# Challenges in high-energy physics computing

Andrei Gheata

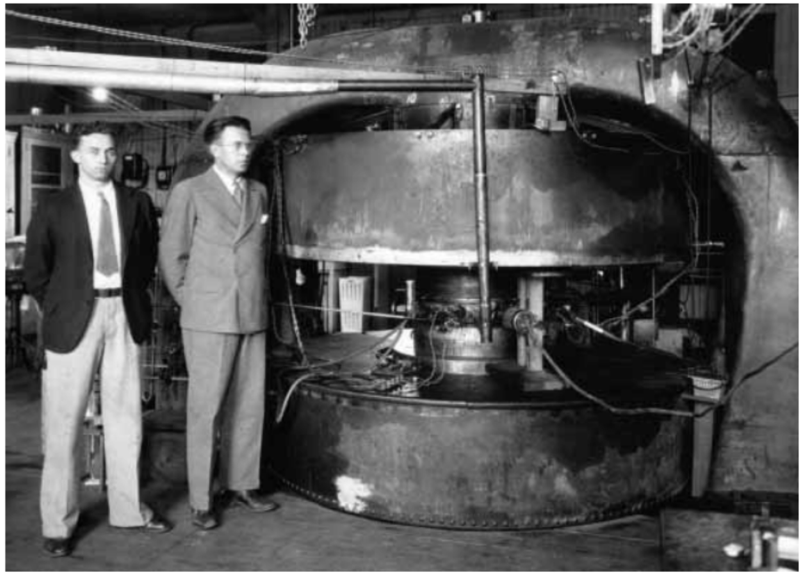
EP/SFT

CERN

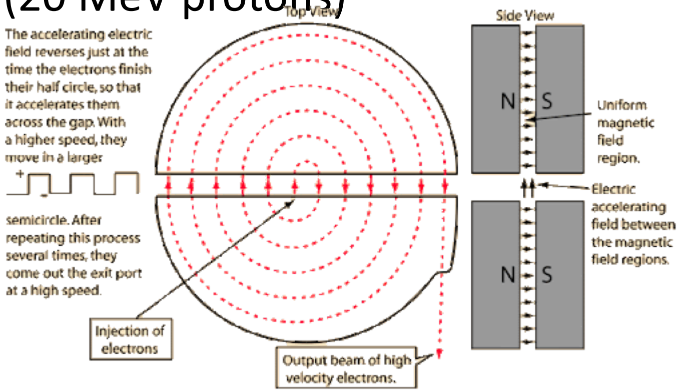
# Outline

- Modern High Energy Physics (HEP) experiments
- Physics software
  - online processing
    - triggering, selection
  - offline processing
  - simulation
- Conclusions

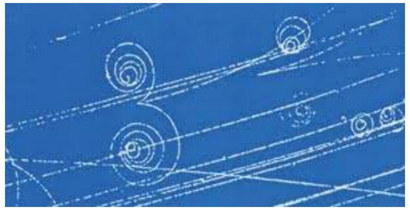
# Accelerators & particle physics

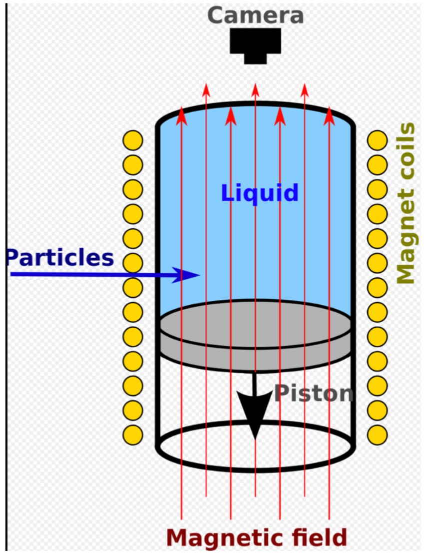


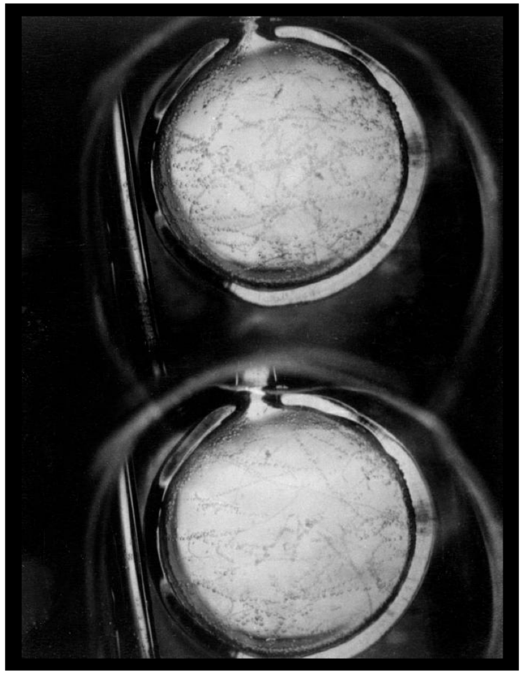
S. Livingstone and E. Lawrence
69cm cyclotron (20 MeV protons)




Bubble chamber experiments





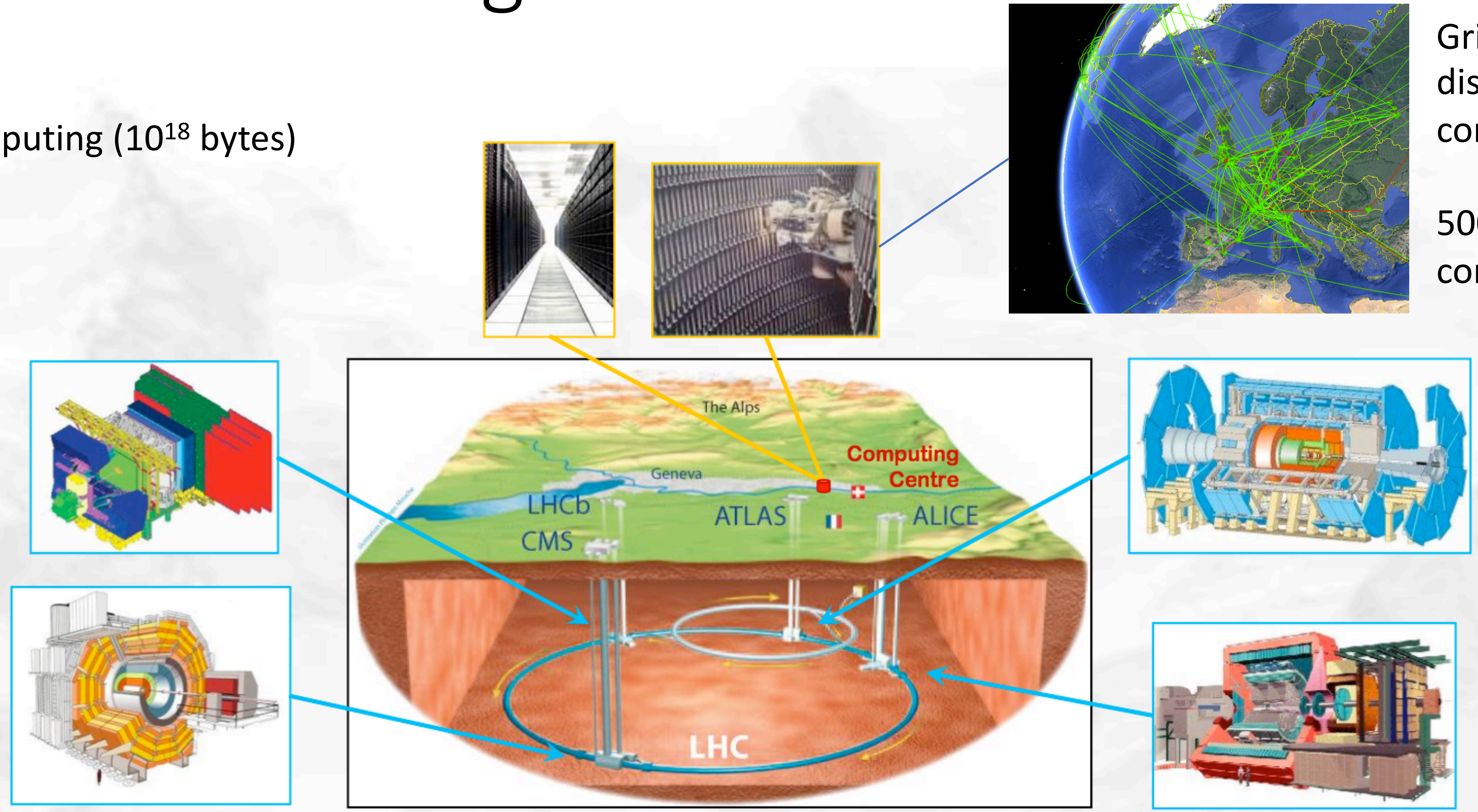
BIg European Bubble Chamber (Microcosm)

# Big science –> big data

Exascale computing ($10^{18}$ bytes)

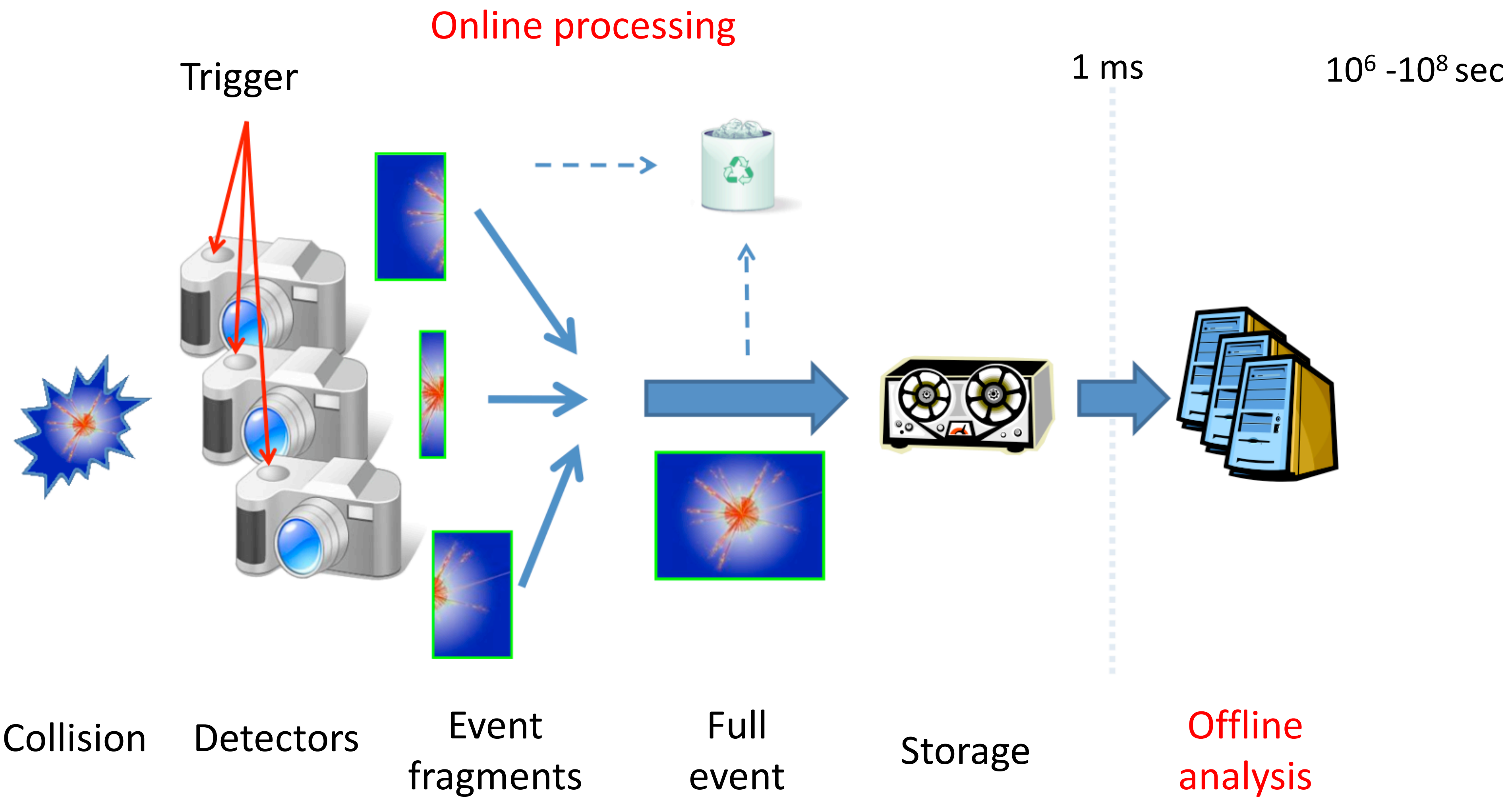
Grid/cloud distributed computing

500k processor cores

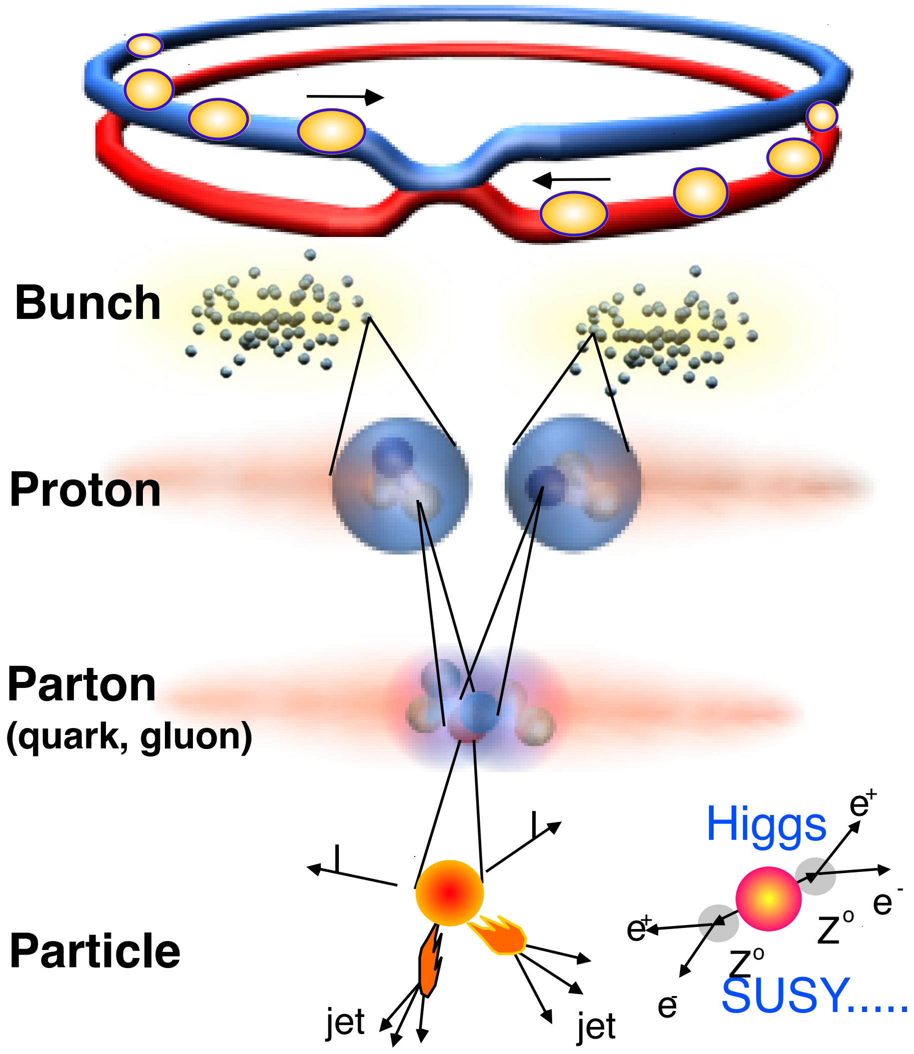



800 million pp collisions per second

# The data processing chain

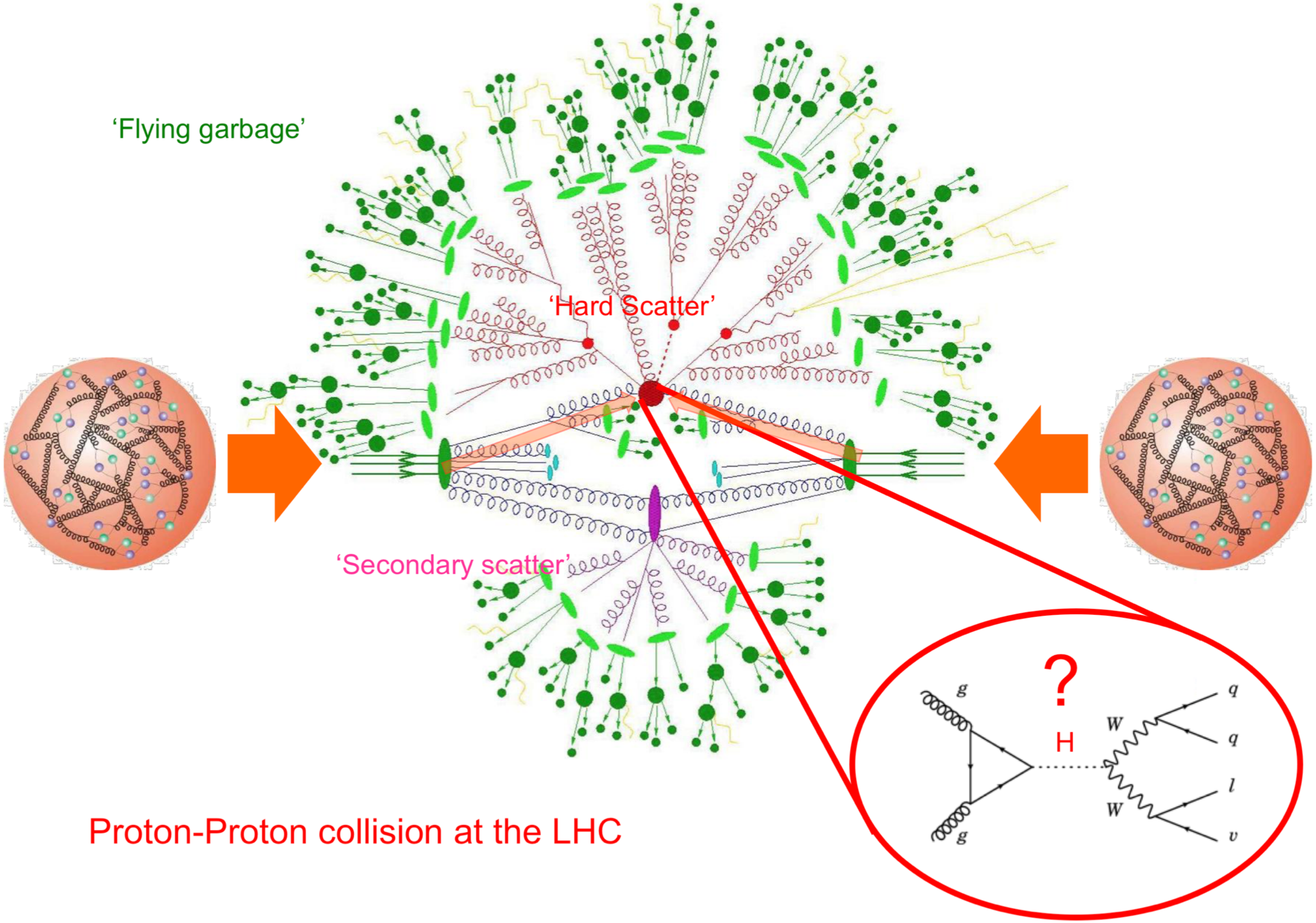


# Collisions at the LHC



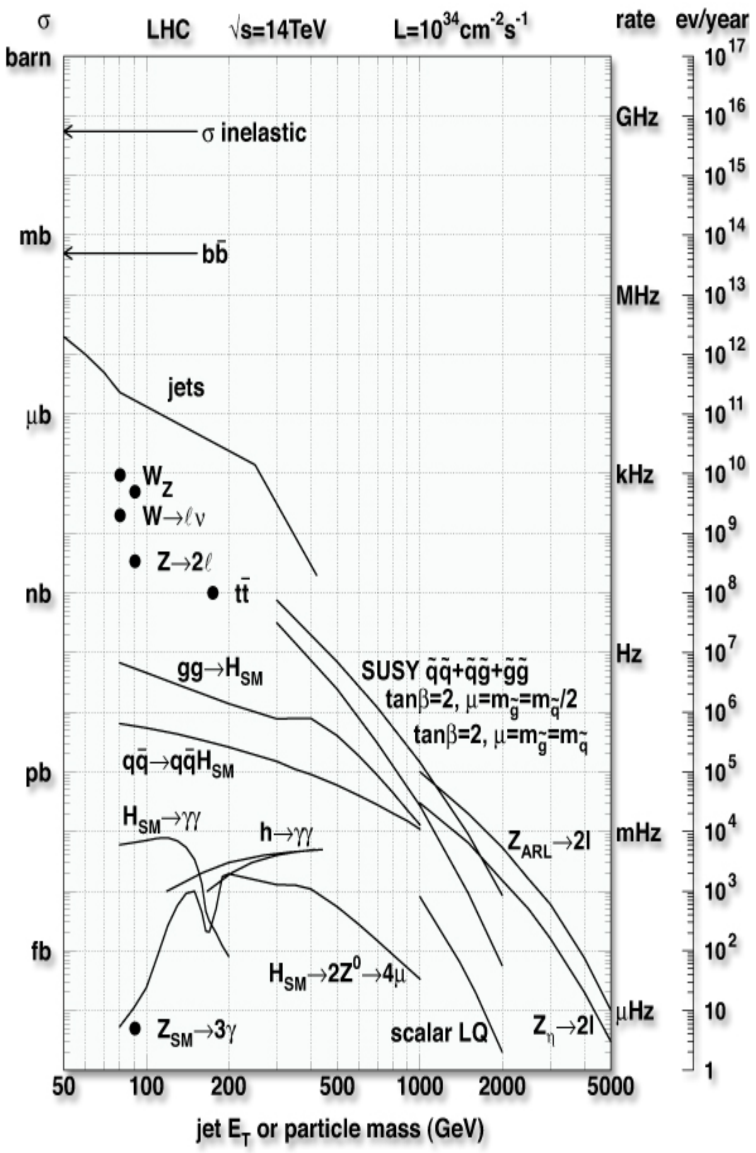

| | |
|---|---|
| **Proton - Proton** | **2804 bunch/beam** |
| **Protons/bunch** | **$10^{11}$** |
| **Beam energy** | **7 TeV ($7x10^{12}$ eV)** |
| **Luminosity** | **$10^{34}cm^{-2}s^{-1}$** |
| **Crossing rate** | **40 MHz (25 ns)** |
| **Collision rate ≈** | **$10^7$-$10^9$ Hz** |

# Searching the Higgs – needle in a haystack




Proton-Proton collision at the LHC

# The needle in many, many haystacks

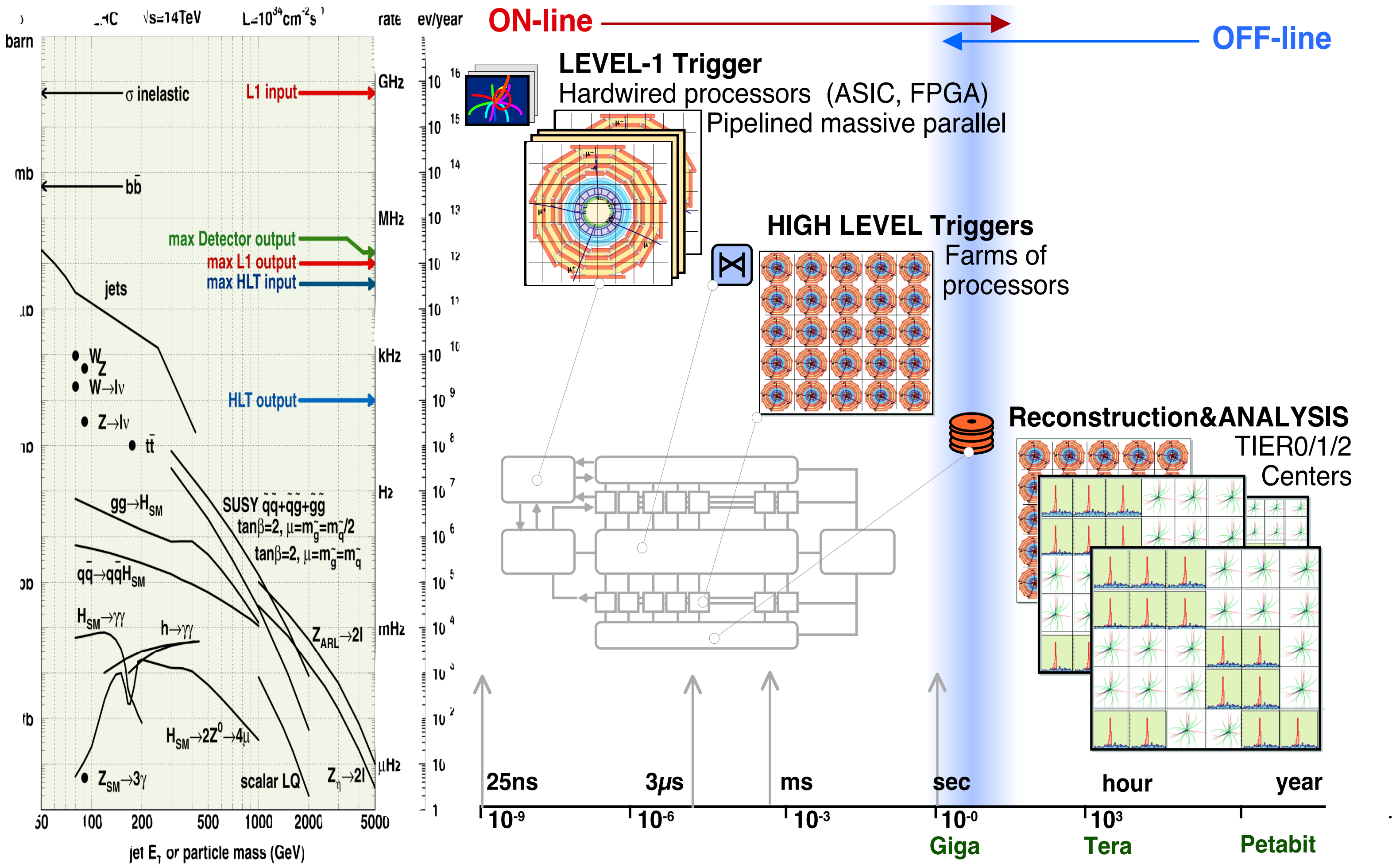



- Cross sections (probabilities) of physics processes vary over many orders of magnitude
  - Inelastic: GHz
  - $W \to \ell \nu$: 100 Hz
  - t $t_{bar}$ production: 10 Hz
  - Higgs (125 GeV/$c^2$): 0.1 Hz
- Selection needed: $1:10^{10-11}$

one Higgs on 10.000.000.000 collisions

~3 million until 2017

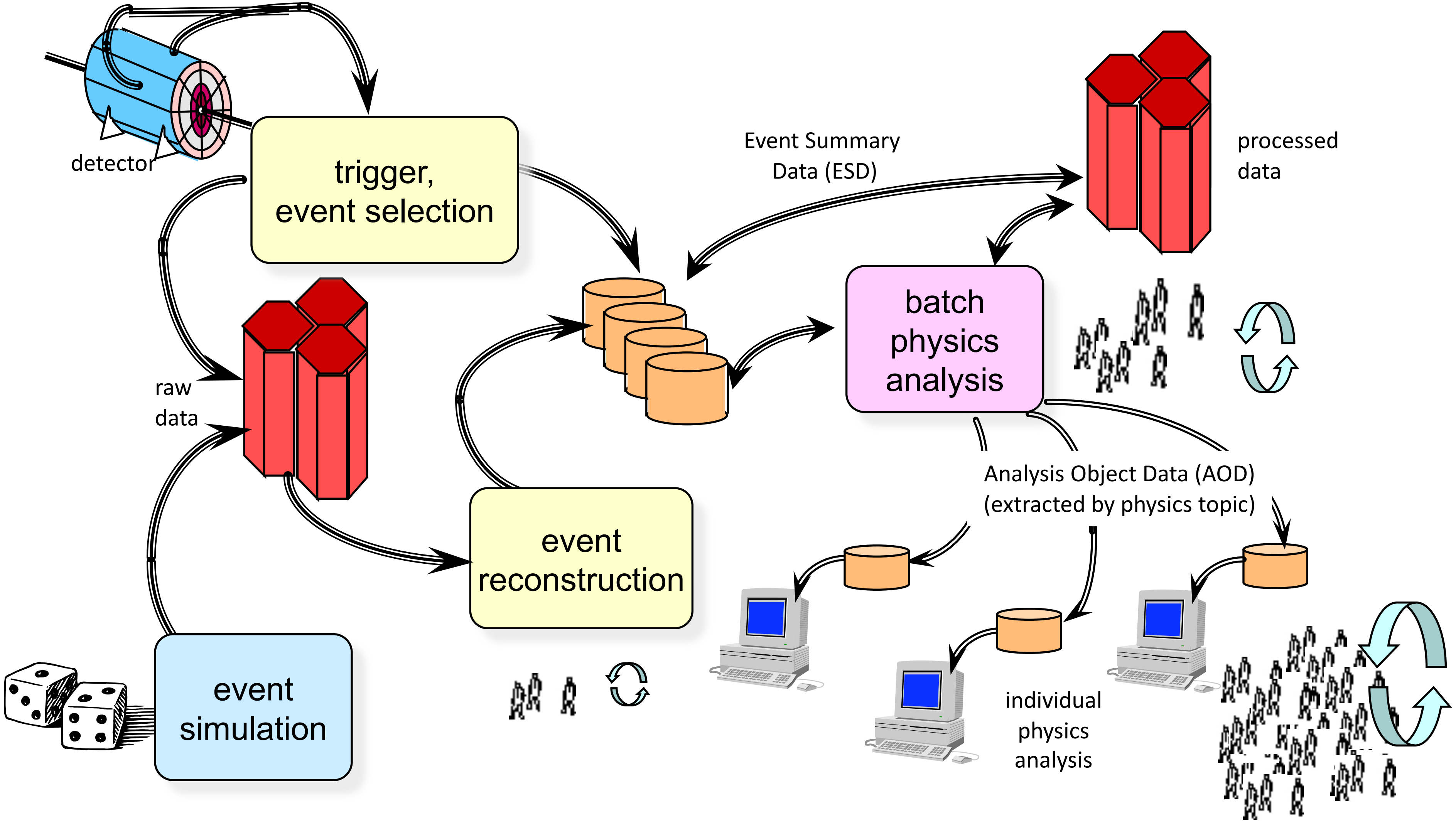
# Physics Selection at LHC



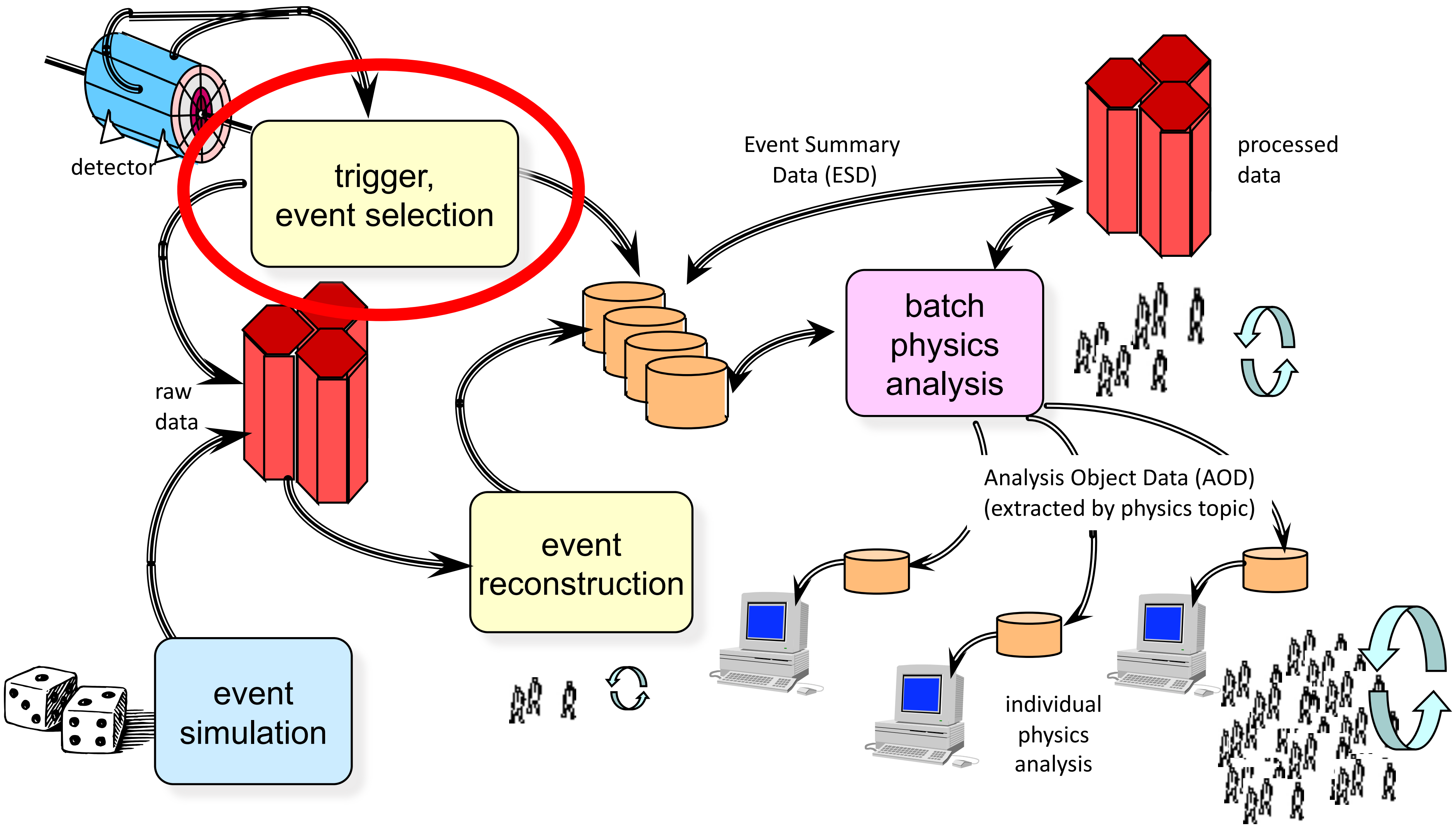
# Physics software

- The scientific software needed to process this huge amount of data from the LHC detectors is developed by the LHC collaborations
  - Must cope with the unprecedented conditions and challenges (trigger rate, data volumes, etc.)
  - Each collaboration has written millions of lines of code
- Modern technologies and methods
  - Object-oriented programming languages and frameworks
  - Re-use of a number of generic and domain-specific ‘open-source’ packages
- The organization of this large software production activity is by itself a huge challenge
  - Large number of developers distributed worldwide
  - Integration and validation require large efforts

# Processing Stages



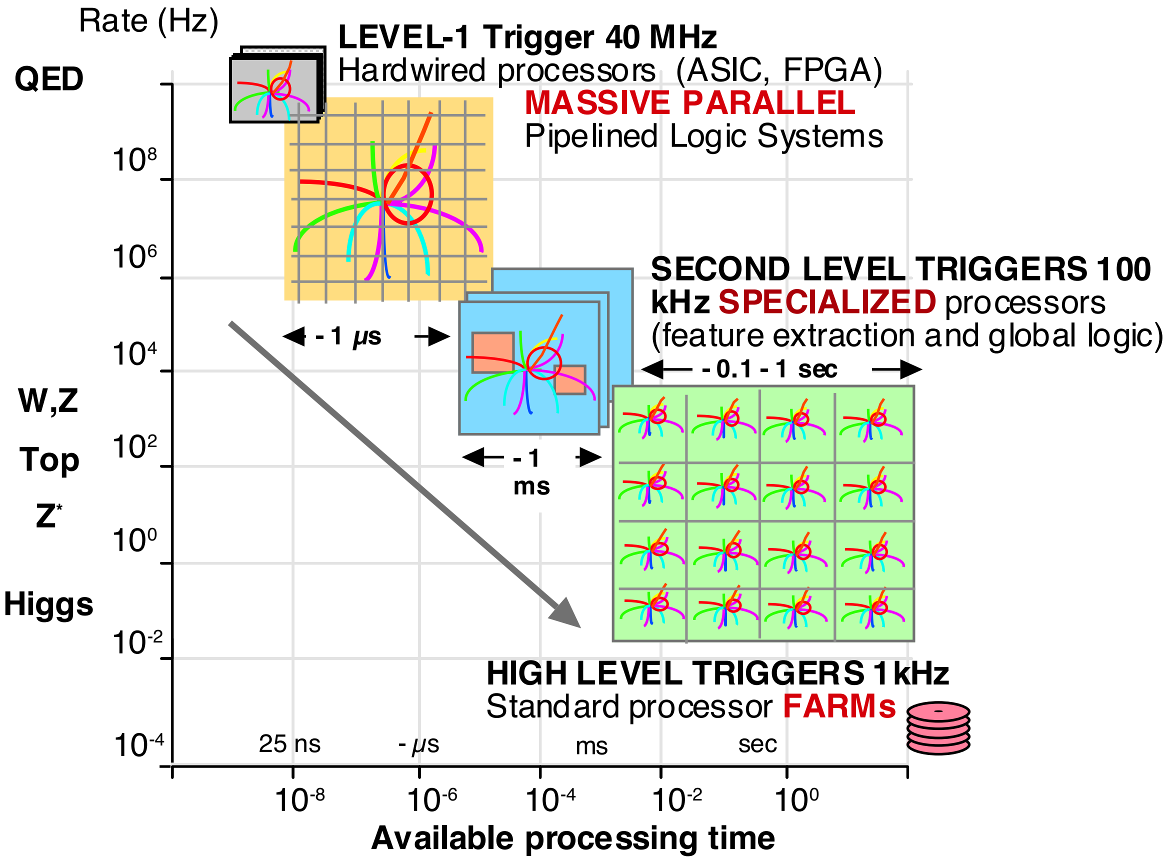
# Processing Stages - Trigger

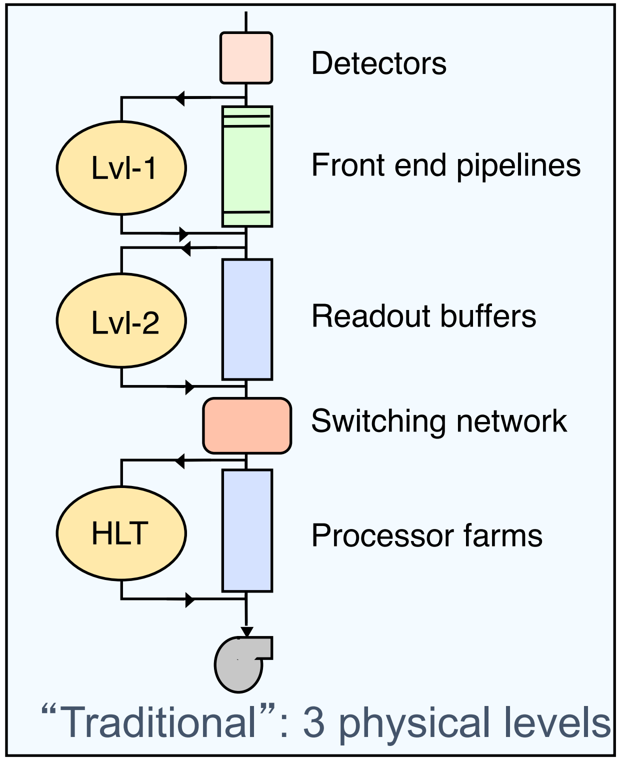


# Trigger Levels

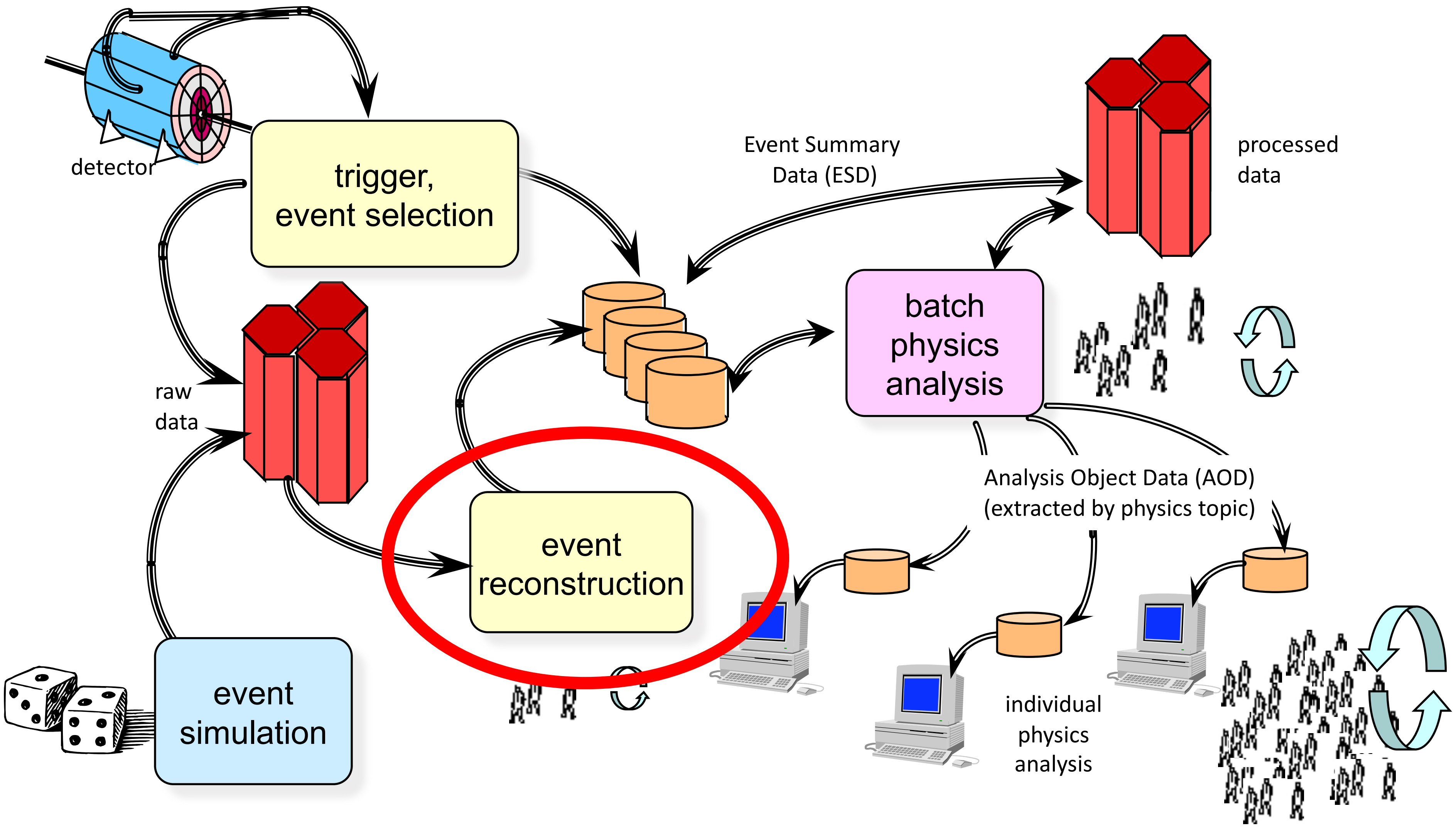
- Level-1 ~ $1:10^4$
  - Hardwired processors (ASIC, FPGA, …)
  - Pipelined massive parallel
  - Partial information, quick and simple event characteristics (pt, total energy, etc.)
  - 3-4 μs maximum latency
- Level-2 (optional) ~ $1:10^1$
  - Specialized processors using partial data
- High Level
  - Software running in processor farms
  - Complex algorithms using complete event information
  - Latency at the level of fractions of second ~ $1:10^2$
  - Output rate adjusted to what can be afforded

# Trigger Levels and Rates



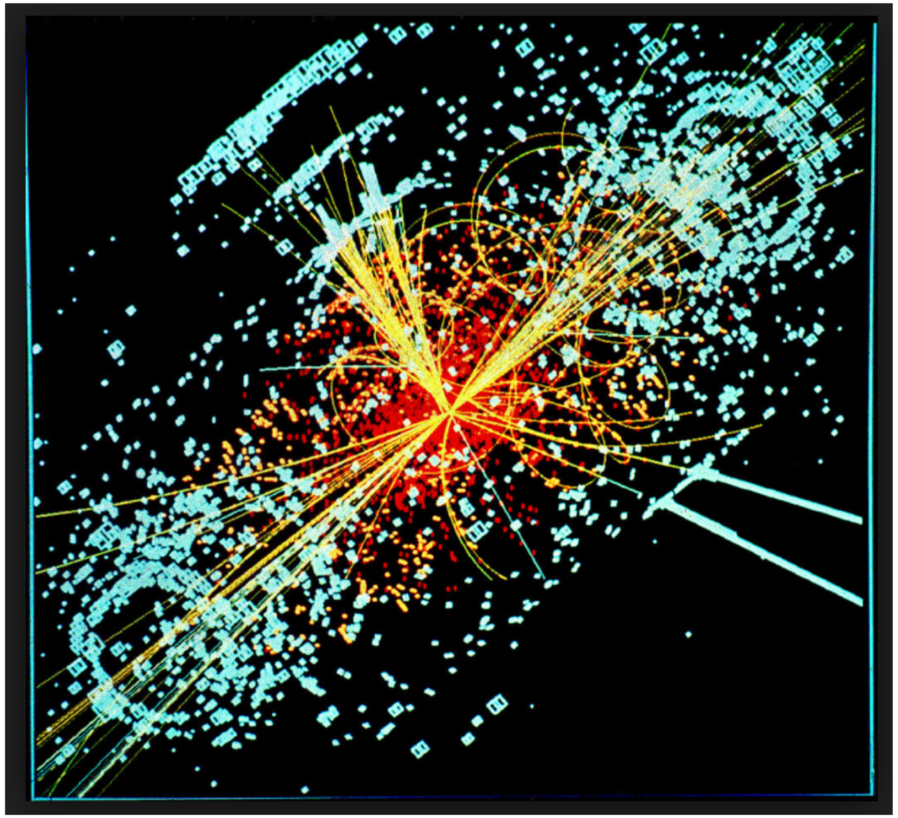



# Processing Stages - Reconstruction



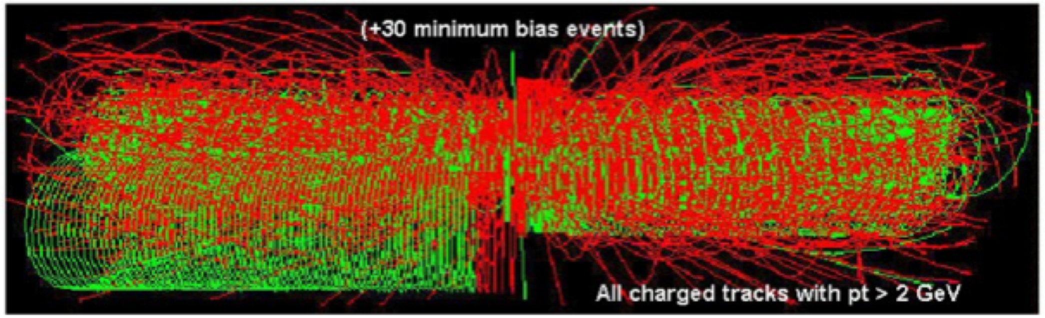
# What is reconstruction

- Tracker ‘hits’ form a puzzle
  - Which tracks created them?
- Each energy deposition is a clue
  - There are thousands of measurements in each snap-shot
- The experiment’s reconstruction must obtain a solution!
  - In well measured magnetic field
  - Matches the traces to tracks

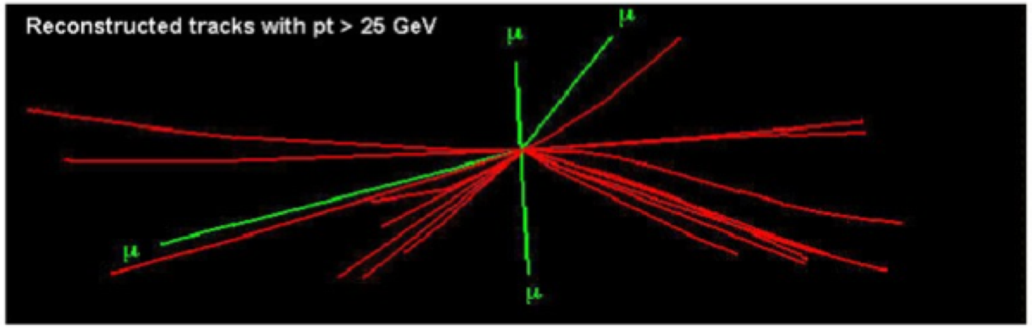


# The Reconstruction challenge

**Starting from this event**




**Looking for this "signature"**

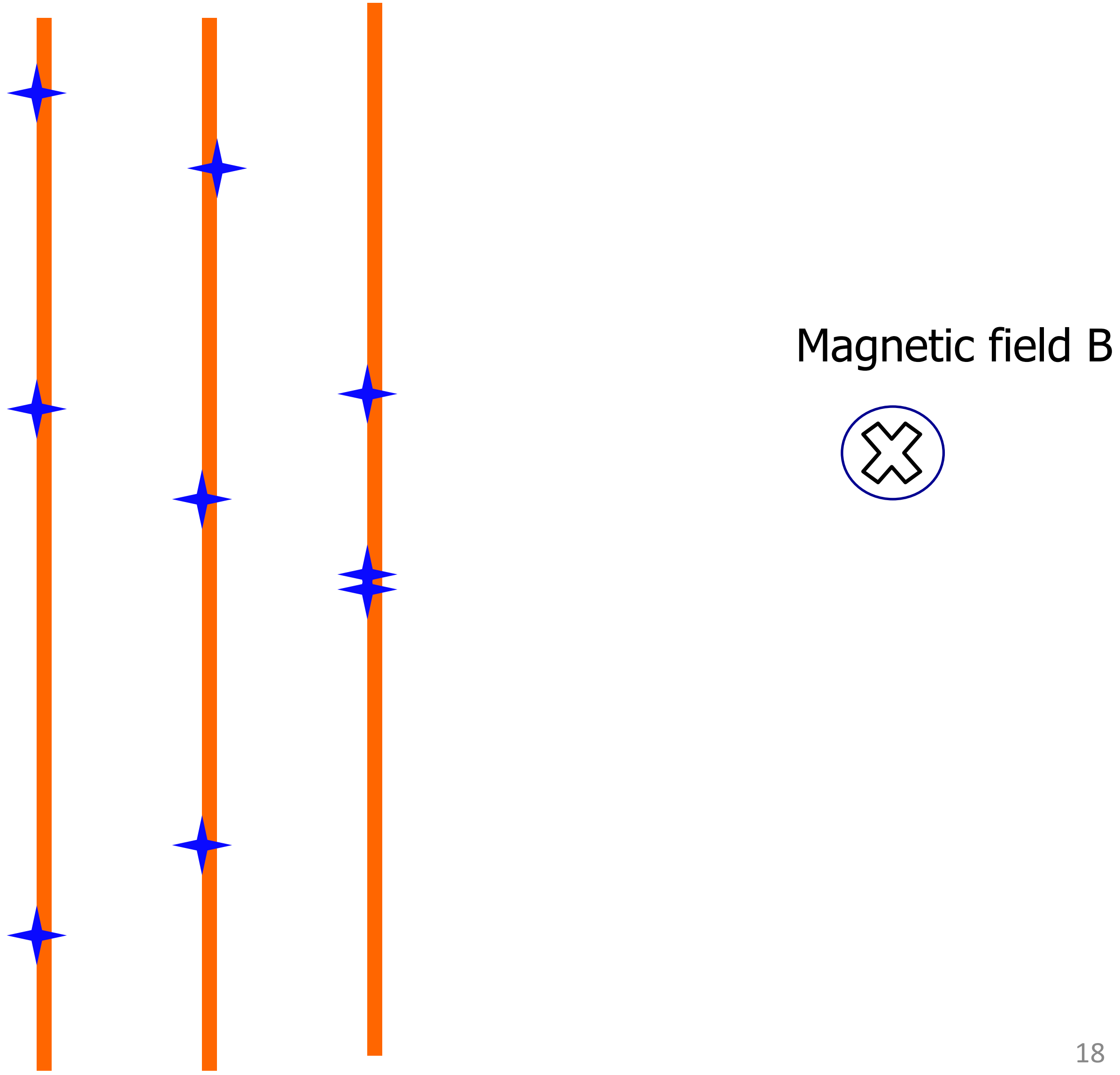



→ **Selectivity: 1 in $10^{13}$**

(Like looking for a needle in 20 million haystacks)

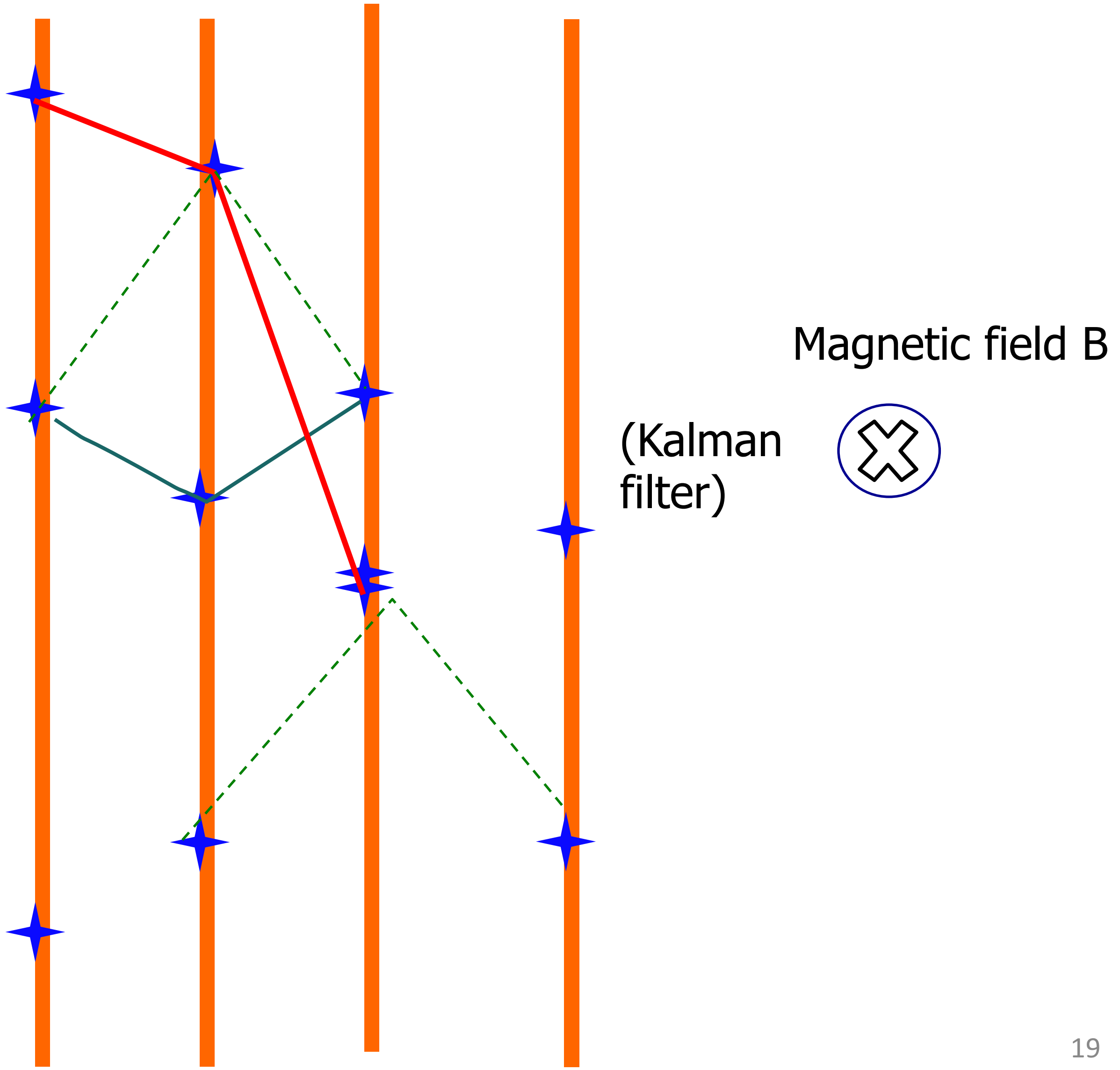
# How it works – a simple example

- Combine space points on first three planes (seeds)



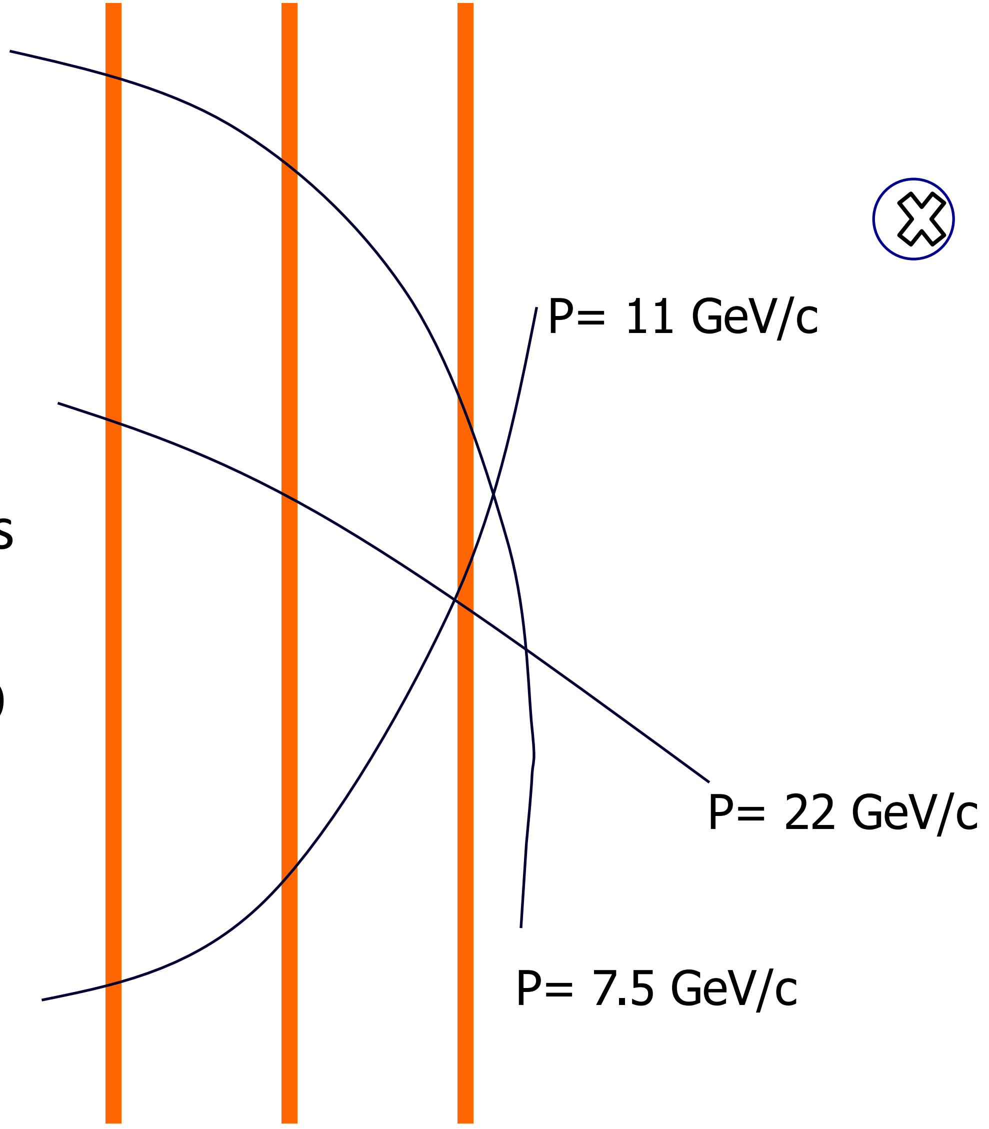
# How it works – a simple example

- Combine space points on first three planes (seeds)
- Prolongate to subsequent planes
  - Calculate differences between measured points and predictions



# How it works – a simple example

- Combine space points on first three planes (seeds)
- Prolongate to subsequent planes
  - Calculate differences between measured points and predictions
- Do track fitting, using PID hypothesis
  - Finally the track candidates are identified



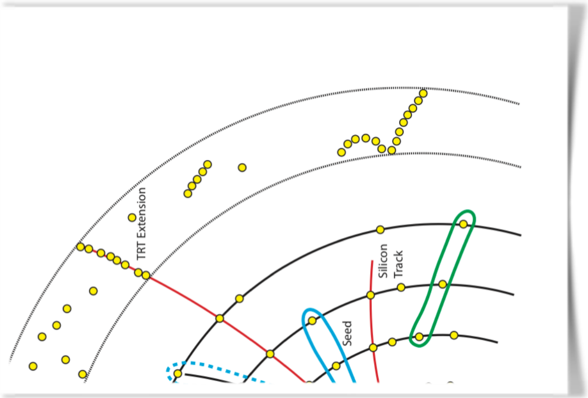
# ATLAS reconstruction procedure

**Inside-out**

- Space point formation
- Seed finding
- Track candidates (Combinatorial Kalman Filter)
- Ambiguity Solving
- TRT Extension

**Outside-in**

- TRT Segment Finder
- Track candidates (Combinatorial Kalman Filter)
- Ambiguity Solving

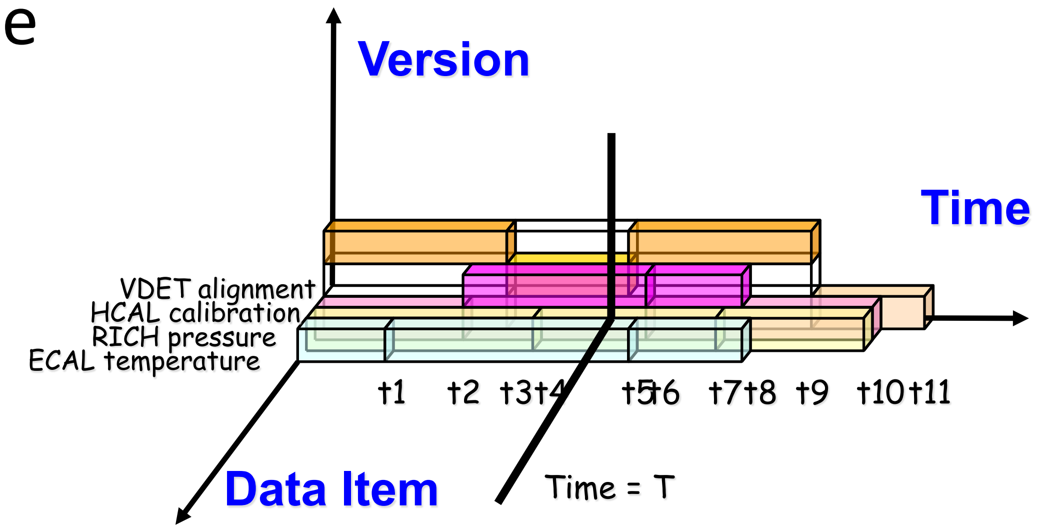



**Special Algorithms**

- Pixel Tracklets
- Large impact parameter tracks
- Electron tracking

# Detector conditions data

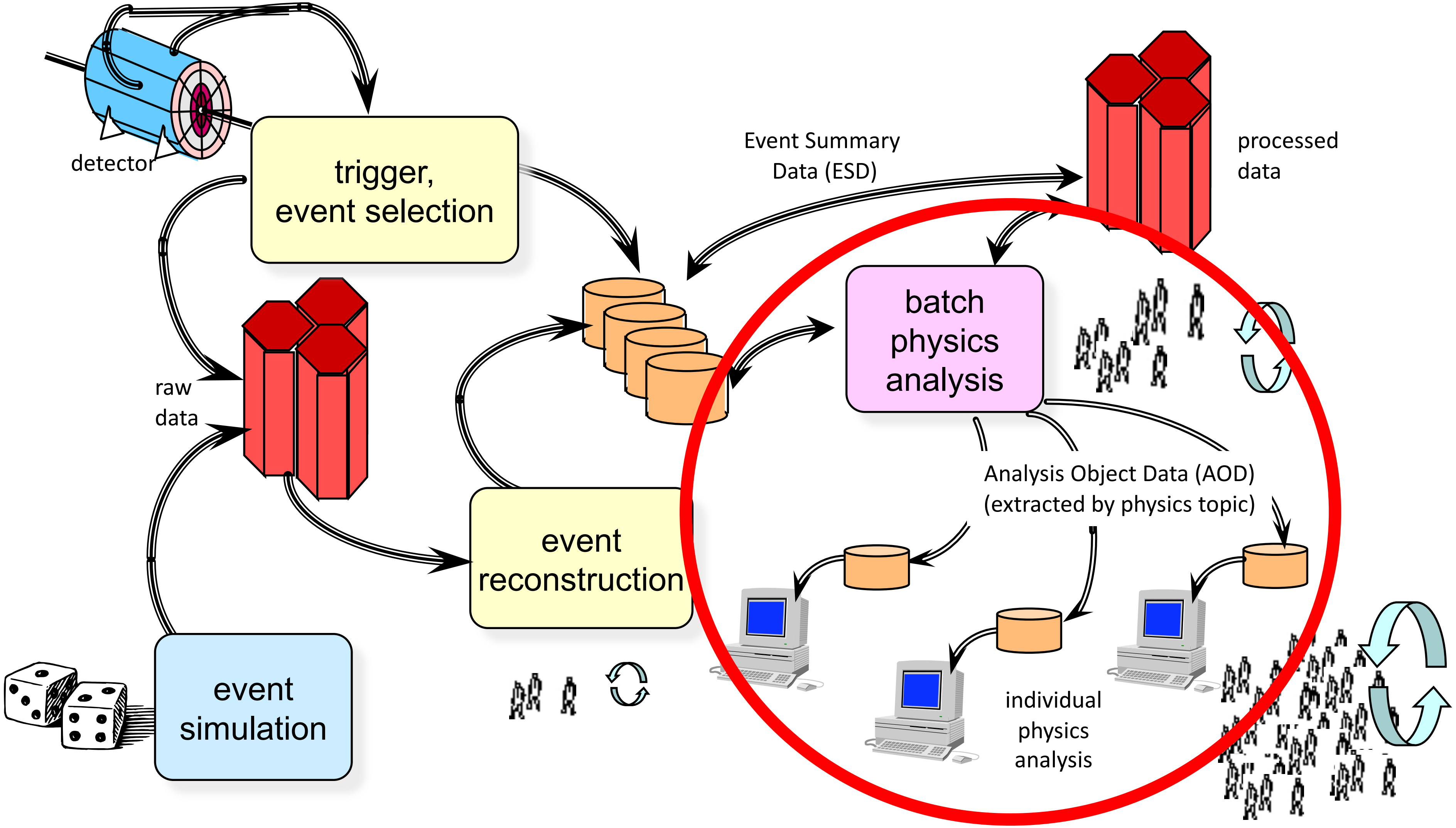
- Reflects changes in state of the detector with time
- Event Data cannot be reconstructed or analyzed without it
- Versioning
- Tagging
- Ability to extract slices of data required to run with job
- Long life-time



# Online and offline reconstruction

- Are collisions first-tagged really interesting enough to keep (given capacity constraints)?
  - Online reconstruction – seek to reconstruct 'as much as you can' quickly to enable decision
- Critical part of experiment – collisions which are not recorded are lost
- Later there is more time to reconstruct the contents of a collision – but this is also complex

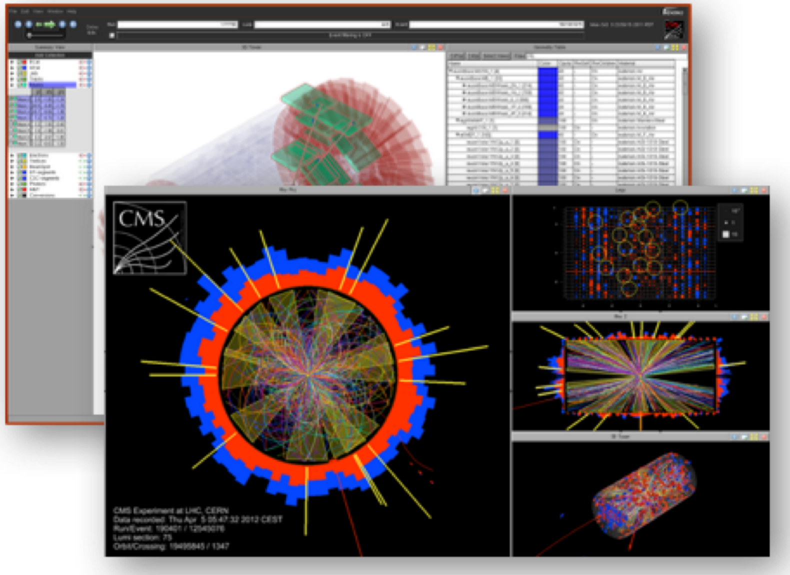
# Processing Stages - Analysis



# Data analysis

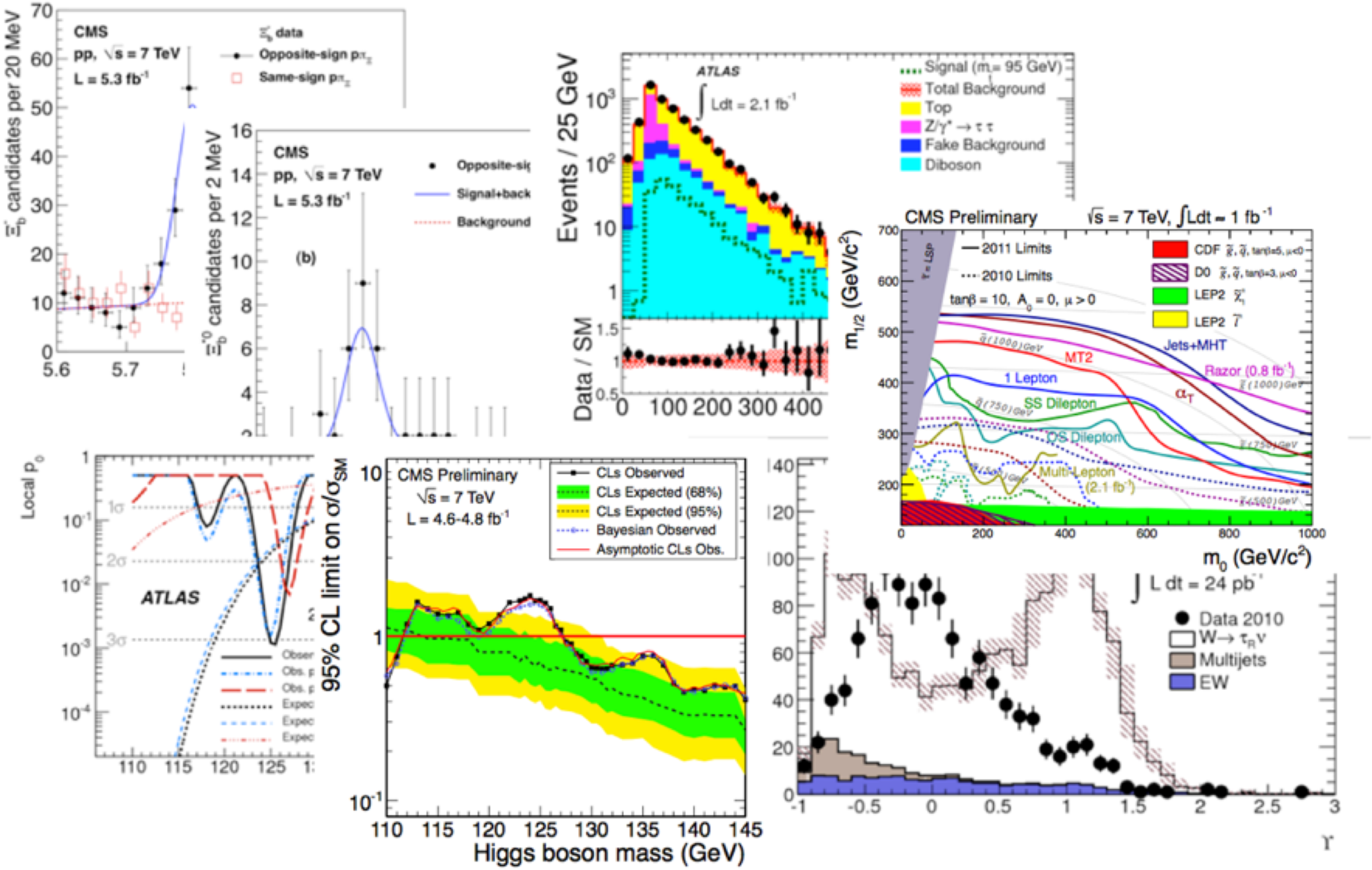
- Uses the results of Reconstruction
  - The products are reconstructed tracks, energy deposits (calorimeters)
  - Hierarchy of data from original (RAW), to summary (AOD)
- Extract observables from data (e.g. invariant mass, particle correlations, …)
  - Understand errors and features, by comparing with simulation
  - Compare with physics hypothesis, theory predictions, explore new physics
  - Programmed mainly in C++ & Python
- An experiment’s physics teams use the (large) pool of data
  - No longer in one central location, but in multiple locations (cost, space of building, computers, disks, network) .... using the GRID

# ROOT

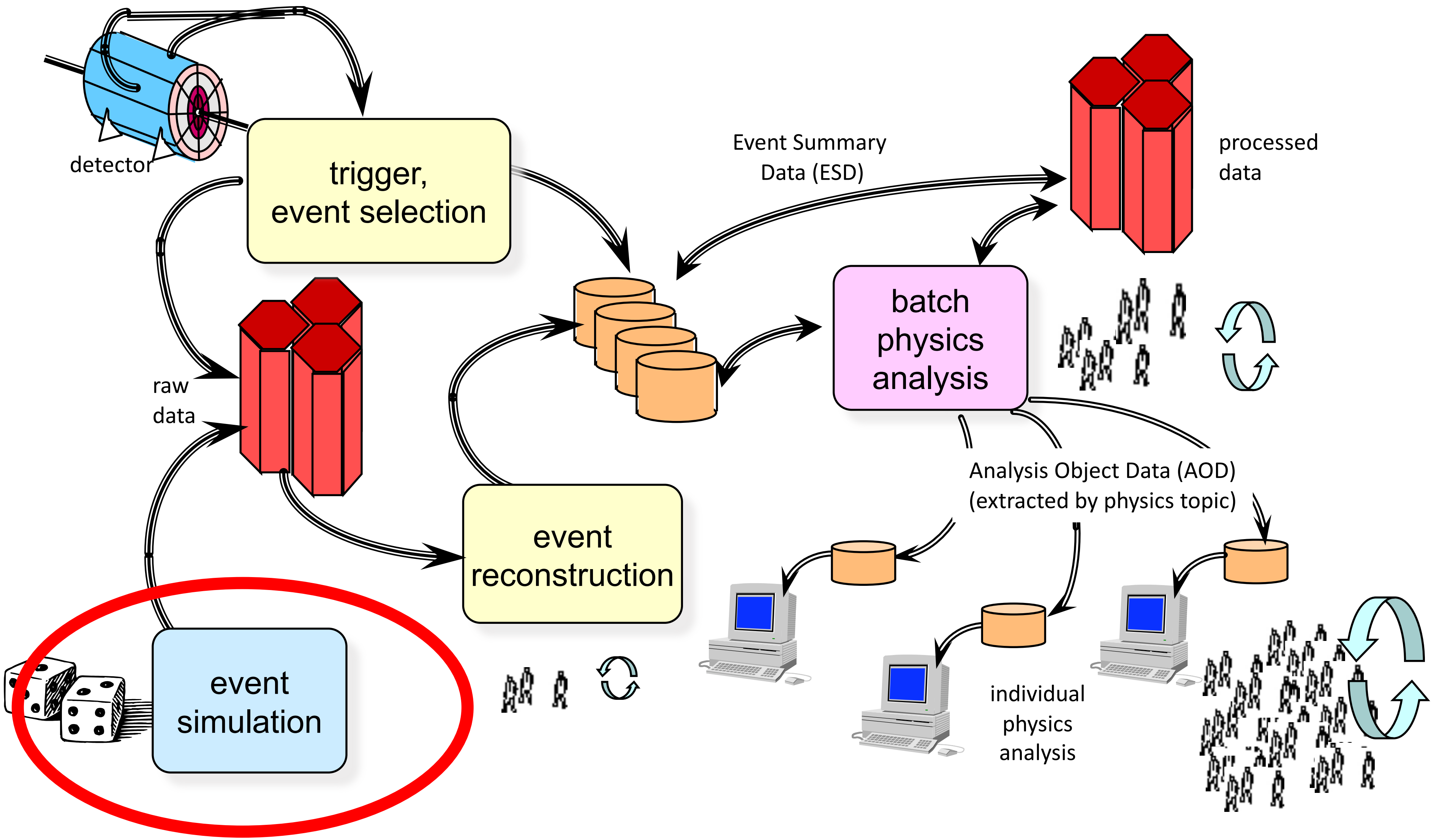



- “At the root of the experiments”, project started in 1995
- Open Source project (LGPL3)
  - mainly written in C++; 4 MLOC
- ROOT provides (amongst other things):
  - Interactive C++ interpreter (on top of LLVM and Clang)
  - Efficient data storage mechanism; 177 PB LHC data stored in ROOT (2015, now about 500 PB)
  - High-level interface for analysis in C++ and Python (RDataFrame)
  - Advanced statistical analysis algorithms
    - histogramming, fitting, minimization, statistical methods …
  - Scientific visualization: 2D/3D graphics, PDF, Latex
  - Geometrical modeler

# ROOT in plots



# Processing Stages - Simulation

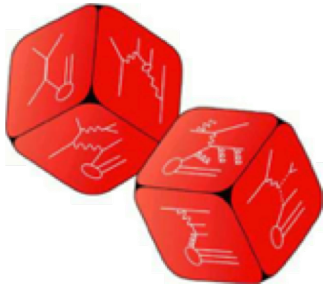


# What is simulation?

- Simulation = doing ‘virtual’ experiment
- Take all the known physics
- Start from your ‘initial condition’ (two protons colliding)
- Calculate the ‘final state’ of your detector to get the ‘experimental’ results
  - Solve equations of motion, detector electronics response, etc
- IMPOSSIBLE to be done analytically

# Monte Carlo simulation

- What is Monte Carlo?
  - Throwing random numbers
    - to calculate integrals
    - to pick among possible choices
- Why Monte Carlo?
  - complexity of the problem
  - lack of analytical description
  - need of randomness like in nature
    - Quantum mechanics: amplitudes => probabilities
      - Noting is certain, but anything that possibly can happen, will!
      - Want to generate events in as much detail as possible
      - get average and fluctuations right
      - make random choices, ~as in nature

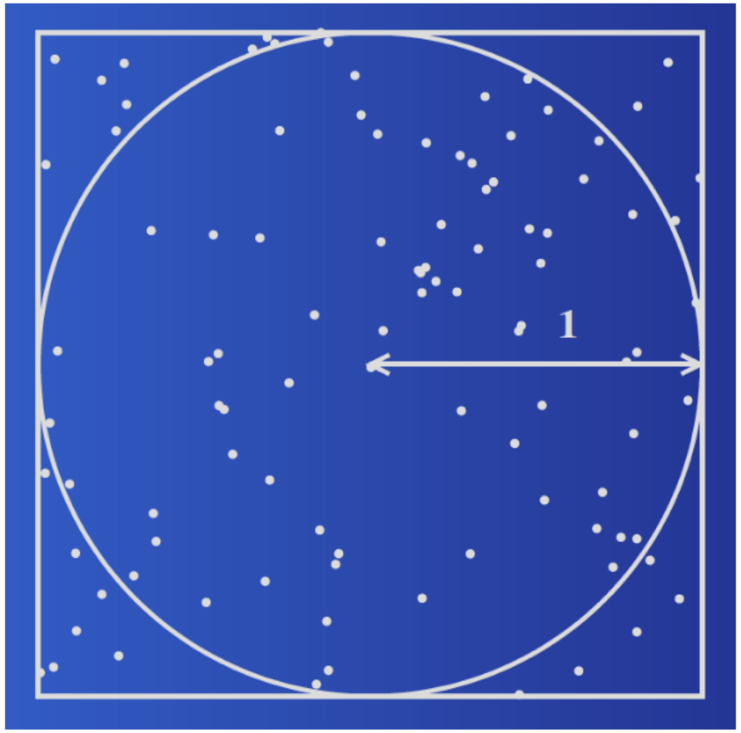


# Laplace method of calculating π (1886)

- Area of the square = 4
- Area of the circle = $\pi$
- Probability of random points inside the circle = $\pi / 4$

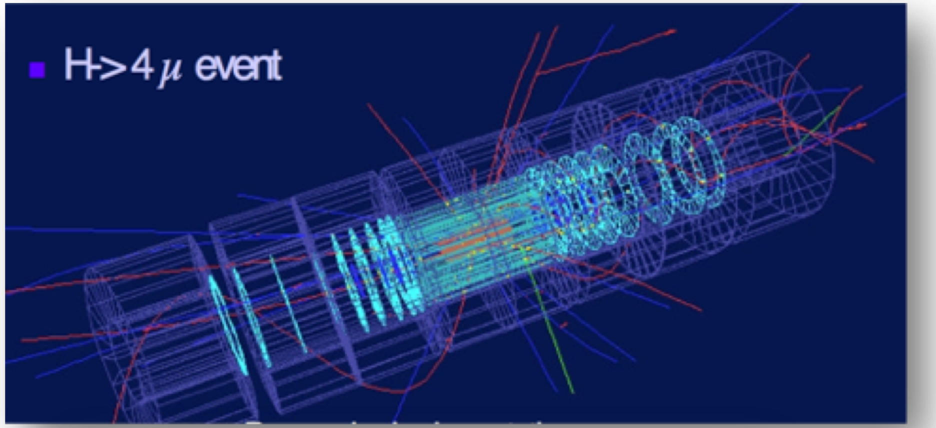
- Random points : $N$
- Random points inside circle : $N_c$

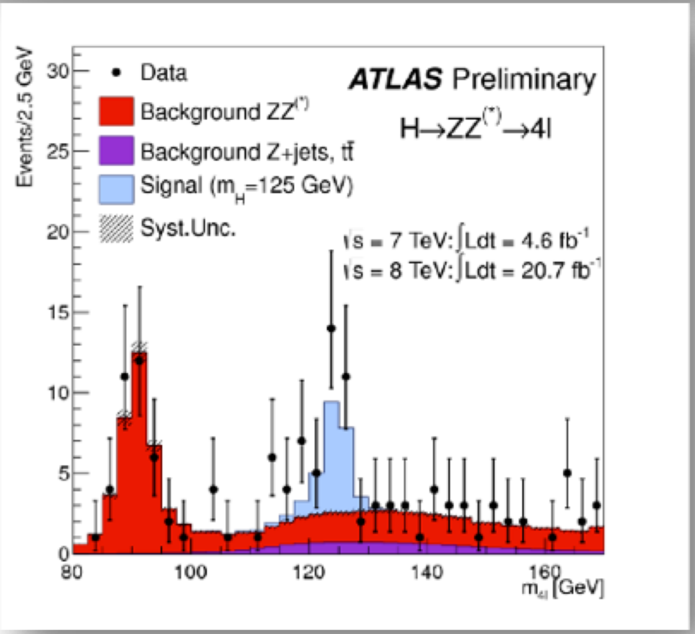
$$\pi \sim 4\, N_c / N$$



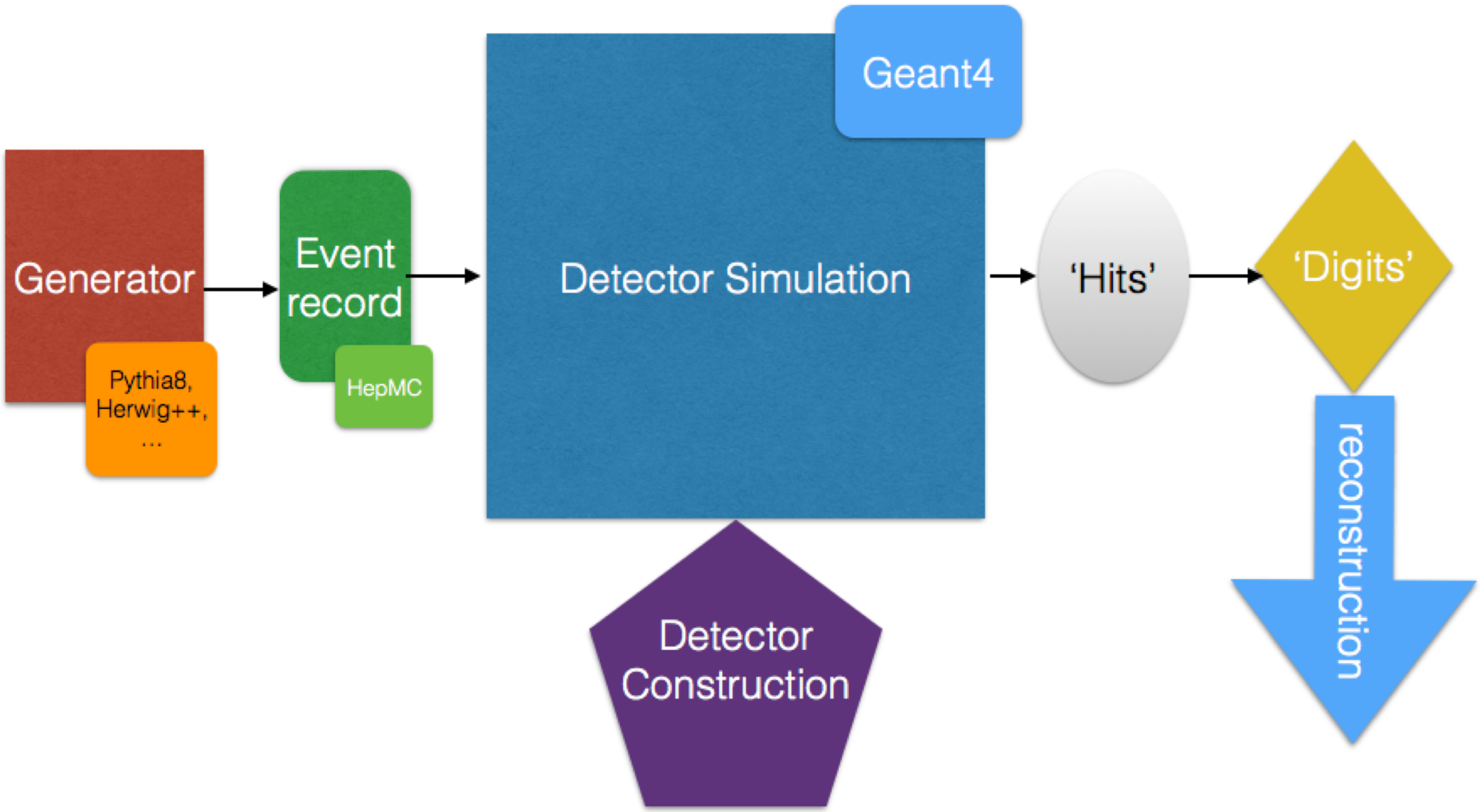
# Why do we need simulation?

- To design the apparatus (detector) to fulfill its role
- To prepare the reconstruction and analysis of results
  - Training on 'known' (simulated) events (MC 'truth')
- To understand the results
  - We need to know what to expect to
    - Verify existing models
    - Find new physics
  - Understand systematic errors



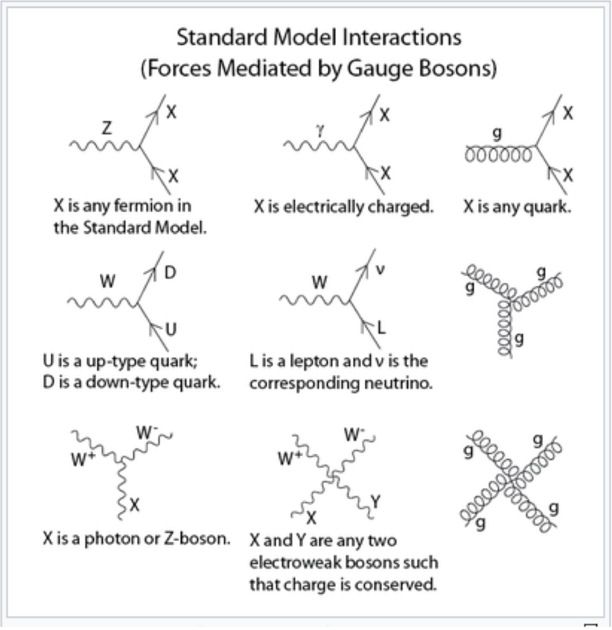



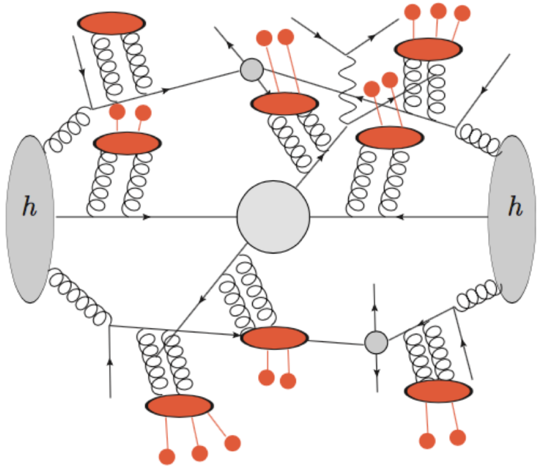
# Simulation chain for HEP experiments



# Monte Carlo generators

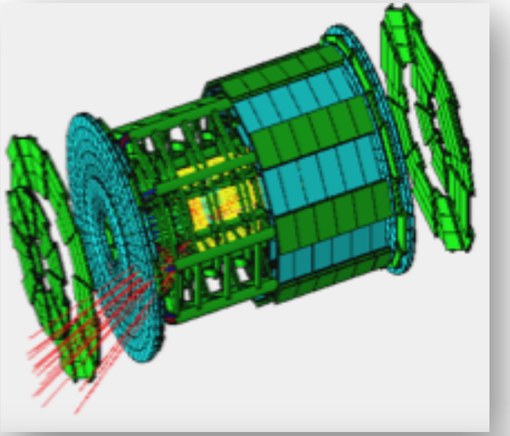
- Simulate particles reaction in vacuum
  - knows nothing about the surrounding detector
- All Standard Model processes are included
- No propagation of particles, just generation of the products of the ‘primary’ collision
- The output of the ‘generators’ is the input to the ‘transport’ code

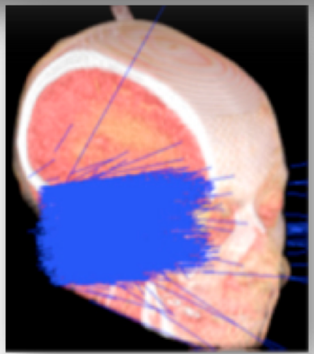


# Transport Code: Geant4

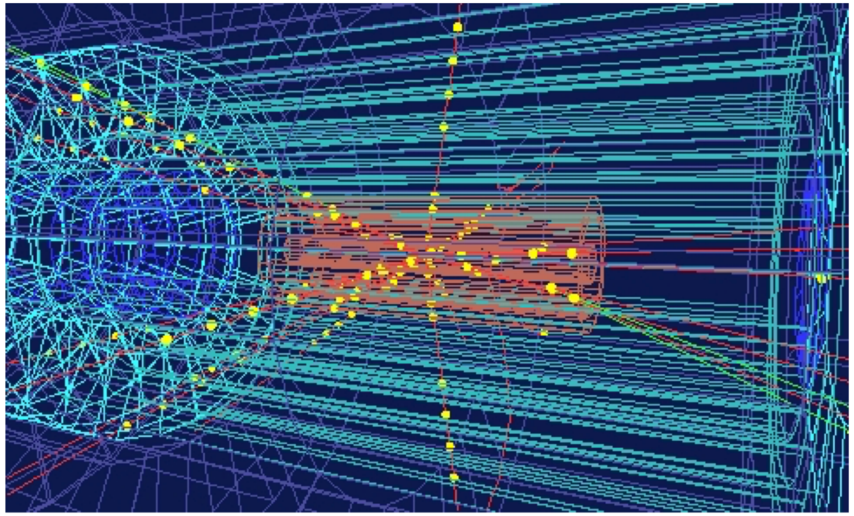
- Geant4 is a toolkit (C++) for the simulation of the passage of particles through matter.
- Its areas of application include high energy, nuclear and accelerator physics, as well as studies in medical and space science
- In HEP has been successfully employed for
  - Detector design
  - Calibration/alignment
  - Data analysis



# What does Geant4 do?

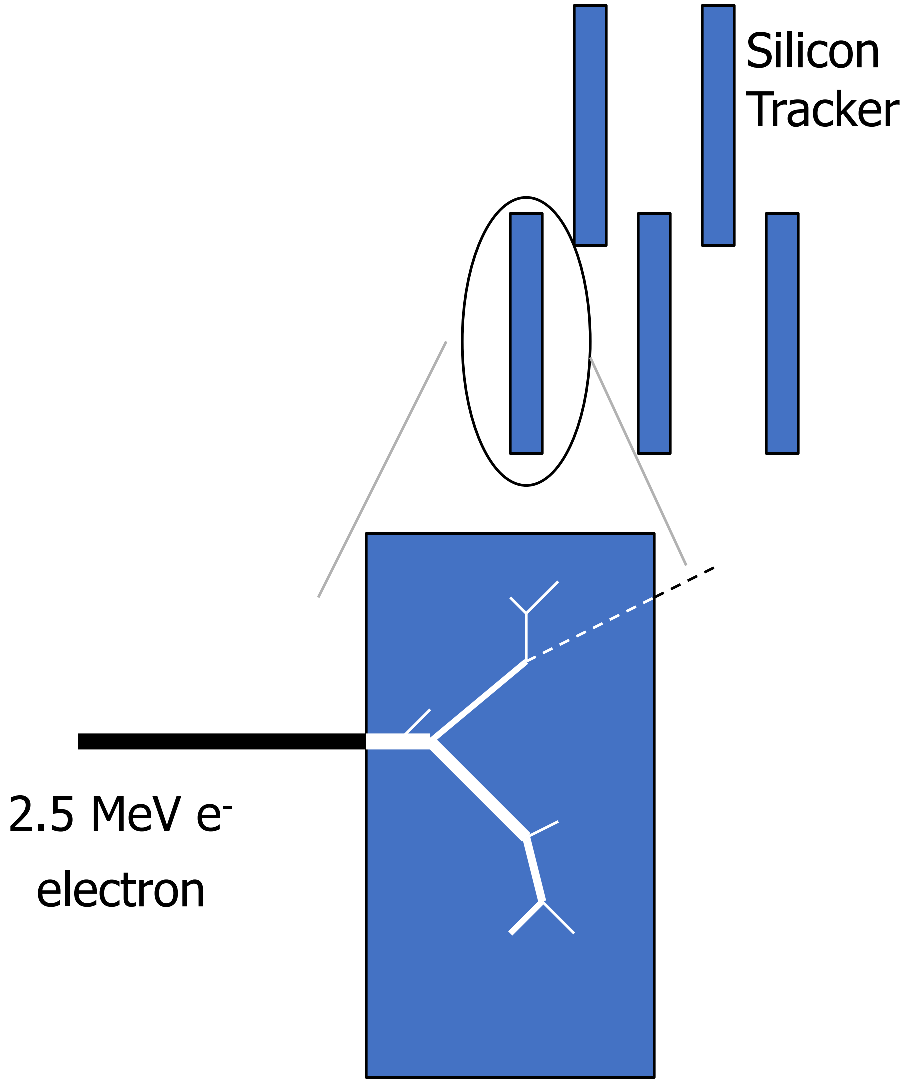
- ‘propagates’ particles through geometrical structures of materials, including magnetic field
- simulates processes the particles undergo
  - creates secondary particles
  - decays particles
- calculates the deposited energy along the trajectories and allows to store the information for further processing (‘hits’)



# Simulation ingredients

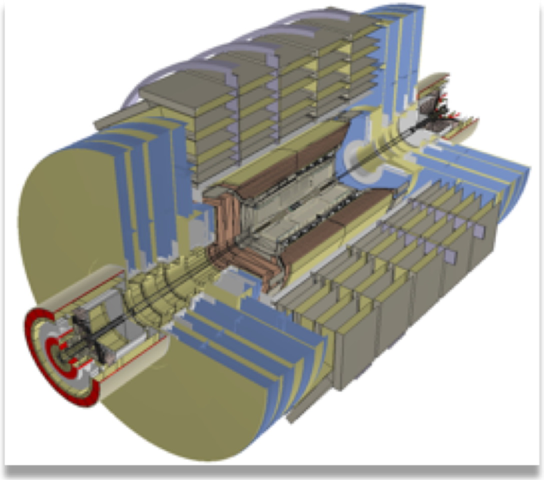
- We model
  - Detector's Geometry
    - Shape, Location, Material
  - Physics interactions
    - All known processes
      - Electromagnetic
      - Nuclear (strong)
      - Weak (decay
  - we 'shoot' particles and 'propagate' them through the modeled detector

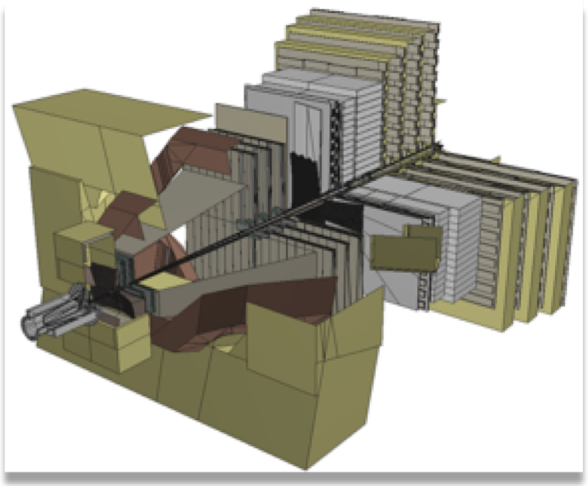
$$\sigma_{total} = \Sigma\, \sigma_{per\text{-}interaction}$$



# Geometry and Materials

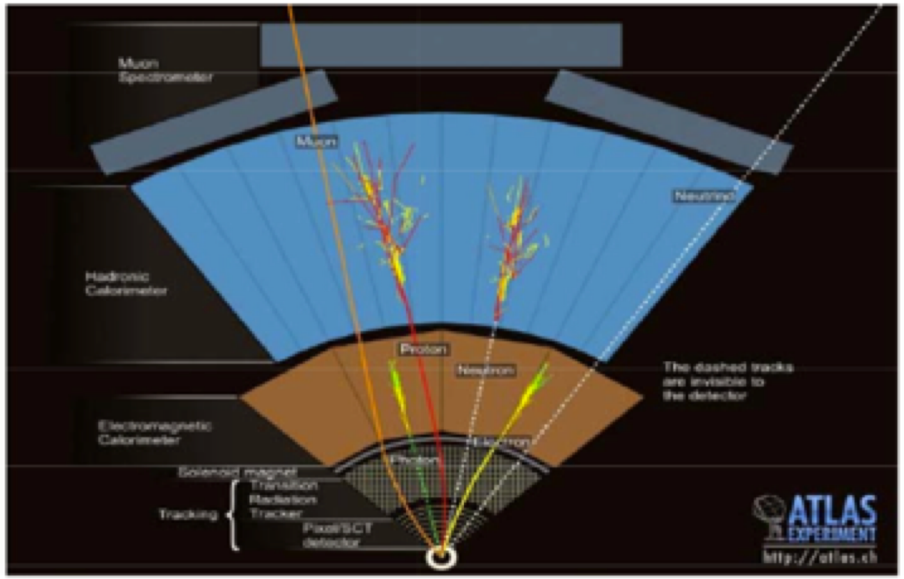
- How to implement (efficiently) this in your computer program?
- You need ‘bricks’
  - ‘solids’, ‘shapes’
  - you need to position them
  - you want to ‘reuse’ as much as possible the same ‘templates’
- Database of Materials
  - National Institute of Standards (NIST)
- Magnetic Fields
  - numerical integration of the equation of motion (Runge-Kutta method)





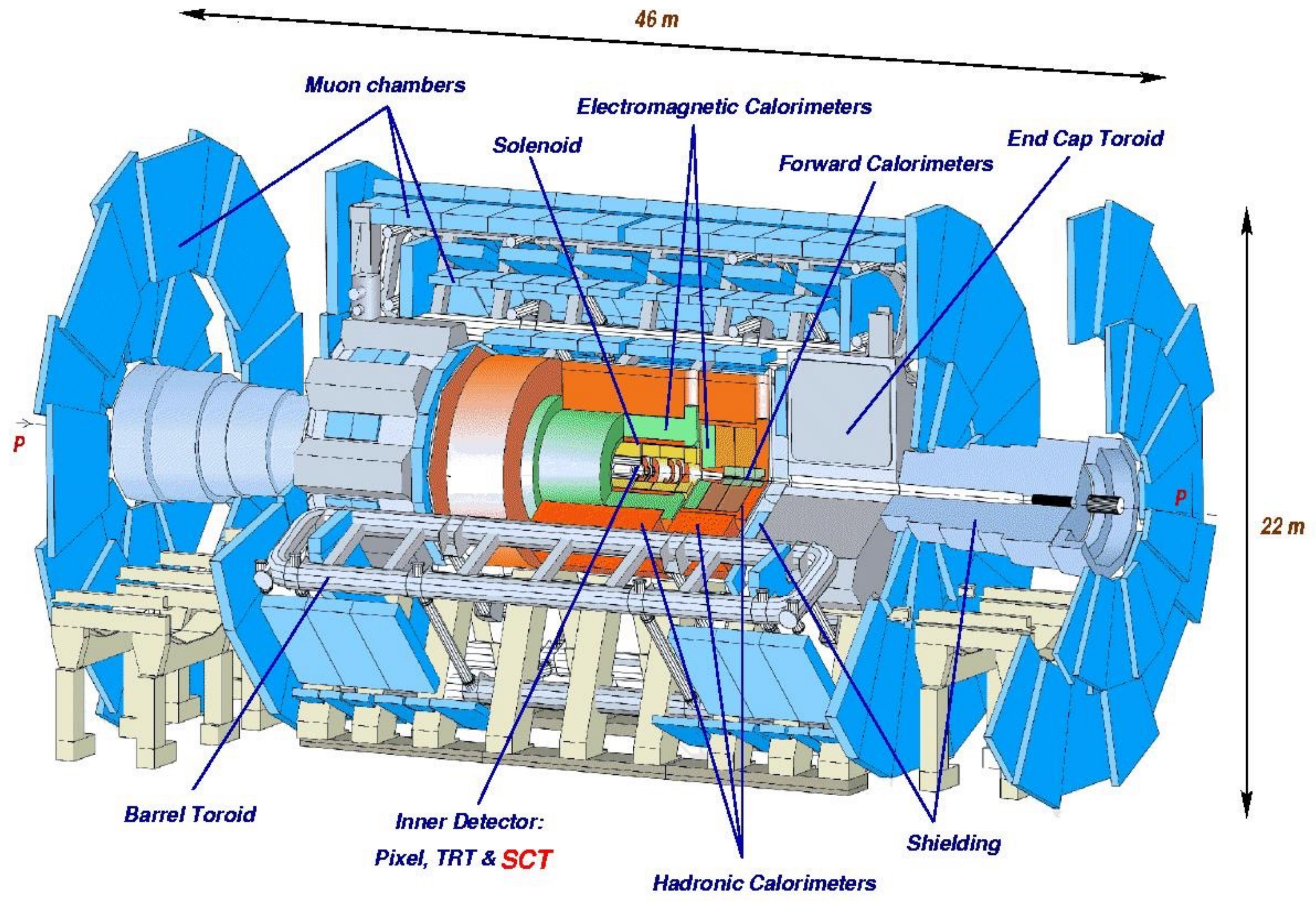
# Physics...

- What happens to particles in matter?
- We want to model the physics we know
  - each possible physics process provides the "interaction length" compared with distance to next geometrical boundary
    - the smallest wins
  - generating a "final state" and secondaries tracks
- Electromagnetic
  - gammas and charged particles
- Hadronic
  - neutrons, mesons (K,π), muons, ...



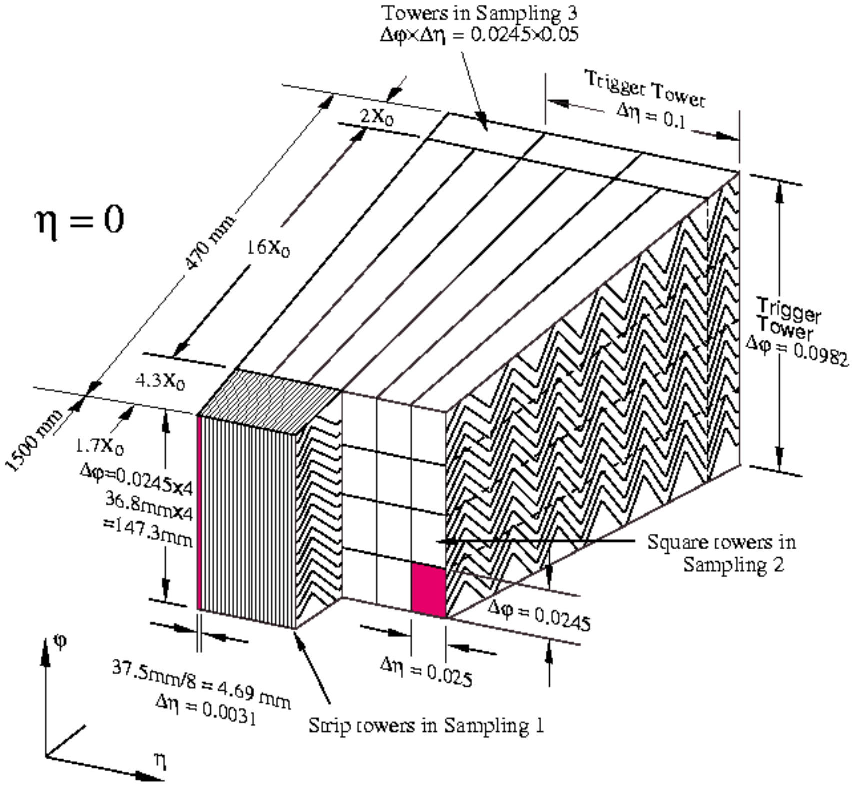

Because of the detailed geometries, the detailed physics and the required precision the simulation is very CPU hungry

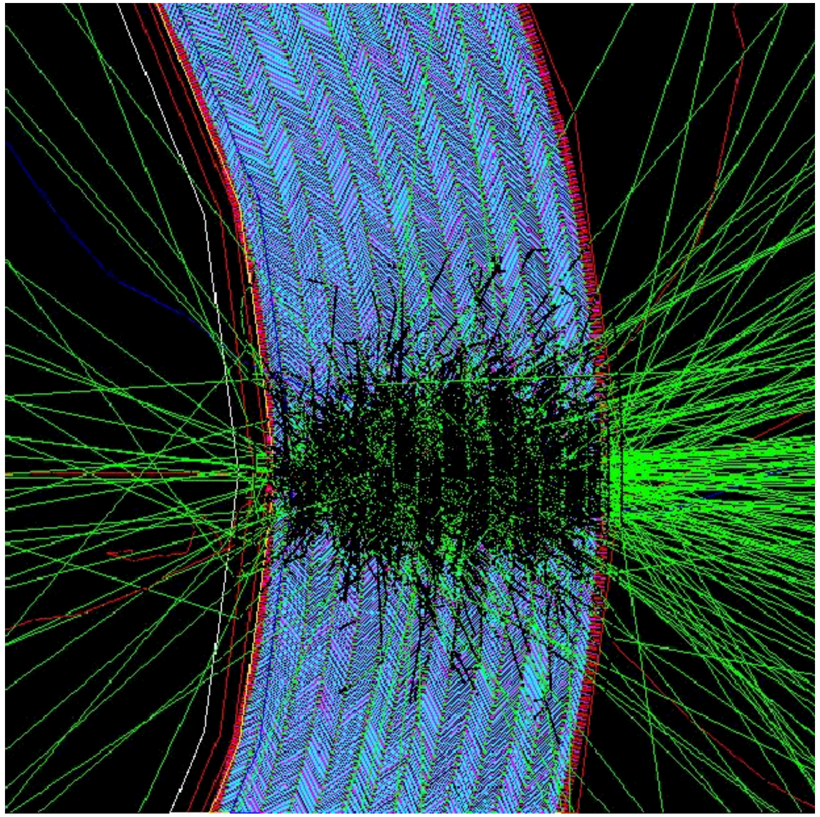
# ATLAS



46 m
Muon chambers
Electromagnetic Calorimeters
Solenoid
End Cap Toroid
Forward Calorimeters
P
P
22 m
Barrel Toroid
Inner Detector:
Pixel, TRT & SCT
Hadronic Calorimeters
Shielding

# ATLAS Calorimeter (a very, very small part of it)



. Kordas "Geant4 for the ATLAS EM calo" — CALOR2000, Annecy, 12 October2000 (4

# Summary: data rates

- Particle beams cross every 25 ns (40 MHz)
  - Up to 25 particle collisions per beam crossing (for Run2, higher for Run3)
  - Up to $10^9$ collisions per second
- Basically 2 event filter/trigger levels
  - Hardware trigger (e.g. FPGA)
  - Software trigger (PC farm)
  - Data processing starts at readout
  - Reducing $10^9$ p-p collisions per second to O(1000)
- Raw data to be stored permanently: >15 PB/year

| Physics Process | Events/s |
| --- | --- |
| Inelastic p-p scattering | $10^8$ |
| $b$ | $10^6$ |
| $W \to e\upsilon$ ; $W \to \mu\upsilon$ ; $W \to \tau\upsilon$ | 20 |
| $Z \to ee$ ; $Z \to \mu\mu$ ; $Z \to \tau\tau$ | 2 |
| $t$ | 1 |
| Higgs boson (all; $m_H$ = 120GeV) | 0.04 |
| Higgs boson (simple signatures) | 0.0003 |

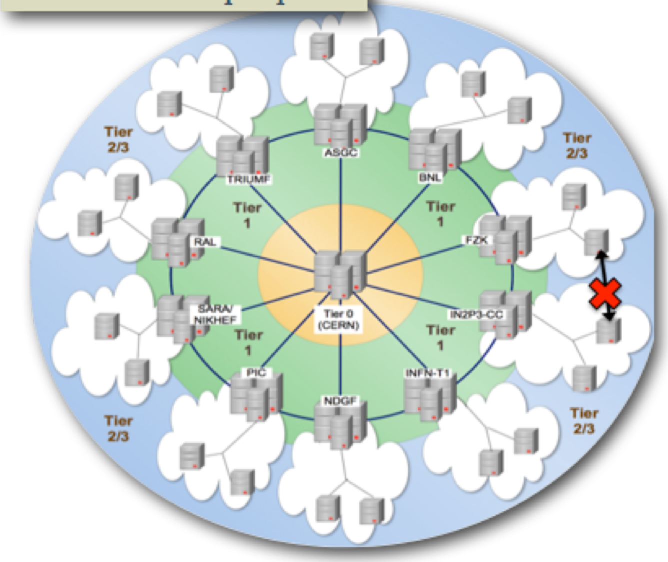
This is our Big Data problem!!

# Big Data requires Big Computing

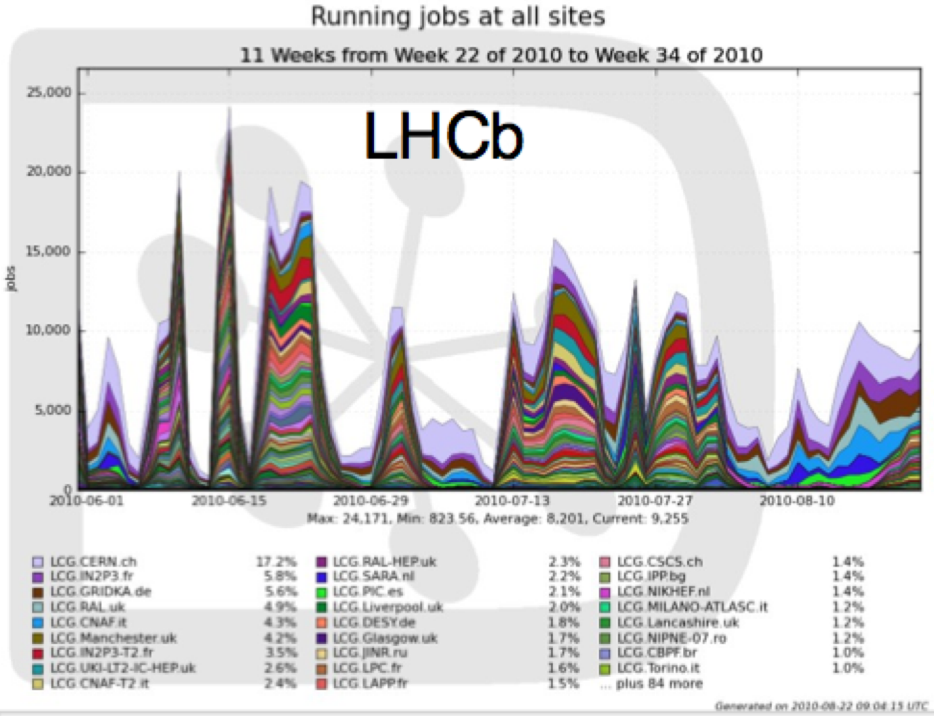
WLCG
Worldwide LHC Computing Grid

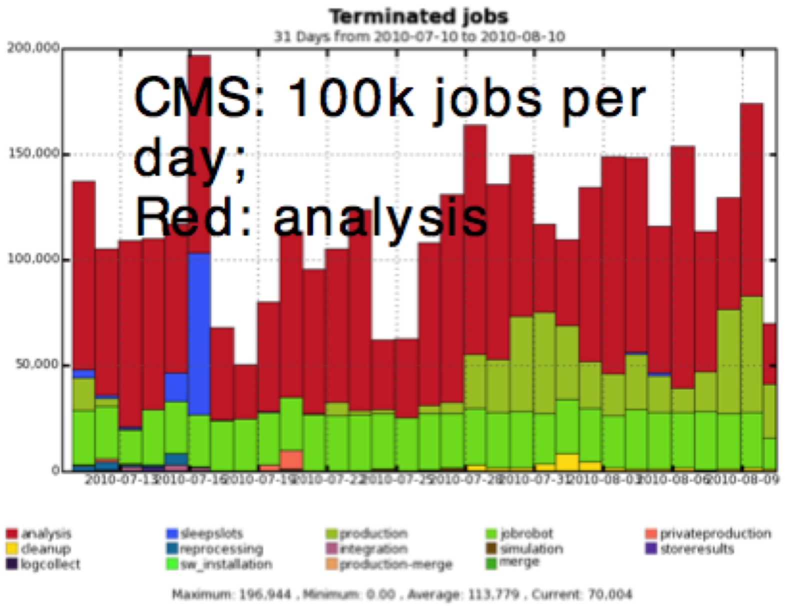
- The LHC experiments rely on distributed computing resources:
  - WLCG - a global solution, based on the Grid technologies/middleware.
    - distributing the data for processing, user access, local analysis facilities etc.
    - at time of inception envisaged as the seed for global adoption of the technologies
- Tiered structure
  - Tier-0 at CERN: the central facility for data processing and archival
  - 11 Tier-1s: big computing centers with high quality of service used for most complex/intensive processing operations and archival
  - ~140 Tier-2s: computing centers across the world used primarily for data analysis and simulation.
- So far computing was not a limiting factor for the Physics program of the LHC experiments

Capacity:
~350,000 CPU cores
~200 PB of disk space
~200 PB of tape space



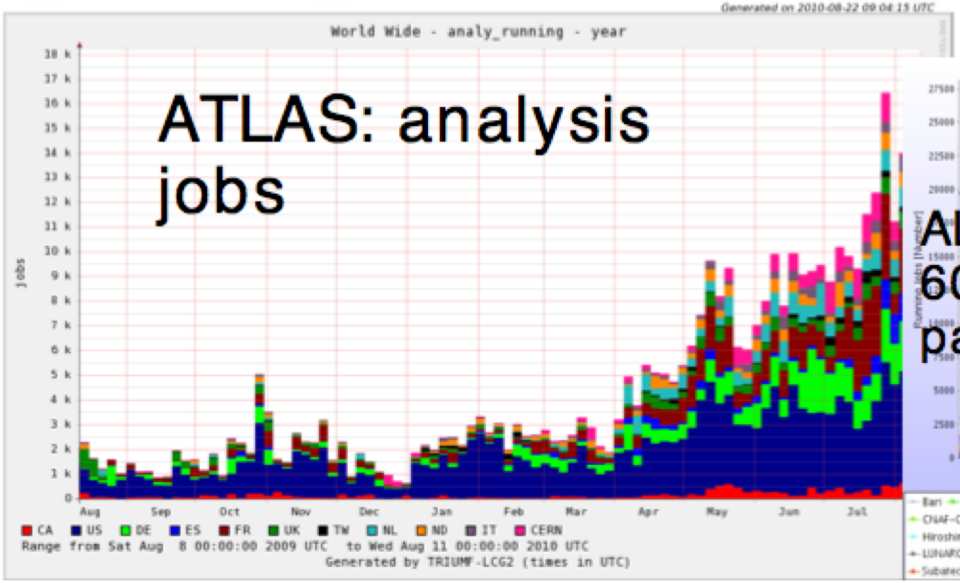
# Running jobs on LCG

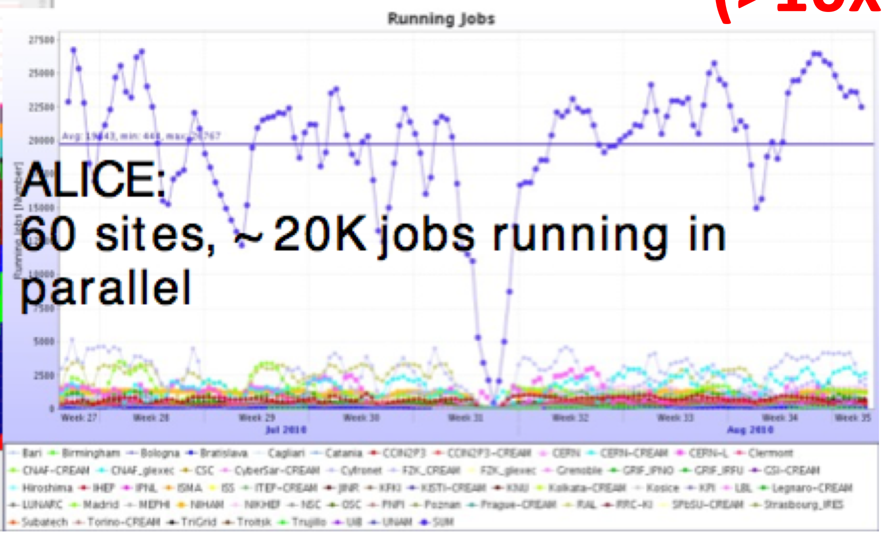


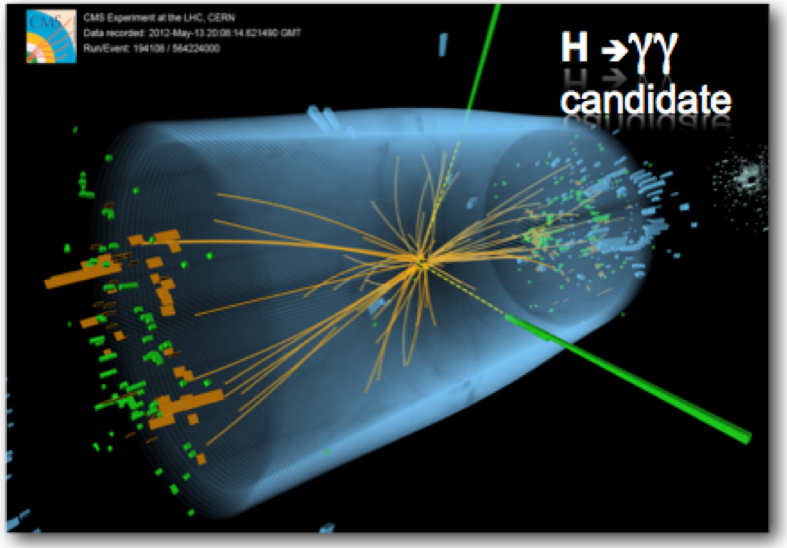

2010

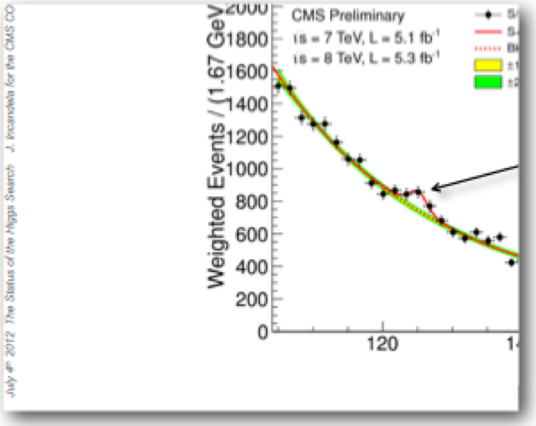
(>10x today)

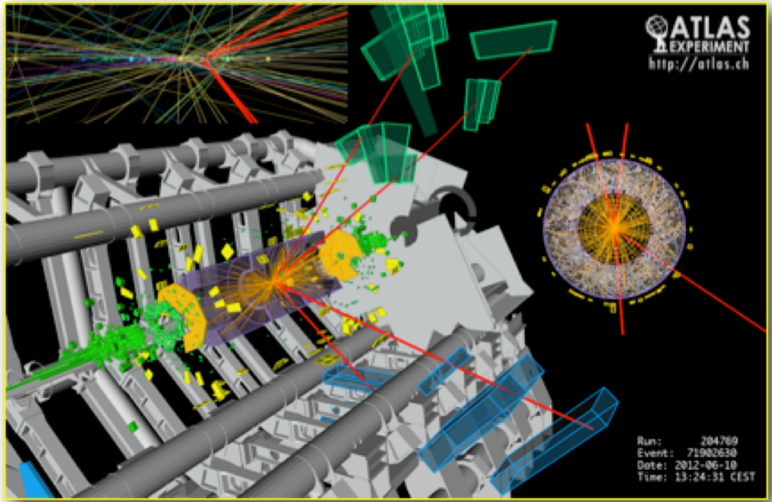


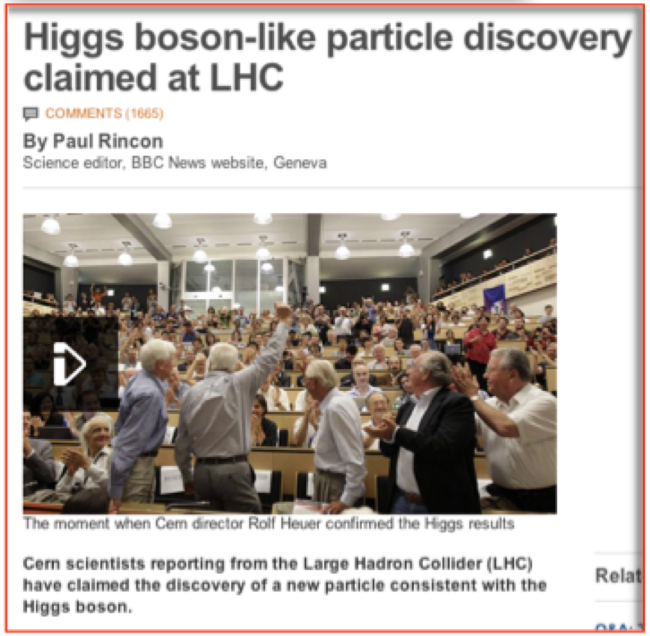



# A Success Story!



## Higgs boson-like particle discovery claimed at LHC

COMMENTS (1665)

By Paul Rincon
Science editor, BBC News website, Geneva

The moment when Cern director Rolf Heuer confirmed the Higgs results

**Cern scientists reporting from the Large Hadron Collider (LHC) have claimed the discovery of a new particle consistent with the Higgs boson.**

# Challenges for HEP Software

- High-luminosity LHC will produce 7x-10x today's event rate
  - More precise Higgs physics (5x), rare signals, new physics
  - Timescale: 2017-2018
  - Constant computing budget
  - Technology evolves, but we need to be able to make use of it
    - Massive parallelism, AI, hybrid computing, …
- Huge pressure for both experiment software systems and common software
  - Important R&D ongoing for experiment upgrades
    - Hardware and software
  - R&D for the common simulation tools

# Conclusion

- Modern HEP experiments would be impossible without computing
    - Online triggering and selection
    - Offline reconstruction, analysis and simulation
- Huge data volumes
- Distributed processing