations
- **MicroBooNE**
  - Liquid Argon technology
  - Neutrino interaction rate and mystery of excess events seen by MiniBooNE
- **DUNE/LBNF (future)**
  - Far detector 4800 feet underground
  - 68K tons of liquid argon
  - 3D imaging



# Data usage from HEP now and towards HL-LHC / DUNE era

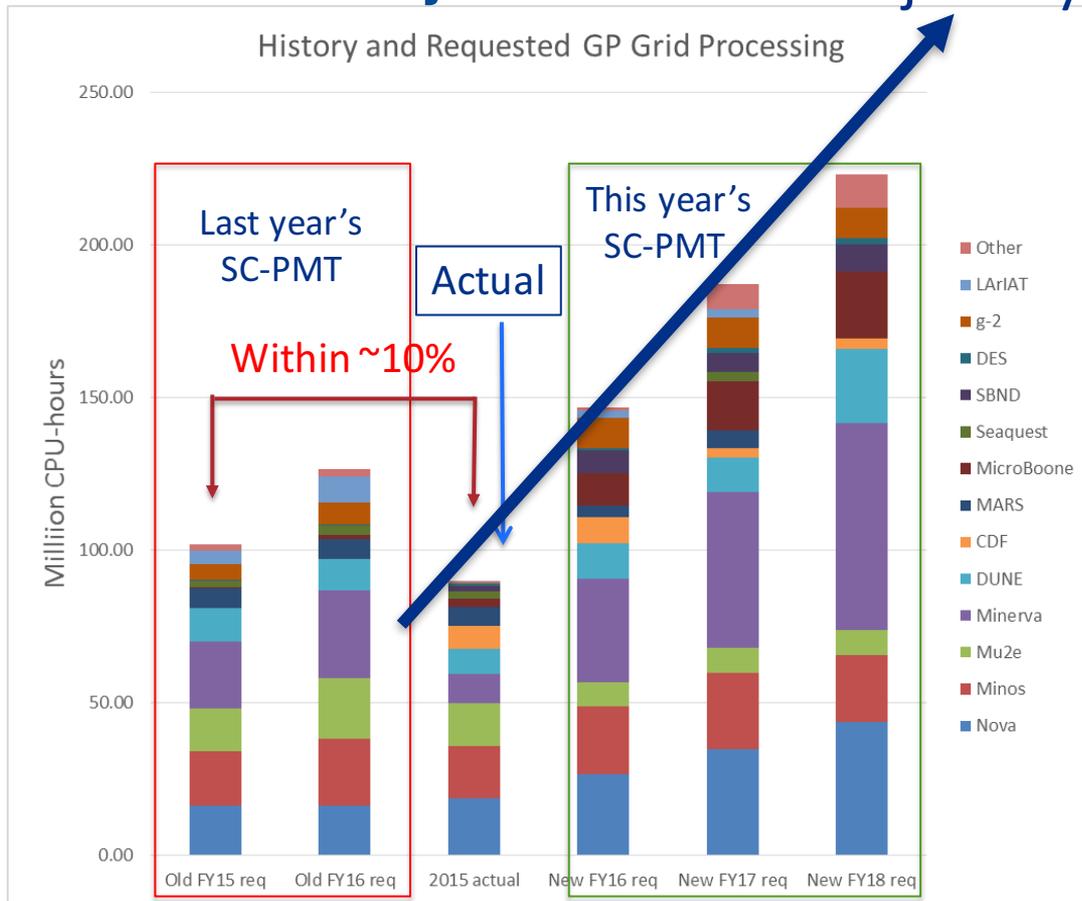
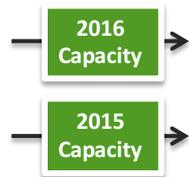
## LHC expected data volumes



- Shown: RAW data expectations
  - Derived data is 8x RAW ! (Reconstruction, Simulation)
- LHC Run 4 starts within the exabyte era
- How do we analyze that much data in the future?

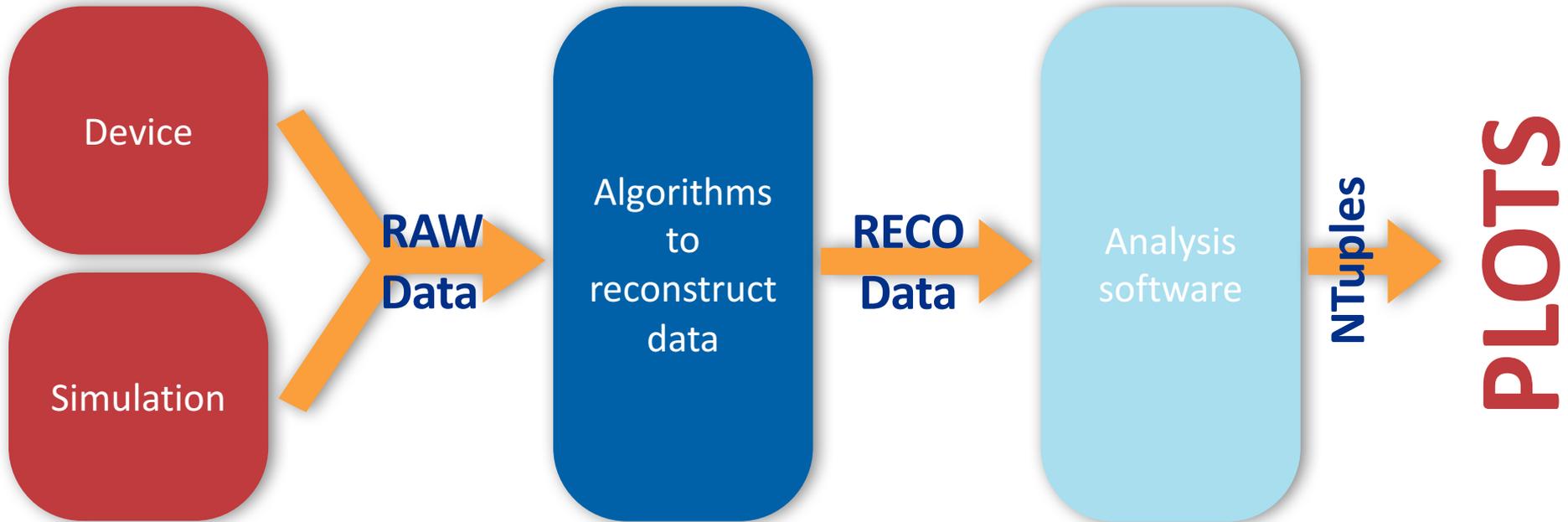
# Intensity Frontier compute needs steadily climbing

- We can predict computing needs
- Steadily climbing as we approach the DUNE era



Trajectory

# Our current practices ...



← This is bigger than it looks

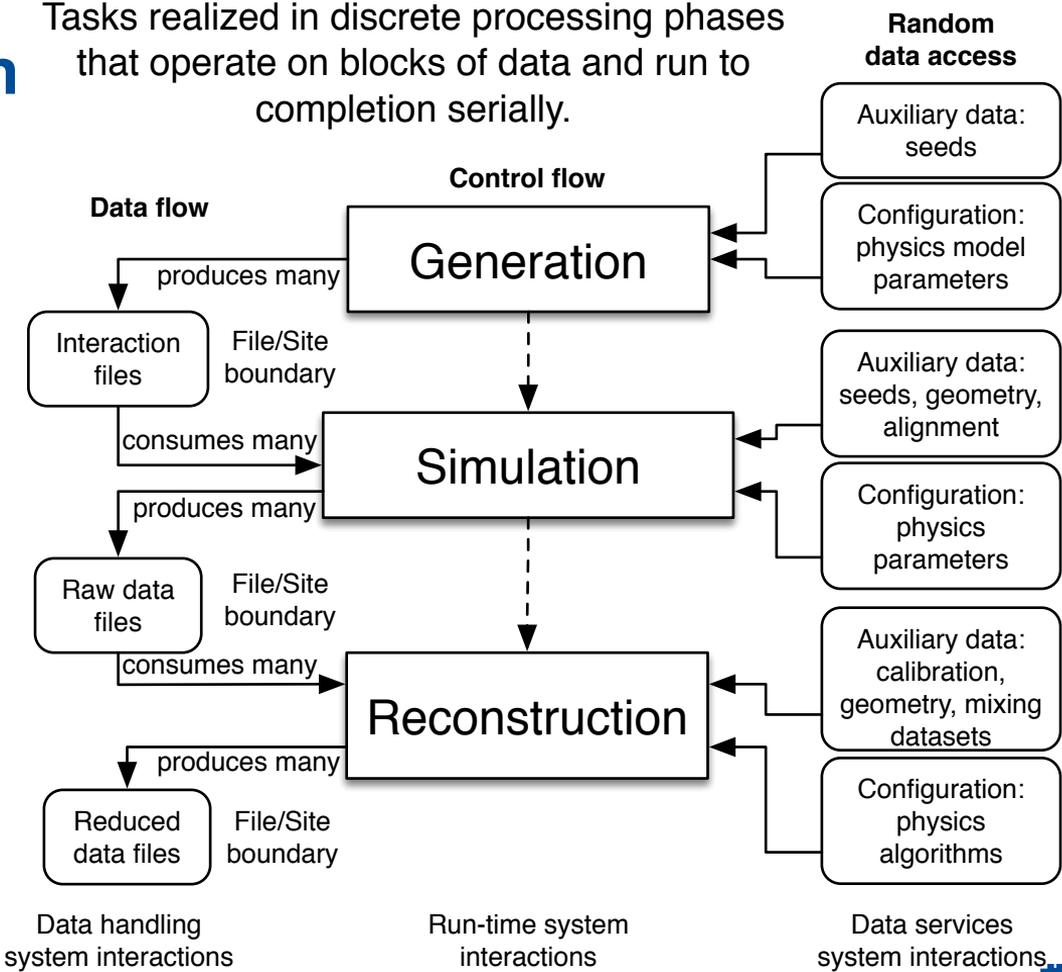
- **No surprise: Software** is important for every step on the way to scientific results

# Abstract production workflow

- Organized to carry out large campaigns
- Output (files) used in downstream analyzes
- Ancillary data handling and use is non-trivial

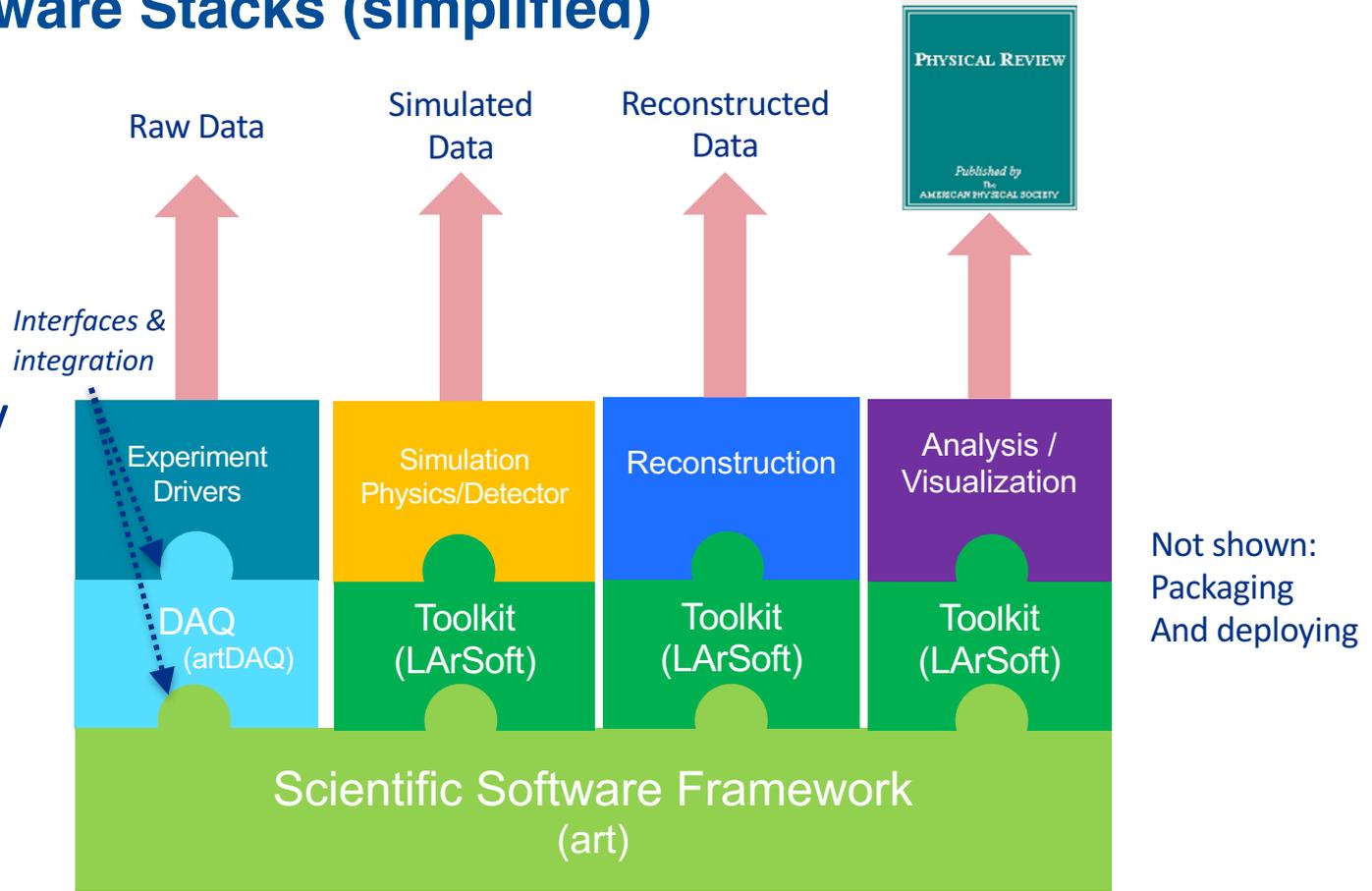
USCMS performed a demo here at SC with Google, running >150K jobs carrying out this workflow, operating with .5 PB of data.

Tasks realized in discrete processing phases that operate on blocks of data and run to completion serially.



# Scientific Software Stacks (simplified)

- Lots of shared infrastructure software
- Used in nearly all stages of production workflows

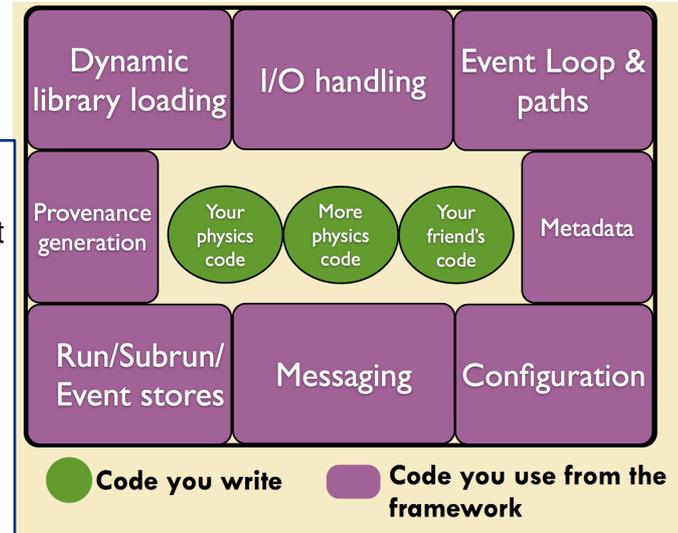


# Detailed information on *art*, the event processing framework

- An integrated and coherent framework for simulation, reconstruction, analysis and data taking
- Began ~2010 as an evolution of the CMS framework adapted for Intensity Frontier experiments,
- Delivered as an external product

## What does the framework do for you?

- Mostly the framework exists to handle the tasks that you don't care much about, but which have to work.
  - reading input
  - writing output
  - loading libraries containing algorithms you want to run
  - configuring those algorithms
  - keeping track of how output were generated ("provenance tracking"), critical for reproducibility
  - organizing histogram output
  - access to "global resources": geometry information, calibrations, *etc.*
  - systematizing the handling of errors
  - timing modules, measuring memory use, tracking program execution

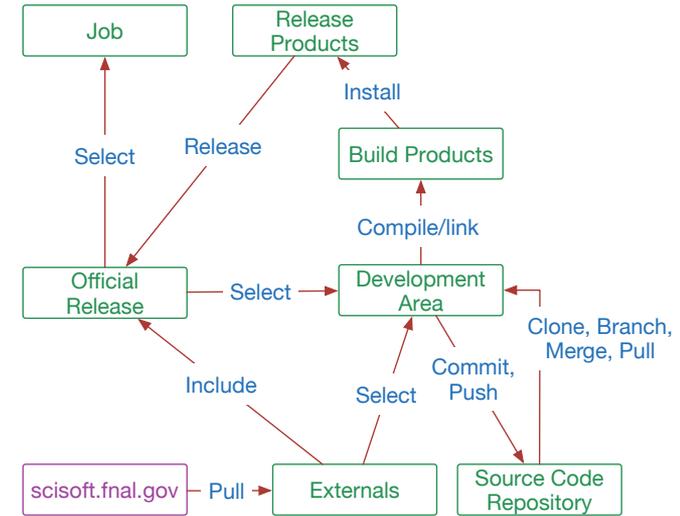


<http://inspirehep.net/record/1229212>

<http://iopscience.iop.org/article/10.1088/1742-6596/396/2/022020/pdf>

# What are the parts of the “ecosystem”?

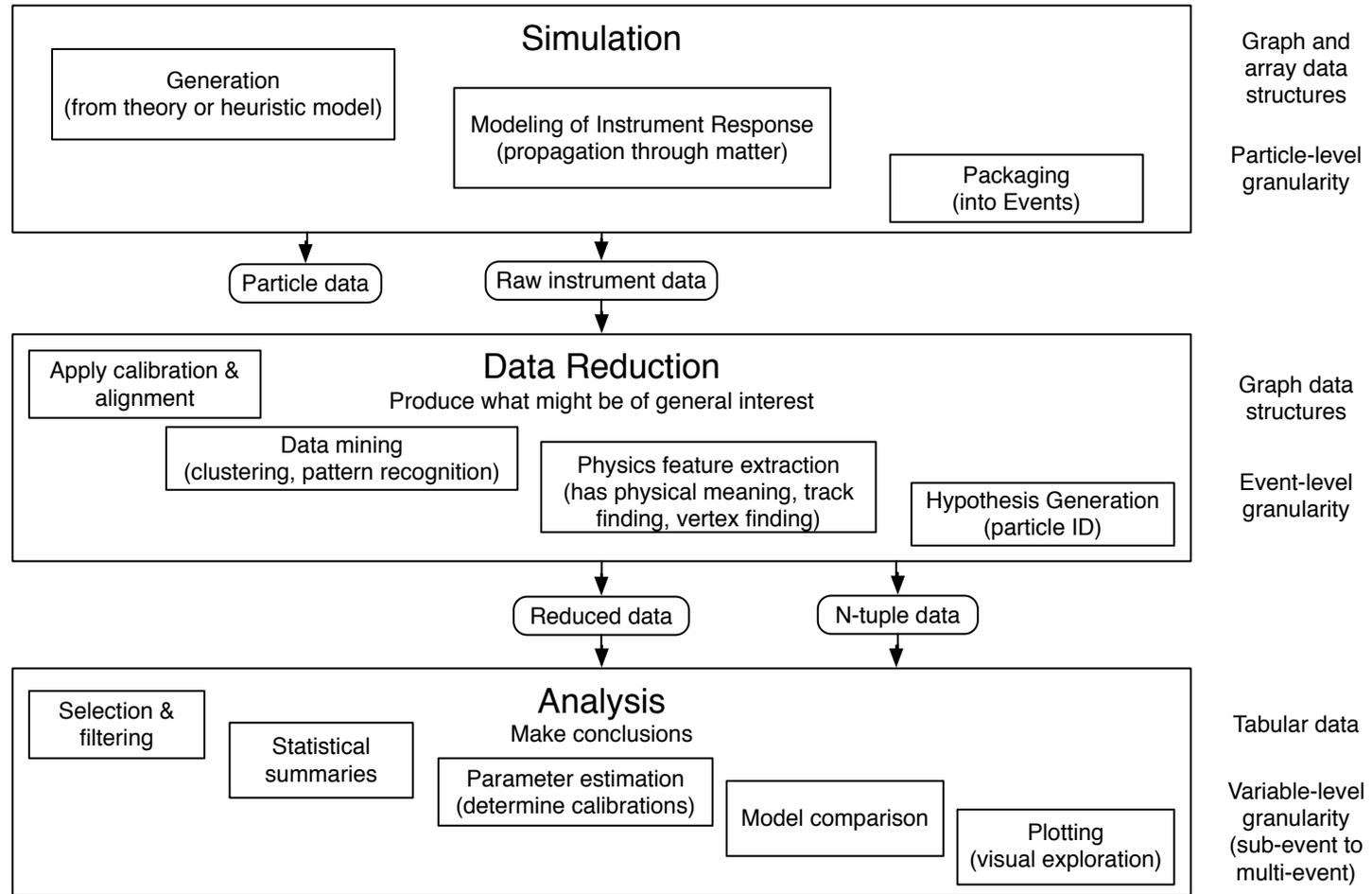
- A build system
  - Art has a specific set of tools based on cmake
  - Experiments using art can use whatever they want.
- Release, dependency and environment control
  - Working environment for any complete version of all software packages can be established
- Source code management
  - A defined process for support and maintenance
- A source (and binary) distribution mechanism



# Can this be applied to Exascale problems?

# Types of Processing Tasks

- Kinds of tasks within a full data processing campaign



# Observations

- A successful scientific application requires that the bulk of the software contributions for data analytics and data reduction come directly from domain scientists
- Emerging computing architectures and hardware can make this difficult.
  - Most domain scientists do not have the required expertise
- Ways that scientific output might be increased given this situation
  - reduce the time spent writing code,
  - reduce the time spent debugging and testing code,
  - greatly increase the number of scientists that contribute, and
  - allow all that code and the results that are produced to be **shared within a community or even across communities.**

# Need for software algorithmic frameworks

- Software infrastructure must exist within a framework, which
  - Handles *coordination of algorithms* and *exchange (and persistency) of data*
  - Allows design principles and features to be leveraged across science programs and experiments, minimizing duplication
  - Provides the concepts, principles, interfaces, process, and constraints necessary for sharing and reuse
- A framework lives within a broader ecosystem, providing
  - Management of development, testing, deployment, and run-time environment
  - Connections necessary for conducting science: *workload* and *workflow* management, *data management*, and *data archival facilities*

## A proven path for HEP experiments

- Principle components realized and refined through
  - two generations of HEP collider-detector experiments
  - two generations of HEP neutrino experiments,
  - deployments for cosmology (CosmoSIS),
  - explorations such as the LSST Dark Energy Science Collaboration (DESC) L3 analysis demo
- Software and principles shared across tens of experiments and across three different science domains within HEP

# Framework instantiations

- The LHC CMS experiment's instance of this framework is shared among thousands of collaborators within a single experiment. A four-fold increase in the number of software contributors since inception
  - CDF experiment with **400 unique contributors** from 1997-2007,
  - CMS experiment with nearly **1800 contributors** over 2006-2016
- The art instance is shared among independent experiments in multiple scientific domains.
  - grown from just one experiment to nine experiments in three different program domains
- Collaborative features of art has enabled higher-level projects and community-supported algorithm repositories
  - **LArSoft** demonstrates sharing of module components (algorithms and simulation integration) across experiments with similar properties
  - **Artdaq** provides data acquisition and real-time filter capabilities

# Moving towards exascale

# Can we take a bigger leap than just porting?

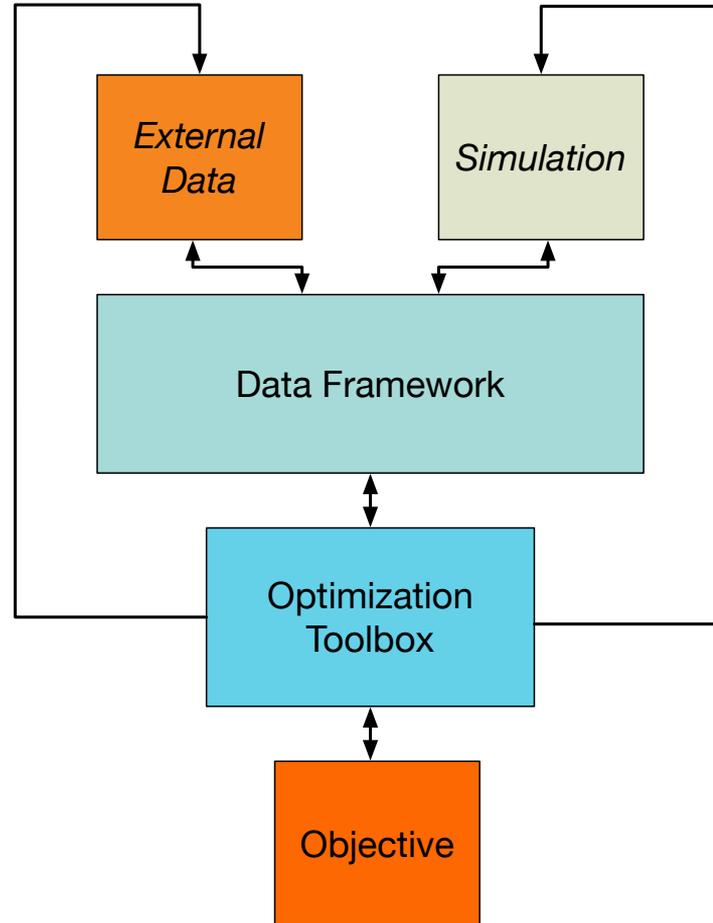
- **Integrate Data Science with Simulation**
- Goals
  - produce a complete software framework
  - enable synergistic simulation and analysis of both simulated and instrument data
  - include both streaming real-time and near-real-time operations
- Features
  - take advantage of exascale hardware I/O hierarchies
  - Allow for efficient cooperative data analysis and simulation
  - coupling with an optimization toolkit
  - *Provide a controls component*
- Not only take our infrastructure to HPC, but do something more

# Integrated Data Science and Simulation

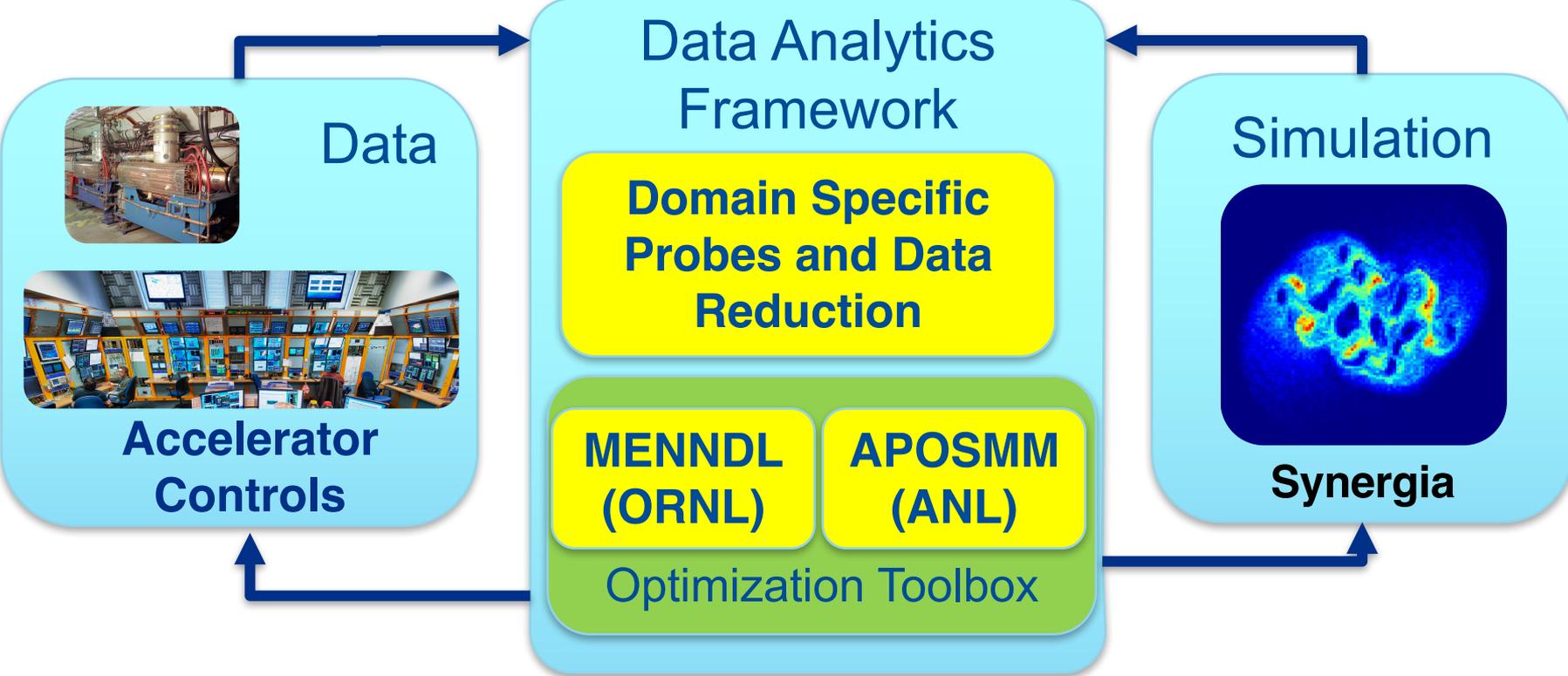
- We have surveyed proposed exascale applications, asking about
  - controls,
  - steered data analysis,
  - and simulation steering
- Found a strong overlap with projects involved with
  - earthquake prediction,
  - advanced accelerator modeling,
  - soft materials,
  - biological neutron science,
  - and manufacturing
- Applications cover an extremely broad scope
  - indicates that tools combining simulation and data analysis with a controls aspect can have impact across the exascale computing landscape.

# A simple view

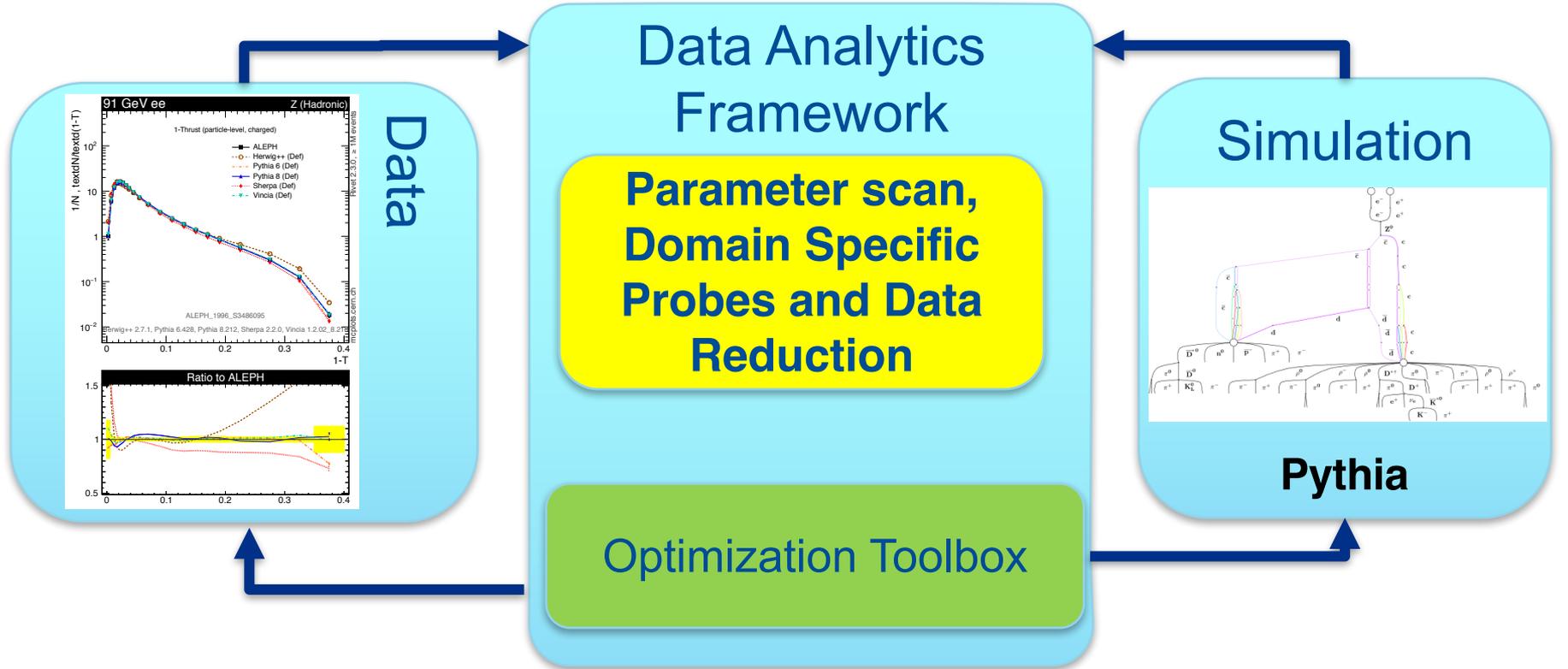
- What might such a system look like?



# For accelerator control optimization

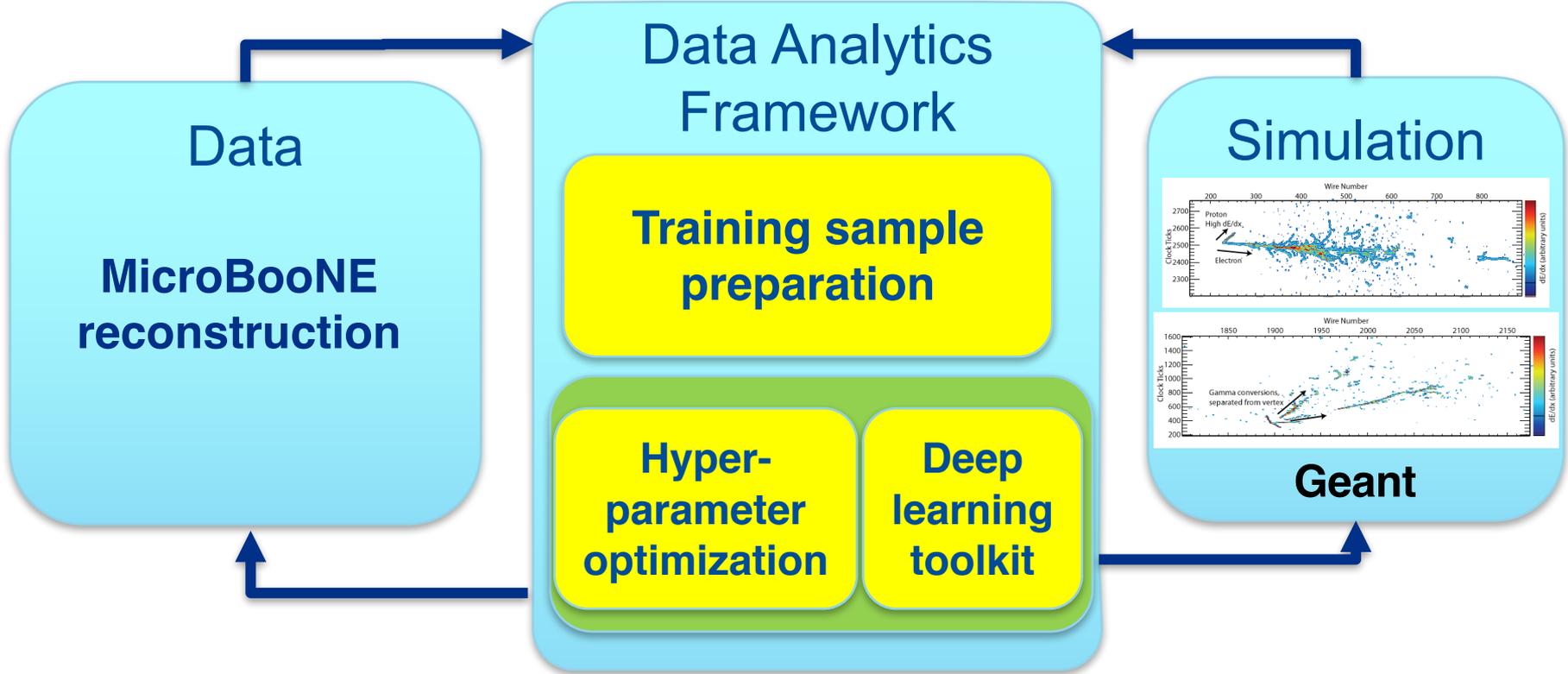


# For experimental collider-detector physics

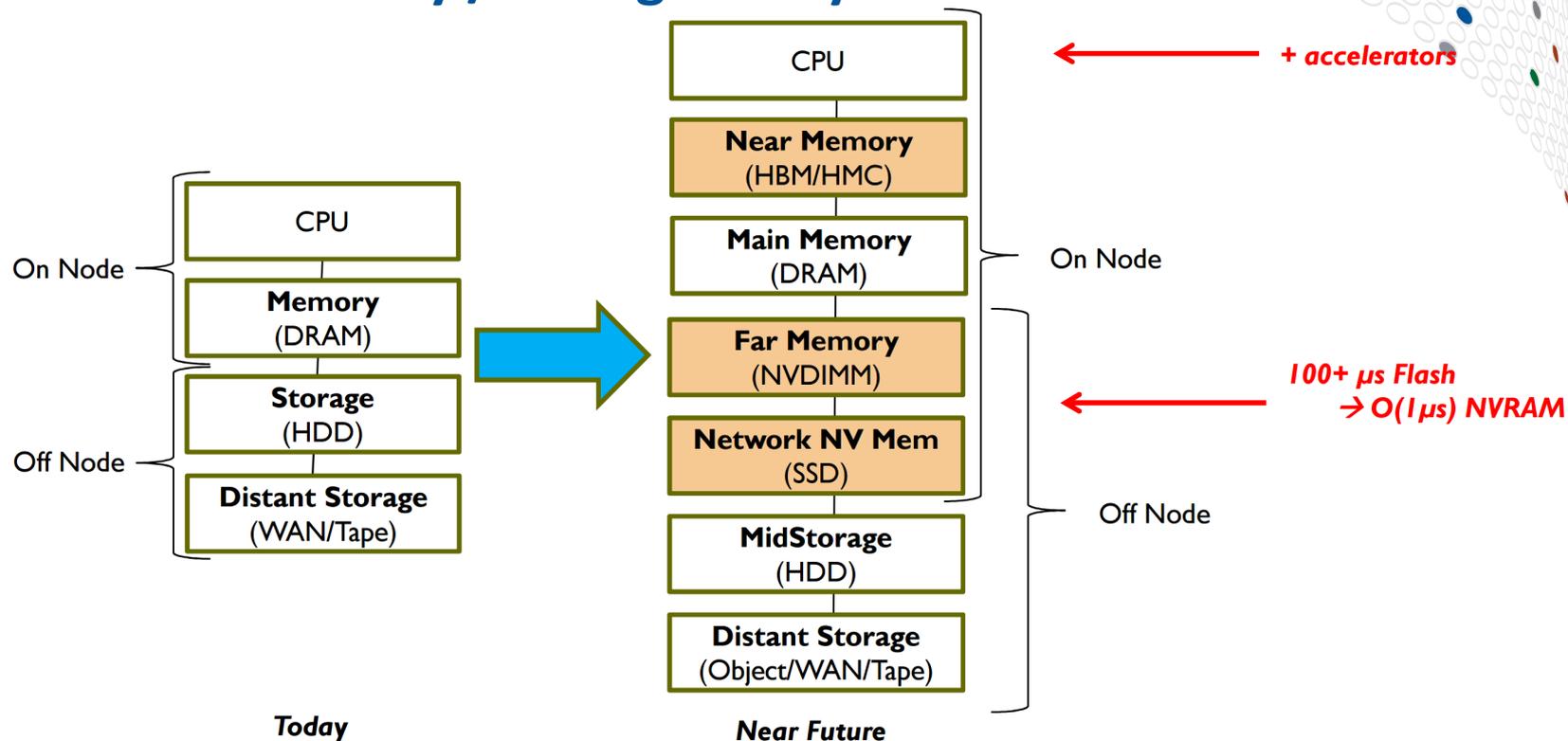


Quality of match?

# For neutrino experimental physics



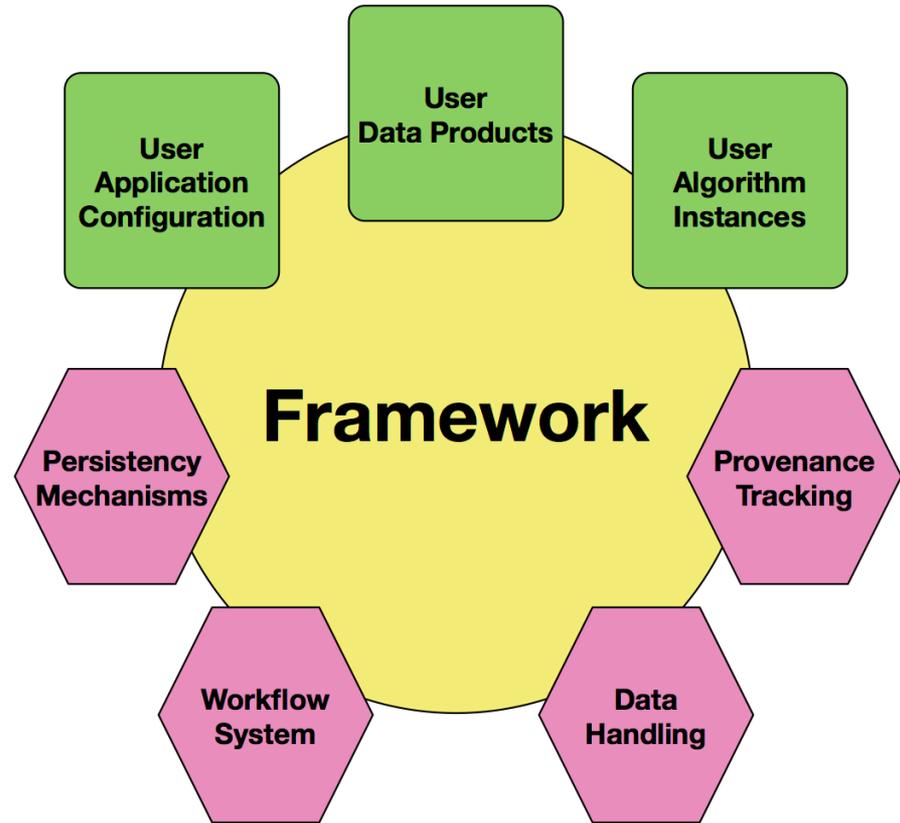
# Trends in the Memory / Storage Subsystem



<https://hpcuserforum.com/presentations/paris-munich/CRAY.HPCUserForumOctoberParis%202015CRAY.pdf>

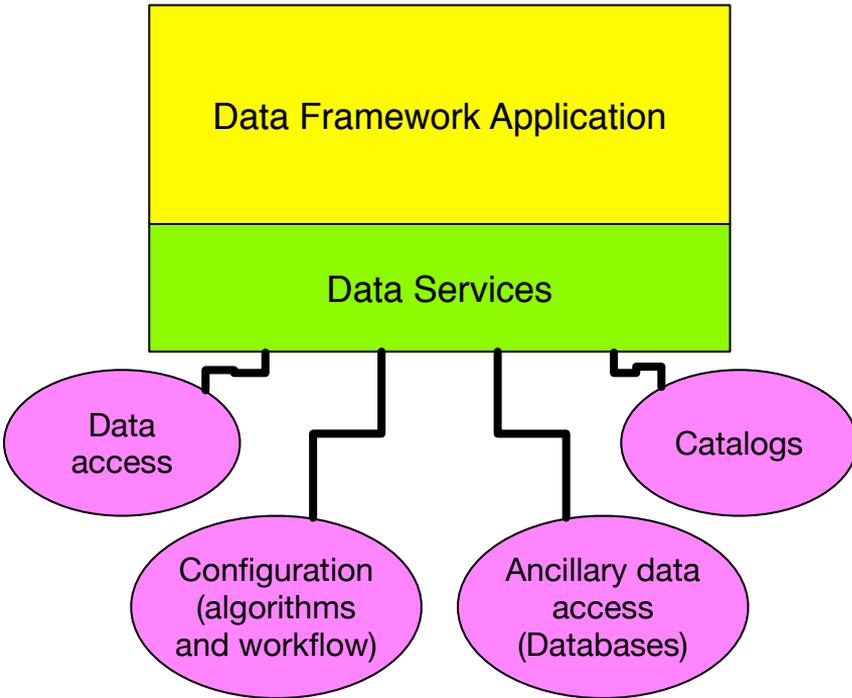
# Data framework overview

- These concepts, abstractions, protocols, and component relationships define this software framework
- Very successful within a High Throughput Computing context for data-intensive science
- Hides complex details within the hexagon components from the scientist-contributor
- Run-time configuration components provide domain-specific parameterization needed to establish workflows from validated “plugin” algorithm libraries.
- The framework links to hardware-specific libraries for optimizing running.

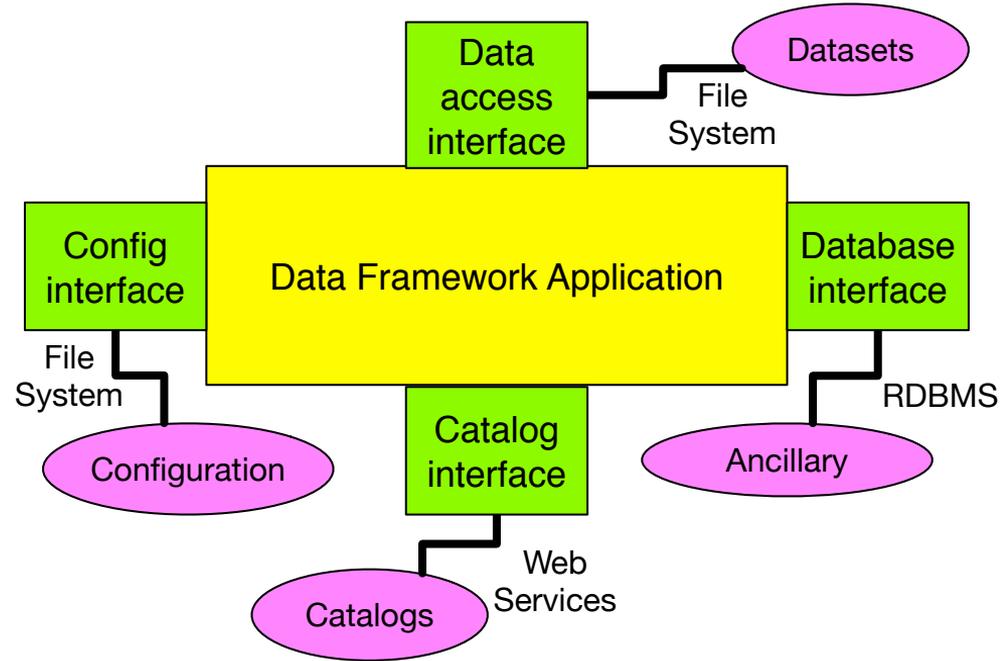


# Framework interactions with storage and memory

## Unified View



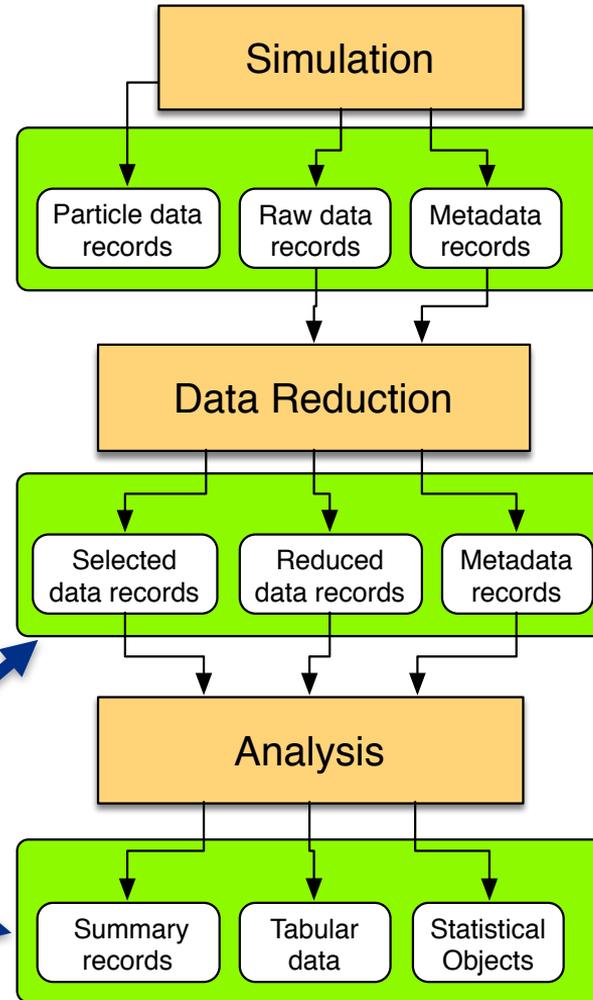
## Traditional View



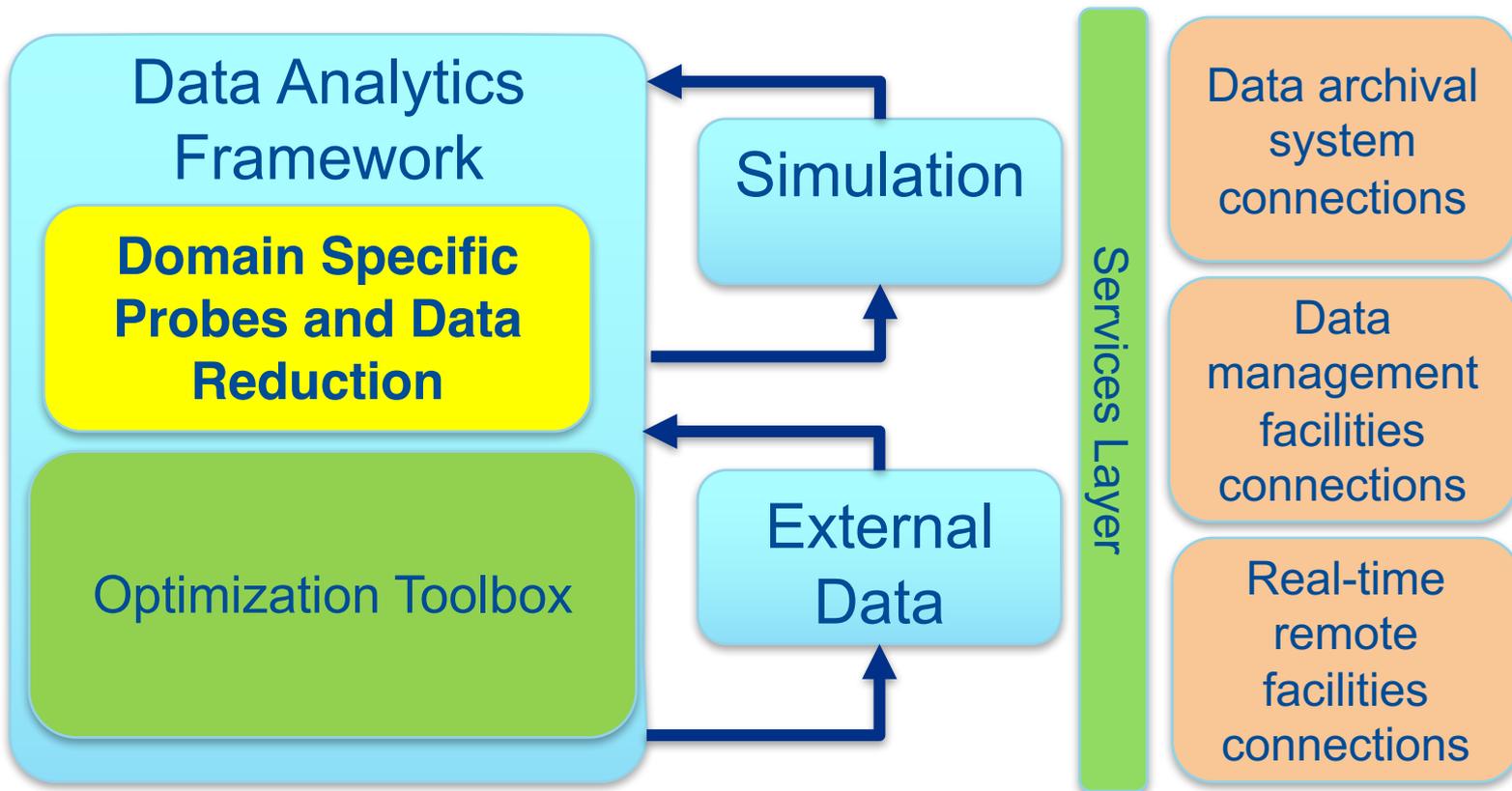
# Addressing I/O, data placement, and data locality

- Putting this in context with the major workflow
- Utilizing memory hierarchy
  - NVRAM / Burst buffer
  - K-V Stores / Dataspaces

High-speed storage  
between layers



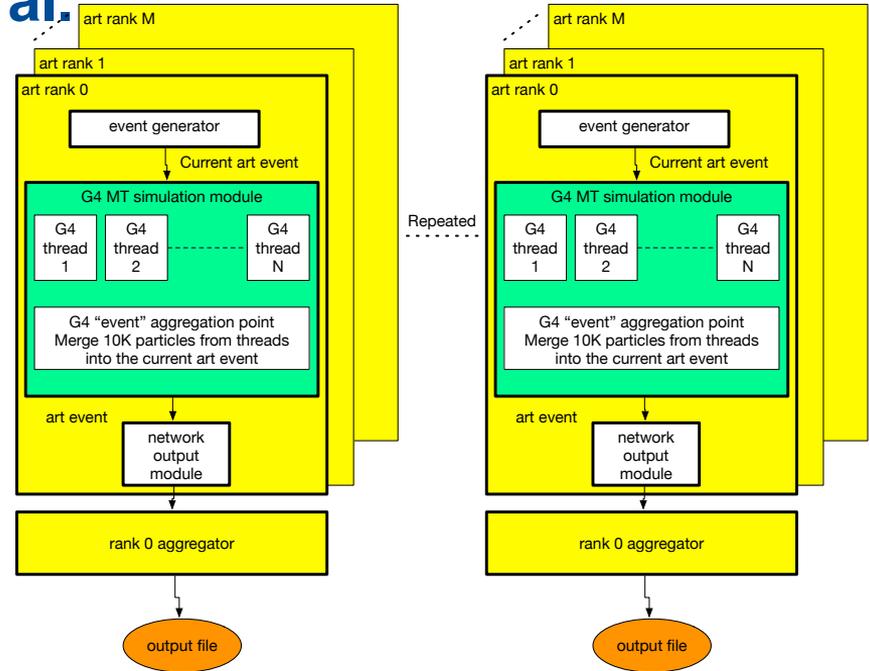
# Addressing data ingestion, archival, and provenance



## Where are we at now ...

# art-HPC: Addressing I/O trends, et al.

- Extending the ART Framework to Support Large Scale Multiprocessing for the Intensity Frontier
  - Partnership with ANL
  - Migration of art to HPC and Mira
  - Using MPI
  - Multi-threaded Geant4
- Target is to produce  $10^{12}$  muons for muon g-2 on ALCF Mira
- Architected to address
  - **limit I/O** to filesystem
  - scaling



NOTE: Same architecture applied to running a multi-parameter tuning of event generators using collider data analysis on Mira using Pythia

<https://cdcvs.fnal.gov/redmine/projects/art-hpc/wiki/>

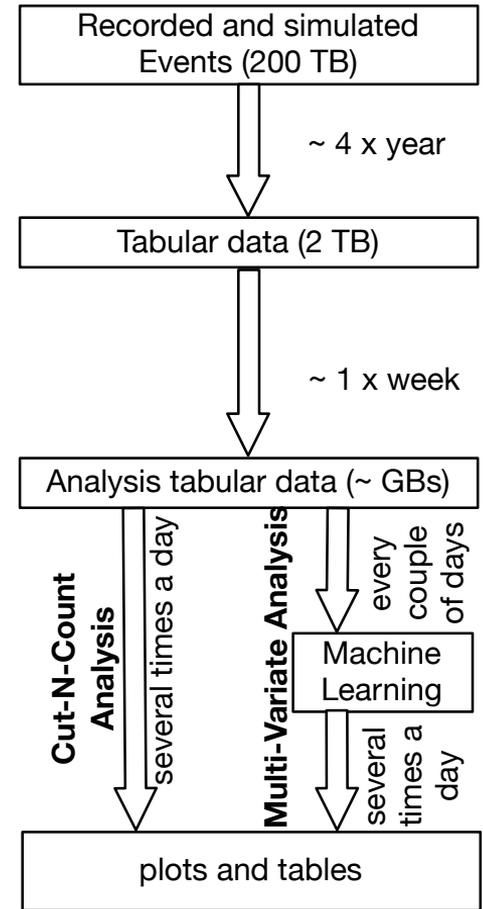
# Big Data Analytics: A CMS Dark Matter use case

## Data Organization:

- Data represented as rows that describe physics objects (particles) and the event in which they were seen
- 400kB/event and  $5 \times 10^8$  simulated events for backgrounds and signals

## Data Processing:

- Event-based processing, sequential file-based solution
- Batch processing on distributed computing farms
- 28,000 CPU hours to generate 2 TB tabular data, ~1 day of processing to generate GBs of analysis ready tabular data, 5 – 30 minutes to run end-user analysis



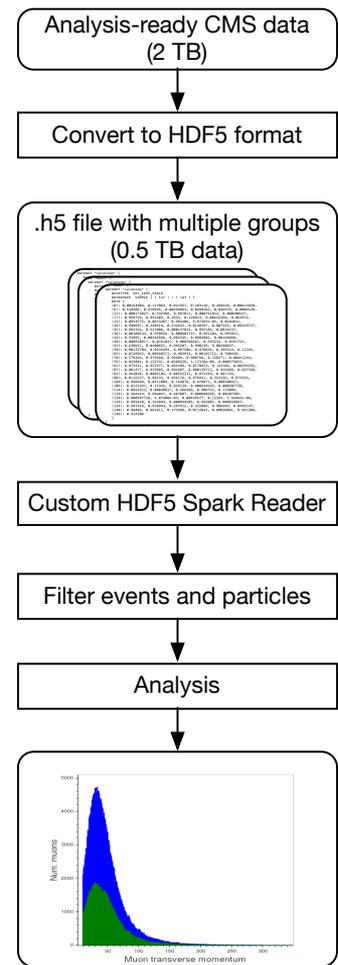
# Our approach to Big Data management

## Data organization:

- We convert the data to column oriented format in HDF5.
- HDF5 is a well-known format for the HPC systems; it also allows us to use non-big data technologies to process these files.
- Read HDF5 files into multiple Spark DataFrames, one Spark DataFrame per particle type

## Data processing

- Define filtering operations on a DataFrame as a whole instead of an event.
- Data is loaded once in memory and processed several times.
- Make plots, repeat as needed.



# I/O considerations underway

- The defacto standard ROOT I/O system is the major format for HEP data.
  - Automated persistency (serialization plus compression) of C++ object oriented data structured
  - No support for concurrency or use of HPC filesystems
- Current efforts are underway to efficiently utilize HDF5 for data storage
  - Already discussed for analysis results in the Big Data project
  - Addresses some of the concurrency issues
- Also require efficient transfer of data objects from process-to-process

# Programming languages

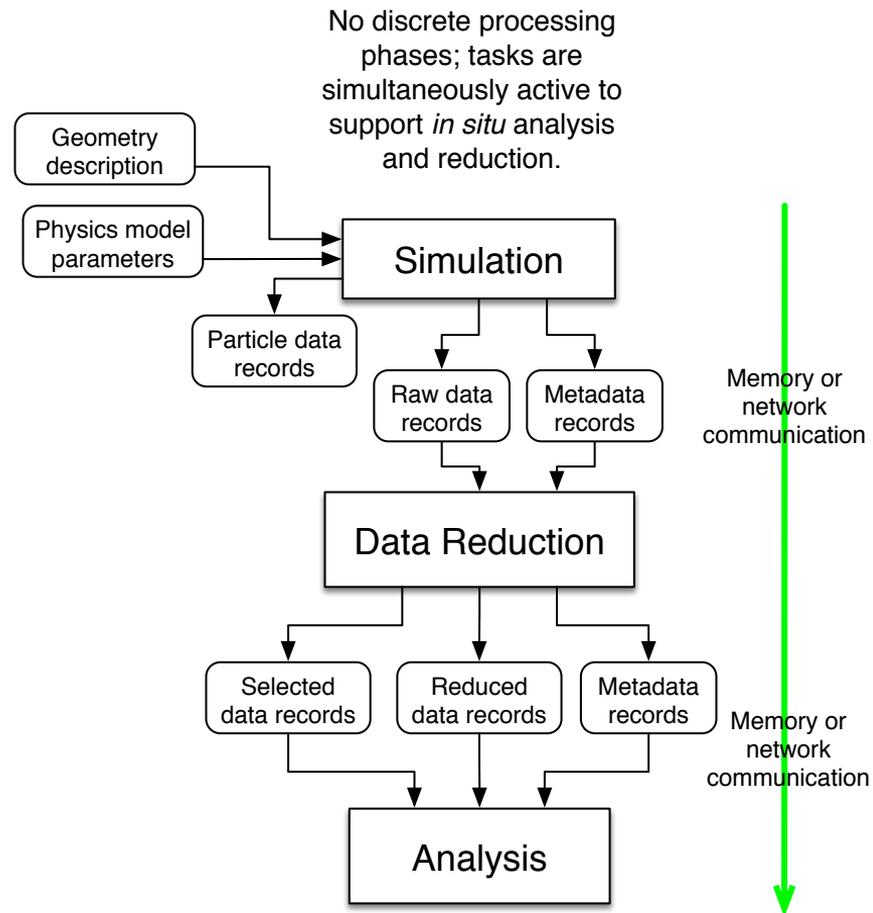
- Language is critical for making a system useable
- We have long employed advanced C++ in our systems
  - Already using features such as *variadic templates*, threads, and lambdas
  - Are exploring *concepts*, *ranges*, memory allocation controls
- Python is still the higher-level language of choice,
  - Partly because of its simplicity and ease of integration with C++
  - Partly because numpy/scipy, and matplotlib are available
- Nothing has yet appeared to supplant C++ and Python
  - Functional languages are very attractive
    - We will be using functional features of C++ as they appear
    - Could help enable parallelism
  - Julia is interesting, and R has been in use for high-level data manipulation

# Relationships and technologies

- Groups we've been talking with
  - HDF5 group for data model representation
  - ORNL Data Analytics group (machine learning optimizations)
  - ANL HPC data services and numerical optimization groups
  - NERSC for Big Data with Spark and working with containers
  - Kitware for visualization with ParaView
- Software technologies that have our attention
  - Flink (to compare with Spark)
  - We will continue our use of TBB, and watch HPX as it evolves
  - Eigen, Blaze, Armadillo for linear algebra

# Future

- Leading to on-demand simulation and reconstruction
  - Driven by science analysis goal, working backwards
  - Exascale path permits up to move towards these objectives
- Data path is different
  - Dataset (collected sample) grows in real time
  - On-demand processing permits the new data to be incorporated on-the-fly
  - Memory intermediate result caching permits reuse across user community



# Conclusion

- Frameworks have permitted successful collaboration amongst thousands of scientists and is essential for dealing with complexity
- HPC technologies are critical for advancing scientific computing involving complex measures from instruments
  - The hierarchical / distributed high-speed memory structure
- The scale of computing that is possible with HPC allows us to reduce the “time to science” through multi-stage workflows
  - Reduces the intermediate I/O processing that is built into the current computing model
- Working towards a software system capable of allowing for multi-scale automated tuning and control
  - Feedback to experimental system
  - Integration of simulations requiring fine-grained parallelism
  - Distinct large-scale optimization stages
- Partnering with experts from leadership facilities

# Bag of slides ...

# Addressing final data analytics

- (is this needed at all?)
- (was thinking of having it as a way to introduce big data technology as a useful component of the framework)

# Outline

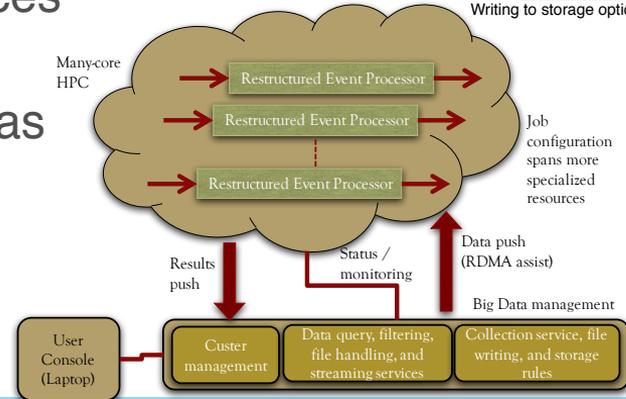
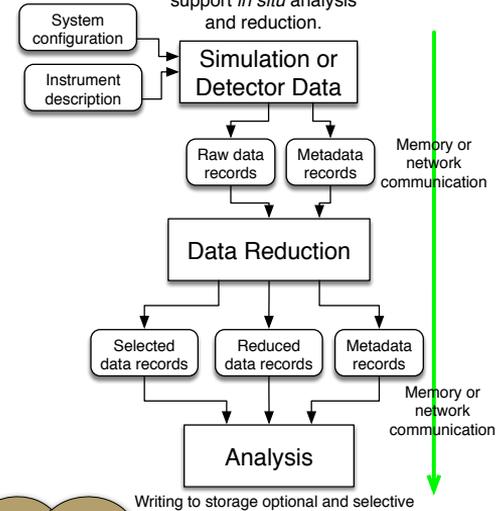
- Problem domain (sim and data coupled to reco and analysis, event processing and accelerator modeling, message visualization and data exploration)
- Usage information from HEP now and towards HL-LHC / DUNE
- What make data analytics possible with such a large group? (definitions and tools coverage)
- Can this be utilized or applied to exascale problems across a wider
- Chance for complex campaigning involving controls (expands reach)
- Challenges in moving towards specialized hardware and exascale (diverse resource needs)
- Addressing I/O, data placement, data locality
- Addressing data ingestion, archival, and provenance
- Addressing final data analytics
- On-demand simulation and reconstruction
- Use cases
- Relationships and technologies
- Direction

# Evolving architecture key elements

- Limit I/O to disk storage
- Utilize large vector units
- Efficient movement of data between processes in a distributed environment using high bandwidth networking
- Shared framework services within nodes
- Localized data caching, as in big data technology
- Tighter integration with workload / workflow management

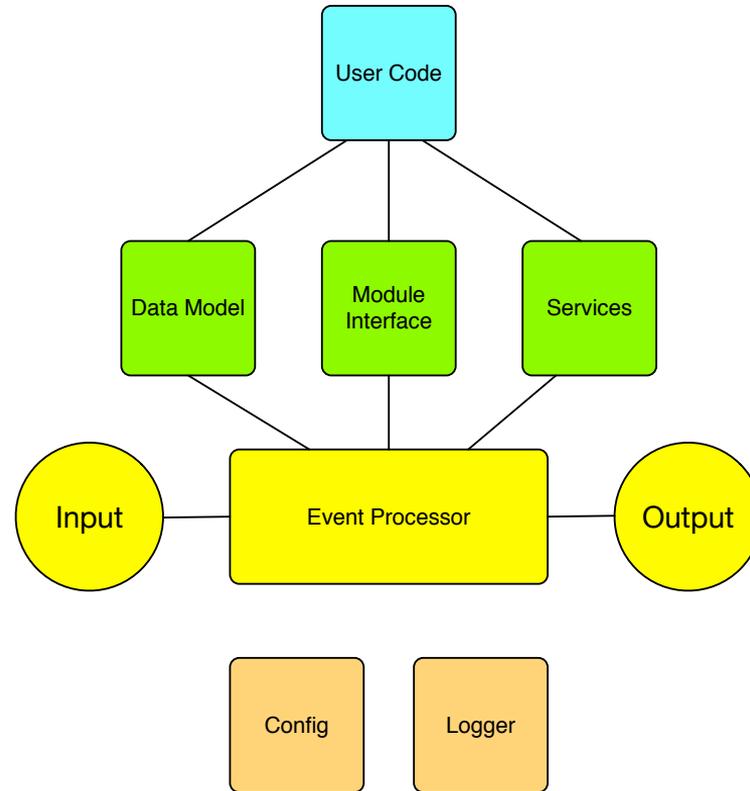
## Future Era

No discrete processing phases; tasks are simultaneously active to support *in situ* analysis and reduction.



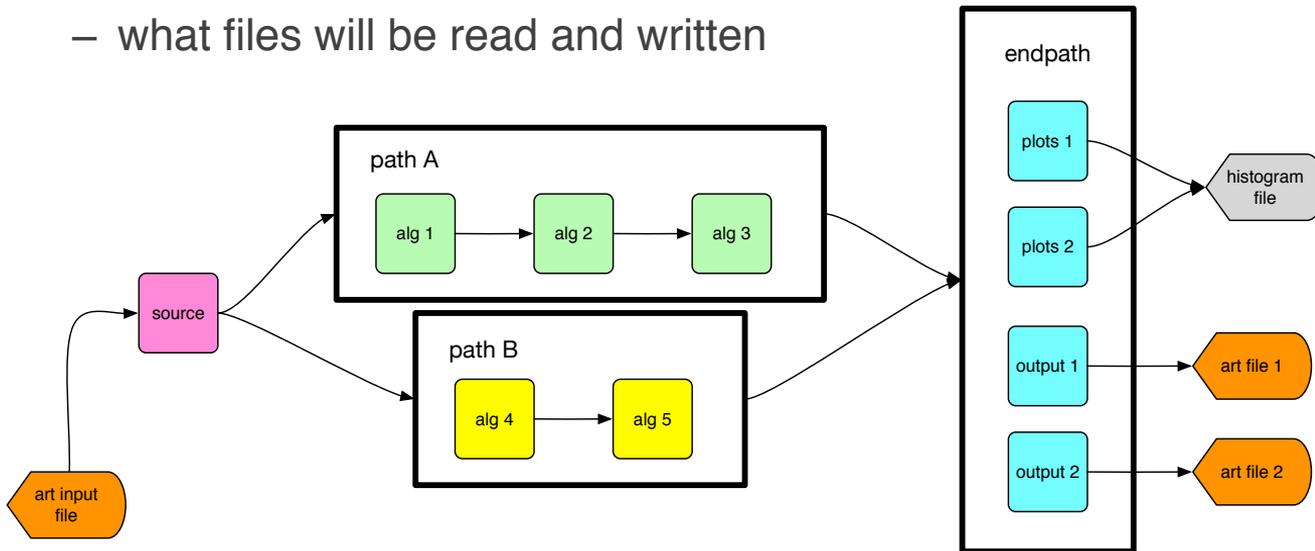
## What are the parts of the *art* framework?

- User code is what you and your colleagues provide.
- Services provide access to global facilities.
- Data model provides the representation of event data.
- Event processor is the “event loop”, the core of the framework.
- Configuration and logger systems can be used by everything.



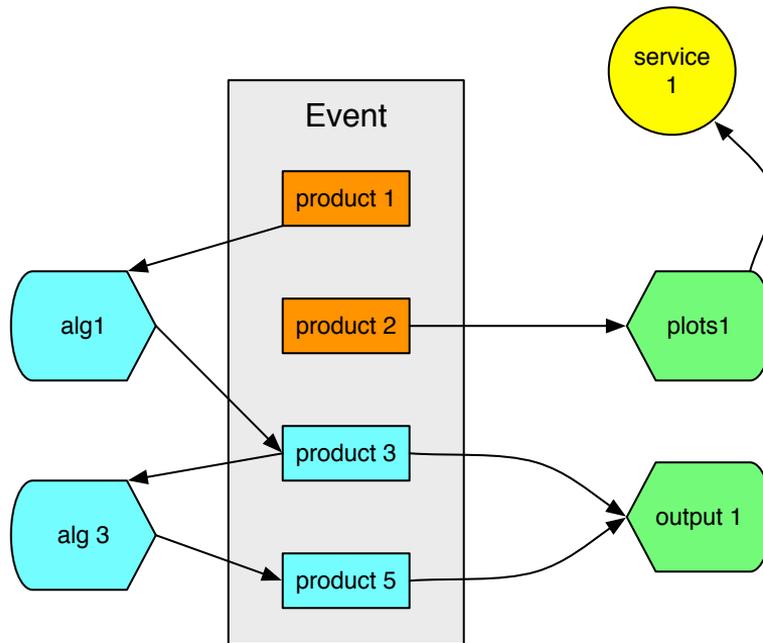
## Choosing algorithms to run

- Algorithms (simulation, reconstruction, or just analysis code) is built into classes, put into dynamic libraries called *modules*.
- Text files (in a language called FHiCL) declare
  - what modules will be loaded, and in what order they are to run
  - what files will be read and written



## Accessing data

- Modules *never* communicate with (call) other modules.
- Modules can call *services* (e.g., to create histograms managed by ROOT).
- Mostly, modules interact with an *Event*.
- An *Event* is just an organized collection of data products, with information about them (metadata).



# Problem domain

- sim and data coupled to reco and analysis, event processing and accelerator modeling, message visualization and data exploration

## Theme (temporary slide)

- Science that Fermilab is interested in, and some computing requirements
- Some of the good things we done with regards to computing
- So we have been moving towards HPC platforms in preparation for Cori, Summit, and Aurora. We believe it is possible to move further ahead than just packaging and running our applications as is.
- Taking Great leap from current HTC scientific data analysis (long latency workflow steps involving data and simulation and many files) practices to automated, integrated workflows heading into the exascale era, utilizing latest memory technology and optimizing with feedback loops.
- Here is where we are now, where difficulties lie, and how they are being addressed
- Here is our anticipated needs for computing.
- Here are the major experiments.