Retrospective and Future of Computing in HEP

Adoption, Scaling and Sustainability

Liz Sexton-Kennedy <lsexton@cern.ch>

Introduction

- Large HEP Collaborations
- Retrospective
  - The Past 3 Big Transitions
  - How they inform the present
- Future Challenges

Large HEP Collaborations

- CMS and ATLAS have many thousands of collaborators from all over the world with different funding structures and constraints.
- World wide resources have been needed for decades.
- CMS' latest census: 6122 members

~ 1300 undergraduate students

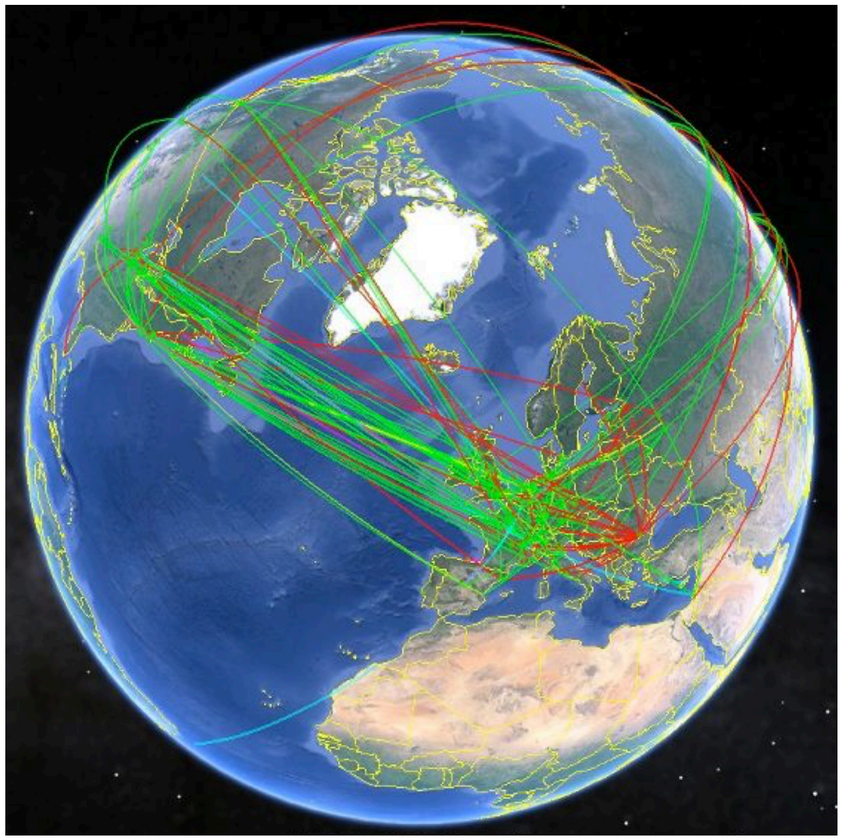
~ 1200 PhD students

~ 2100 PhD physicists

~ 1100 engineers

~ 400 technicians and administrative personnel

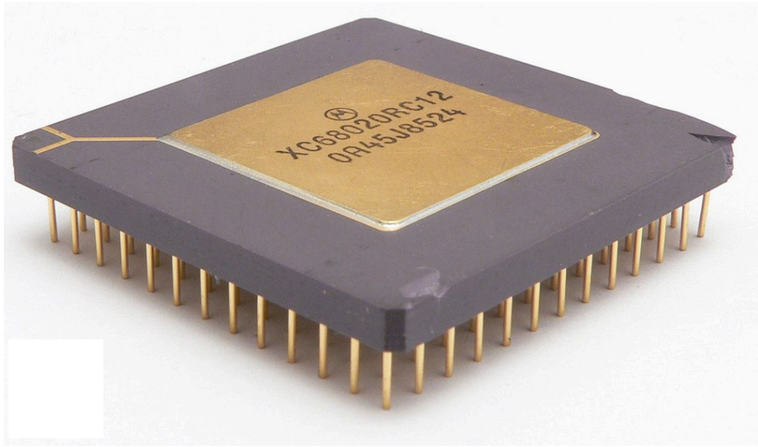
- The size of these collaborations create their own inertia.
  ->Reluctance to embrace change



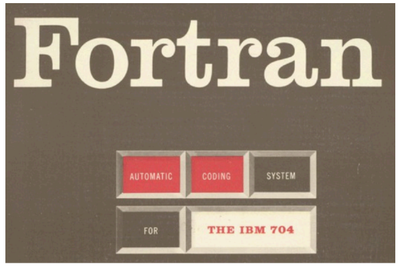
“Retrospective”

Early Computing

- In the early days of CDF we were building our own computers, ACP … and there was no internet …
  - Motorola processors on custom mother boards with no operating system
  - Maual management of very limited memory (overlays & DMA)
  - Fortran compiler was a “toy compiler” (used in game consoles)
- Offline used VAX/VMS with a real compiler and OS. It was a big contrast!
- DARPAnet exponentially grew into the internet. Fermilab became the 2nd American web site after a visit from Tim Burners Lee
- We were at the forefront of computing but then we needed to get over that



HEP Revolutions (v1)

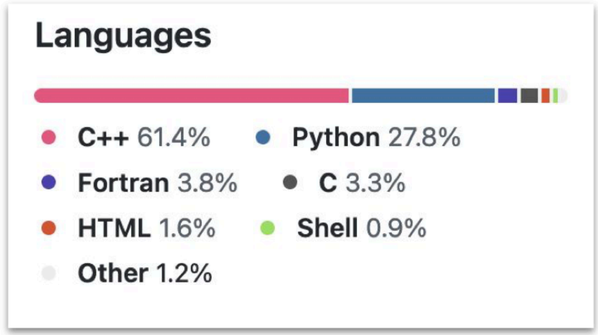


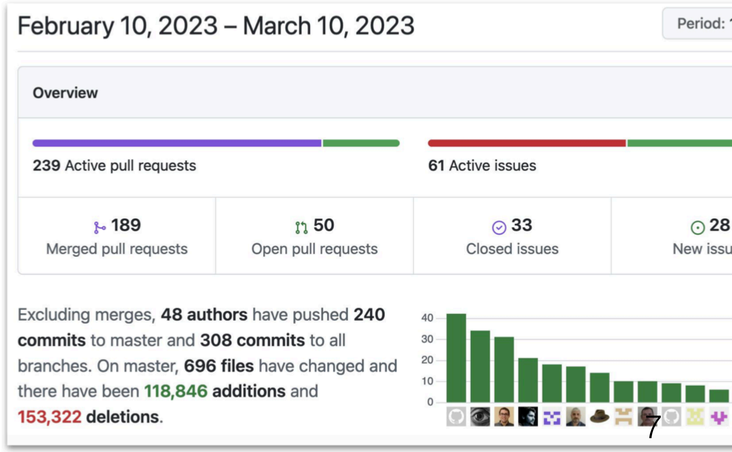
- Transition from Fortan77 / C to C++ in the mid-90s
  - Collaborations & detectors were smaller but becoming bigger.
  - Memory allocation and the need for data structures lead to the need for change. We were not being helped by the compiler at all.
  - Educated ourselves and a core group of algorithm/detector experts
  - Wrapped F77 code in C++ to allow for piece wise migration.
  - *Throughout the entire process the applications were kept running and validations were run demonstrating that the results were the same, or better.*

HEP Revolutions (v2)

- Transition from COBRA-Objectivity_Data_Model to CMSSW in the mid-00s

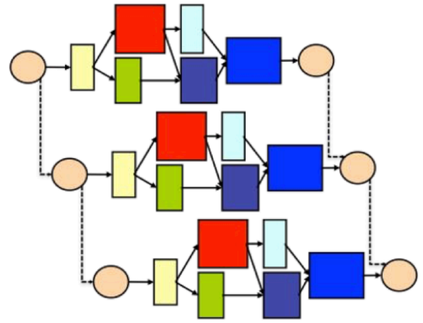
  - Mandated in CMS by the failure of the 2004 data challenge
  - Entire application originally in C++, later Python configuration added
  - Started in 2005 was ready for use in HLT cosmic data taking in 2006 with bare minimum algorithmic code. Was completely transitioned by 2008 data taking.
  - *Throughout the entire process the applications were kept running and validations were run demonstrating that the results were the same, or better.*





HEP Revolutions (v3)

- Transition from single threaded to concurrent multi-threaded applications - As of Run3 both ATLAS & CMS have transitioned
  - Memory and other OS resources in a many-core hardware evolution motivated change.
  - CMS started migration in 2014 by 2015 230 of 2270 modules were thread-friendly and running in the HLT in time for run 2.
  - Tooling was very important clang and helgrind allowed some centralization of the effort.
  - *Throughout the entire process the applications were kept running and validations were run demonstrating that the results were the same, or better.*



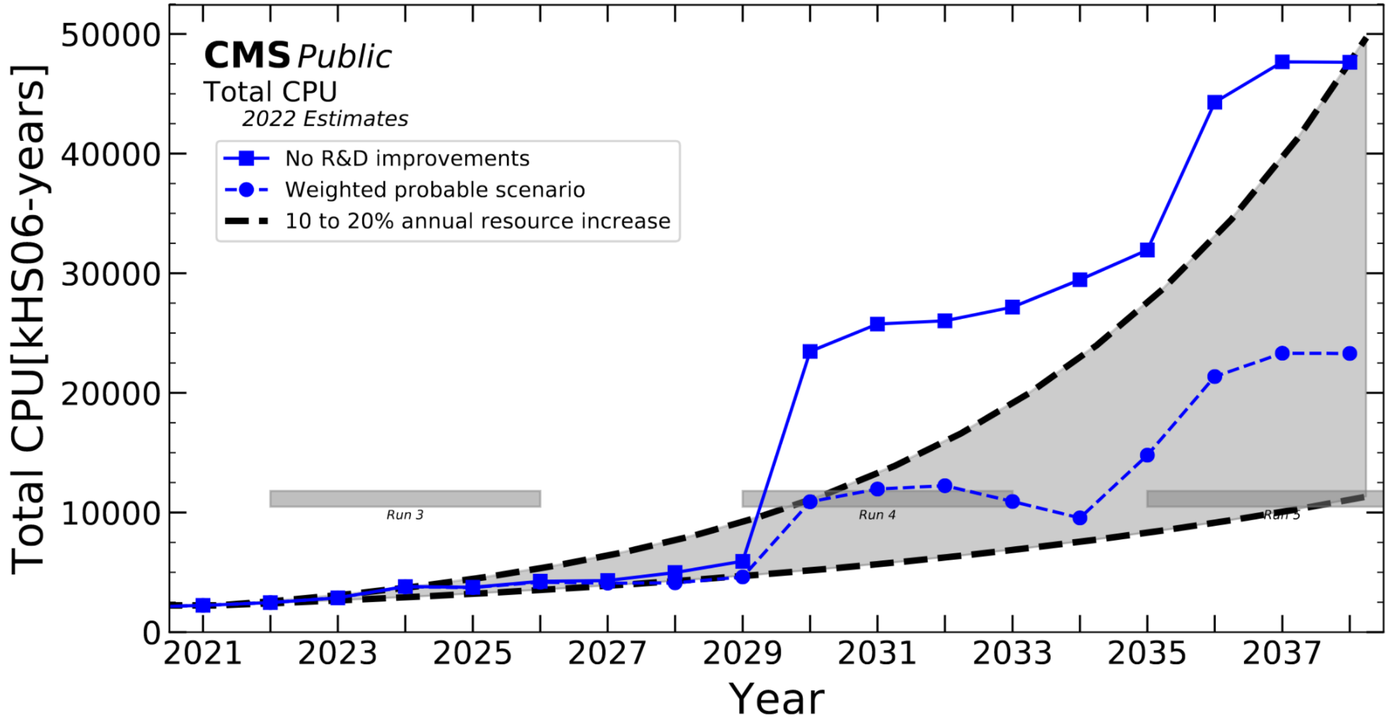
Parallelism at all levels

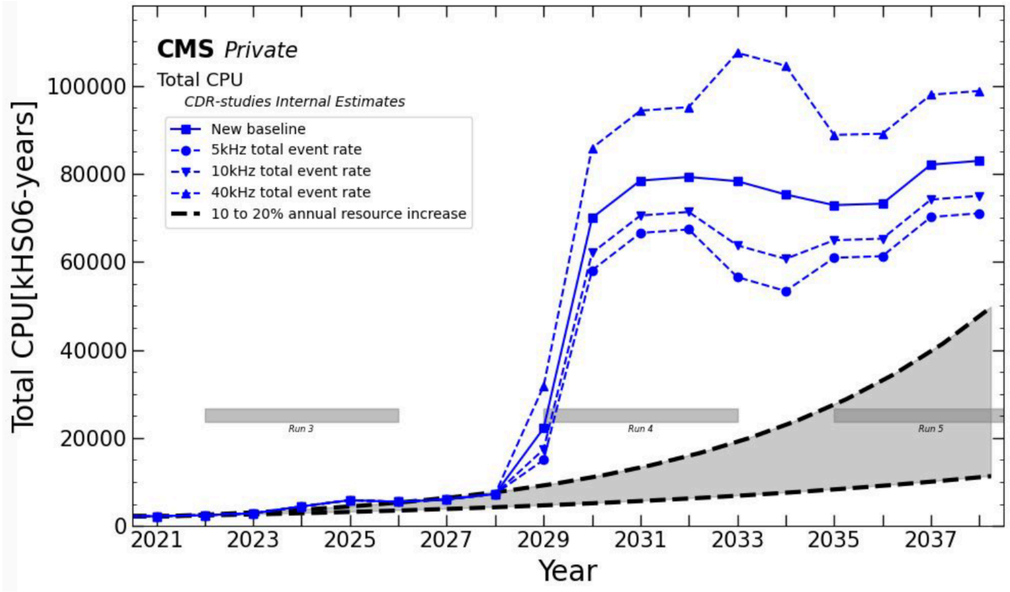
“Future Challenges”

Computing Needs Challenge

- What we said at the last 2022 update for CPU needs

- Changes in computing model parameters can have a big impact on needs





- Data velocity needs will also be a big challenge as we move to intensity physics

GPU Software R&D Needs

- We are loosing the simplicity of a homogeneous computing infrastructure.
  - Opportunity: Based on what has already been achieved on the HLT farms GPUs will go a long way to solving our computational complexity problems for HL-LHC
  - Challenge: Distributing such solutions to WLCG
- Further R&D effort in the area of frameworks and workload management is needed in order to enable operating our primary workflows across multiple heterogeneous hardware architectures.

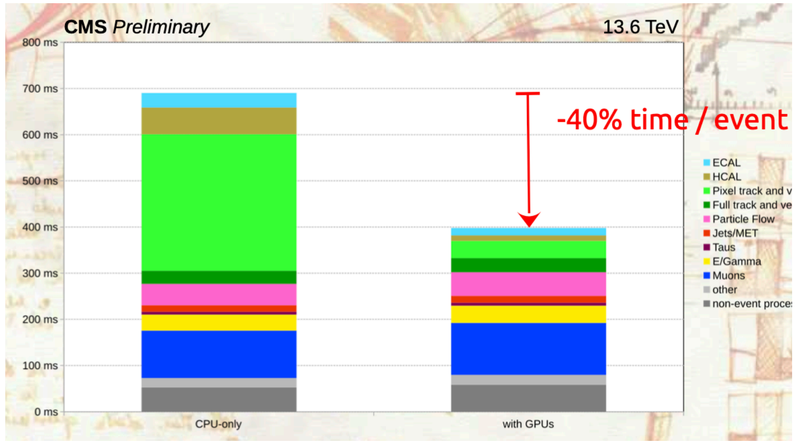
HEP Transitions (v4) - Needs and Opportunities

1. In the coming years computing hardware will again drive software revolutions.

   - The move to heterogeneous architectures, non-X86 CPUs, GPUs and other accelerators

2. The need for more precise numerical theory calculations

3. The rise of columnar analysis models to improve turnaround, debugging, and the needed data velocity for analysis.

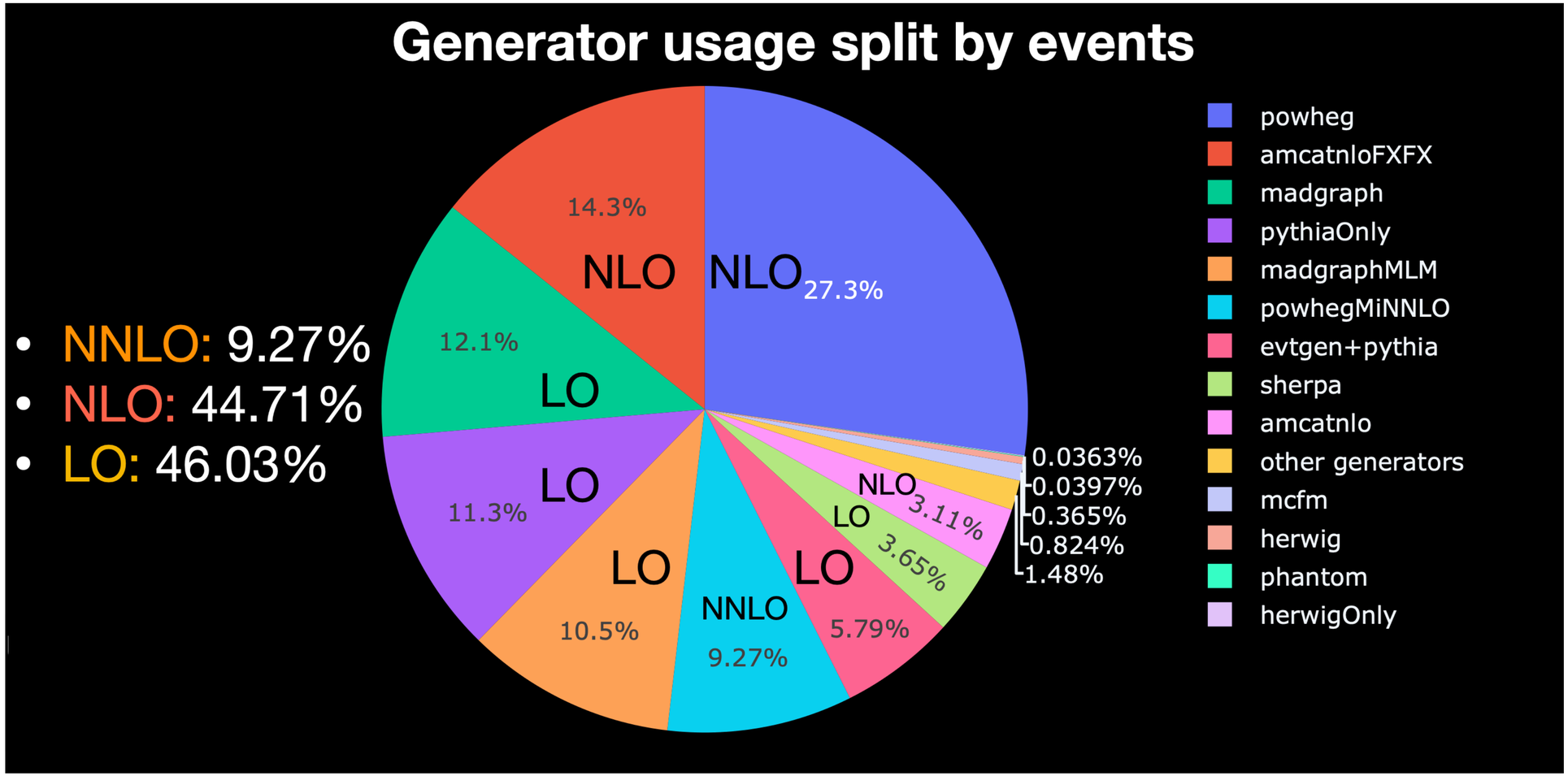


Realizing these types of gains on the computing GRID will be challenging

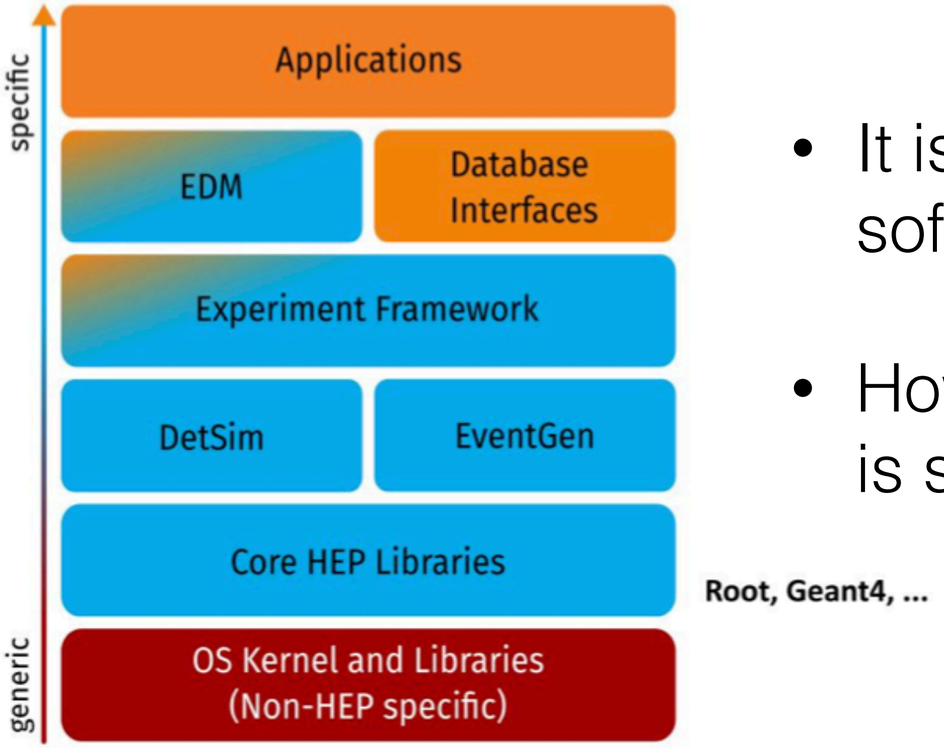
Other S&C Upgrade Needs

Generator Software R&D Needs

- The transition to NLO generators has happened for run2; dramatically increasing their CPU requirements. Deep interaction with generator authors is needed to keep this affordable.



Simulation Software R&D Needs



Standard HEP Software Stack

- It is possible to define an experiment independent software stack.
- However effort on common software tools like Geant is still an issue.

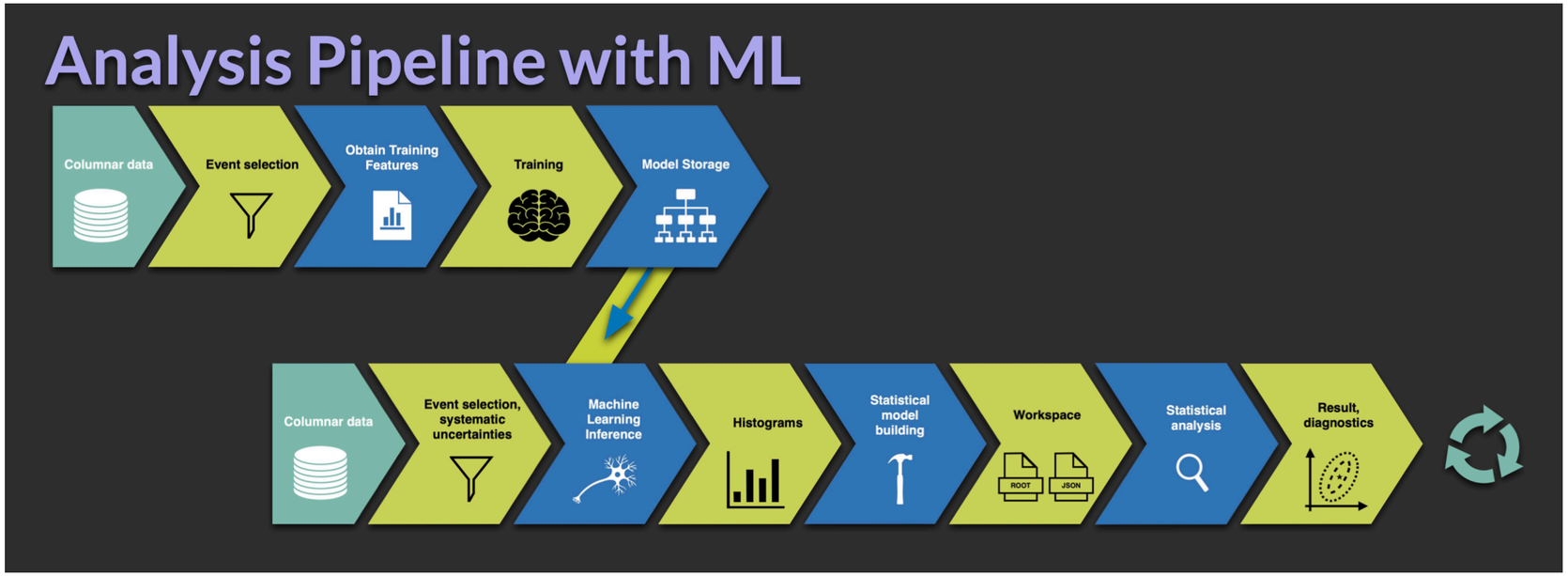
From D. Elvira's nice summary talk at CHEP of the Snowmass Process for the Computing Frontier:

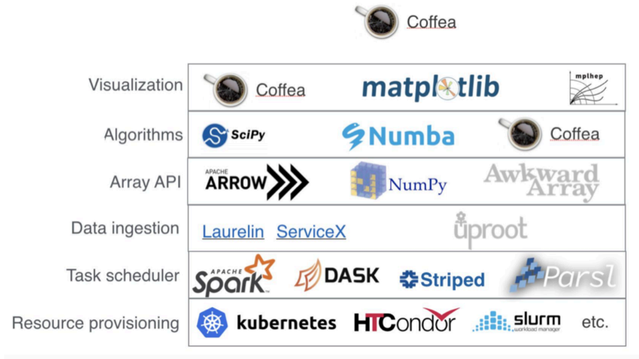
> *Recommendation*: **the US HEP community should take a leading role in long-term development, maintenance, and user support of essential software packages** with targeted investment.

Columnar Analysis

- There is more resistance to this change then I expected. It is time to move from R&D to deployment -> 200 Gb challenge, to see if it is justified.

- It will play out as BDTs did in the 90s. When sensitivity to rare processes increases because of these new analysis methods, they will inevitably win out.





Elliott Kauffman

Disruptive Technologies

"New directions in science are launched by new tools much more often than by new concepts. The effect of a concept-driven revolution is to explain old things in new ways. The effect of a tool-driven revolution is to discover new things that have to be explained."

– FREEMAN DYSON

- In the 90s ML was a new tool
- Shallow networks were what we could afford
- We are now solidly in the era of deep networks which increase our physics potential

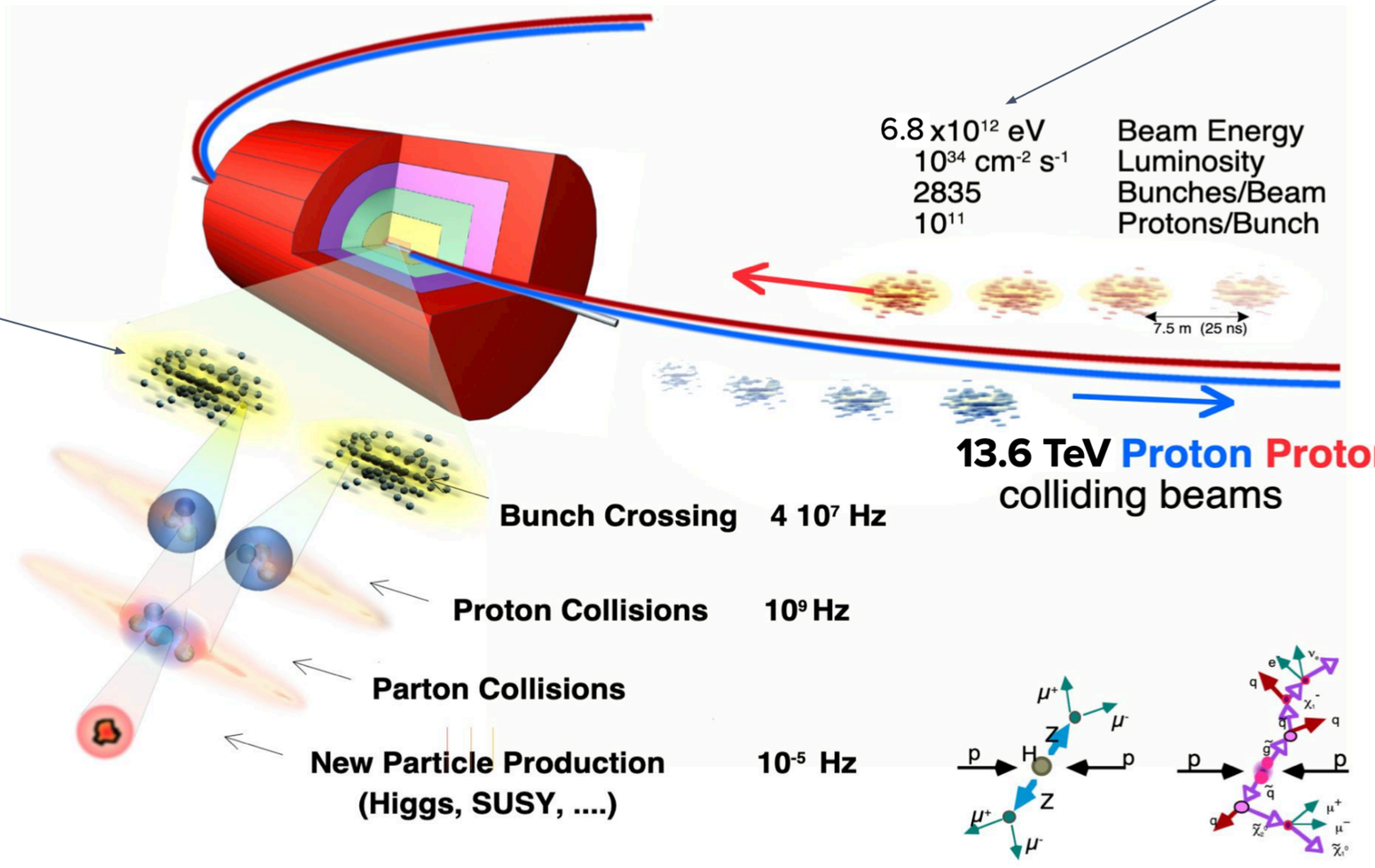
Kyle Crammer's CHEP talk Transformational ML

Backup

Collisions at the LHC: Summary

Approximate values!

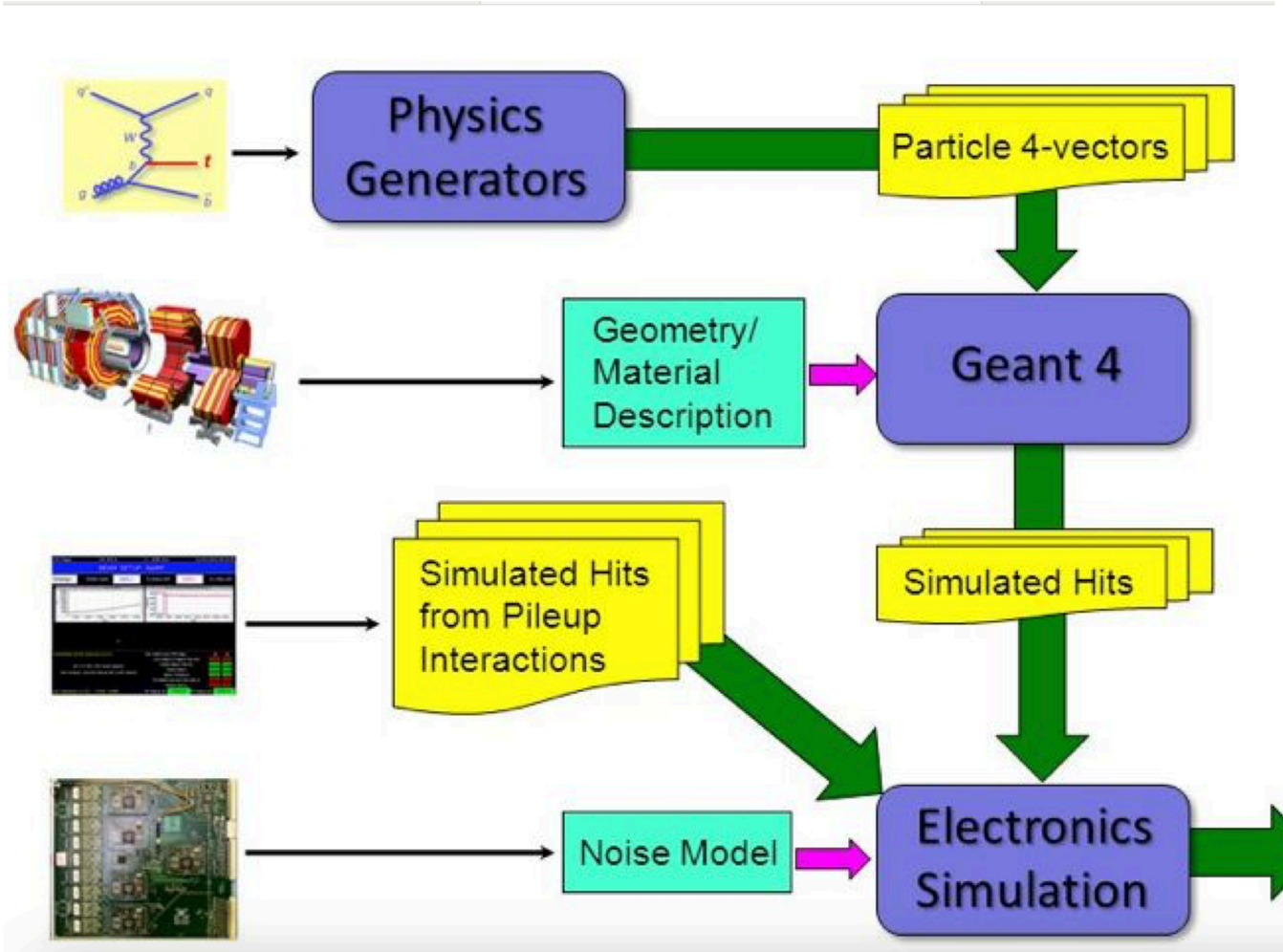
Key concept: bunches of protons collide, generating several parasitic collisions on top of the interesting one (if any)



Adapted from V. Innocente, 3rd Openlab Workshop on Numerical Computing

Trigger selection: ~1 event in 1,000,000,000,000

HEP Workflows



GEN:
Hard scattering, Multiple interaction, Hadronization

What physics are we studying?
What generator should be used?
Which parameters should be modified?

SIM

CMS Geometry, Magnetic field,...

DIGI, L1, DIGI2RAW, HLT

Detector electronics simulation, Pileup situation, Alignment-Calibration, Trigger, ...

RAW2DIGI, L1Reco, RECO

Where real data and MC meet
Reco. algorithms, High level object creation

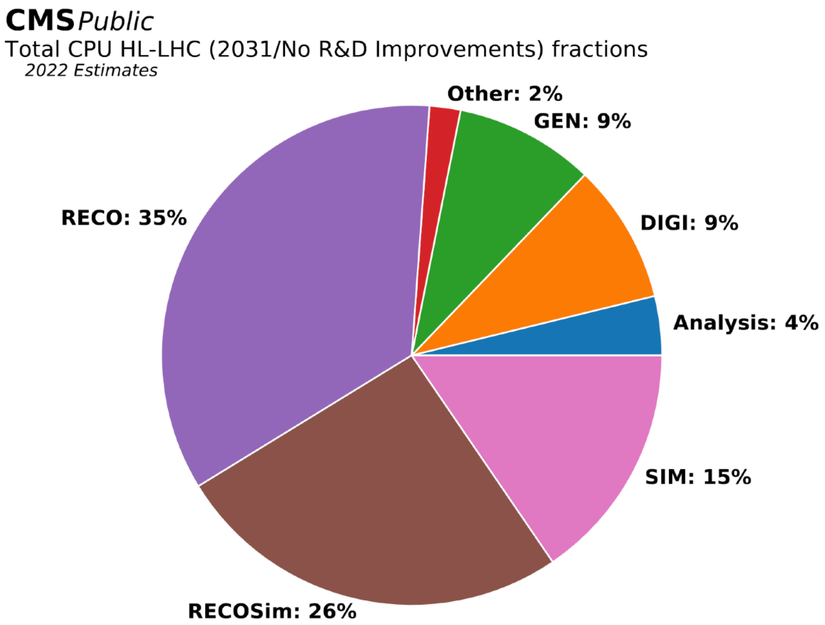
MiniAOD, NanoAOD

Data reduction

Your analysis

Different experts are needed at every step

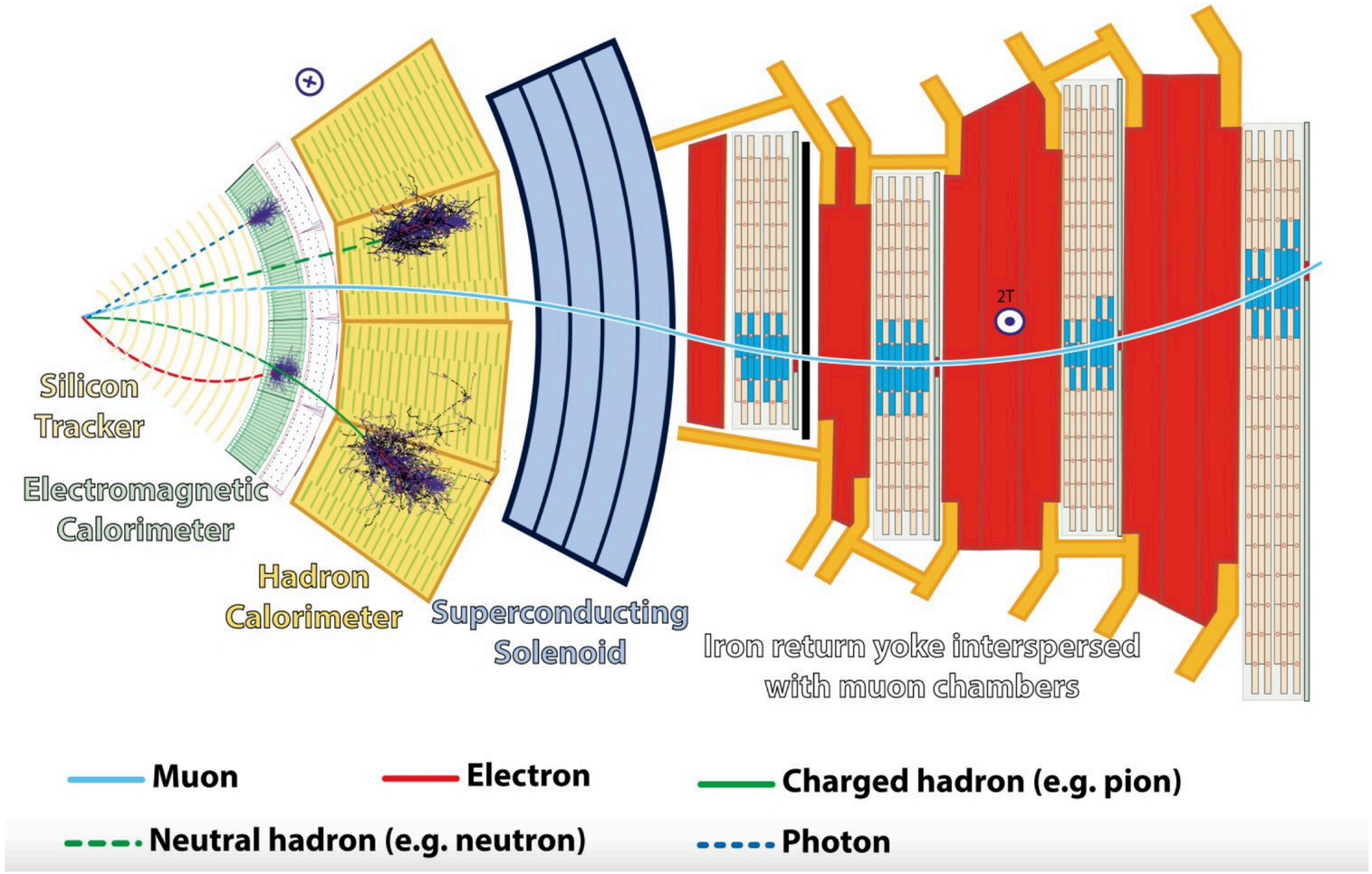
Workflow Steps CPU Fraction



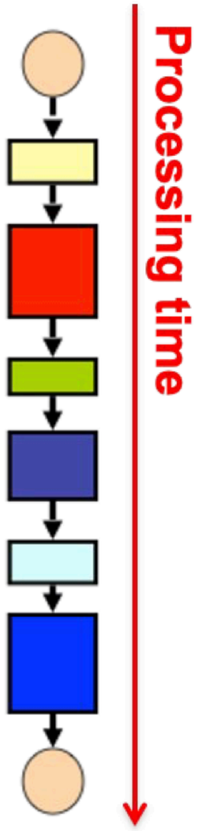
- Amount of CPU resources that go into each step on the previous slide as a fraction of the total amount of CMS computing resources.

Modular Architectures

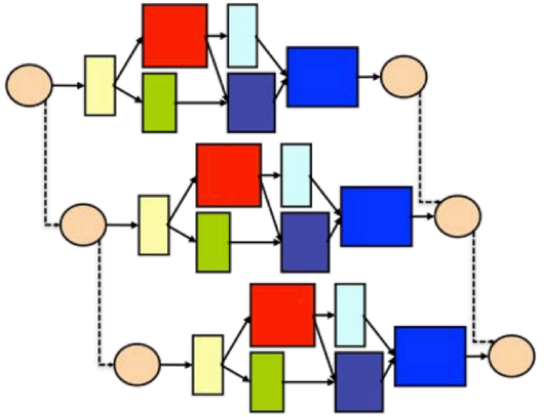
- Large teams of developers require modular architectures with well defined interfaces.
- The goal of the reconstruction workflow is to identify final state particles seen in the detector.
- Modules or Algorithms declare what they need as input and what objects they will output. Once an object is output, it is read-only.



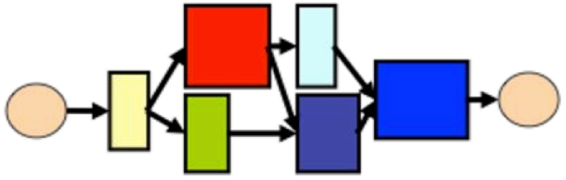
From Sequential to Parallel



Our data processing software is parallelised through a multithreading approach and can use accelerators such as GPUs



Parallelism at all levels



Parallelism within an event