ACADEMIC TRAINING

From Raw Data to Physics Paper

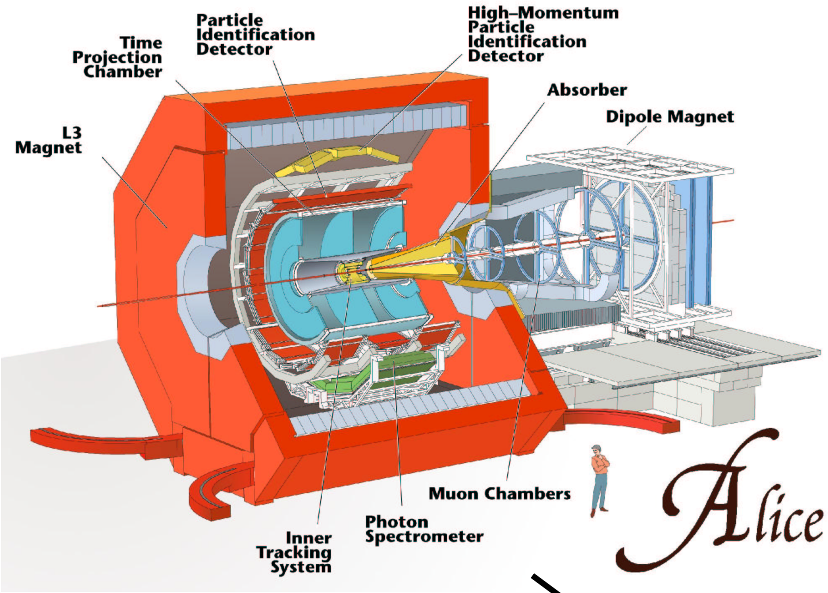
B. Panzer – CERN/IT,
F. Rademakers – CERN/EP,
P. Vande Vyvre - CERN/EP

Academic Training CERN

Introduction

- What is HEP data analysis
- Complexity of the problem
- Software development
- Data analysis frameworks
- Experiment frameworks
- Simulation
- Distributed computing and the GRID
- Experiment distributed computing solutions
- Conclusions

From Raw Data to Physics Paper



ALICE 2000-28
Internal Note / PHY
24 November 2000

Day One Proton-Proton Physics with the ALICE Central Detector

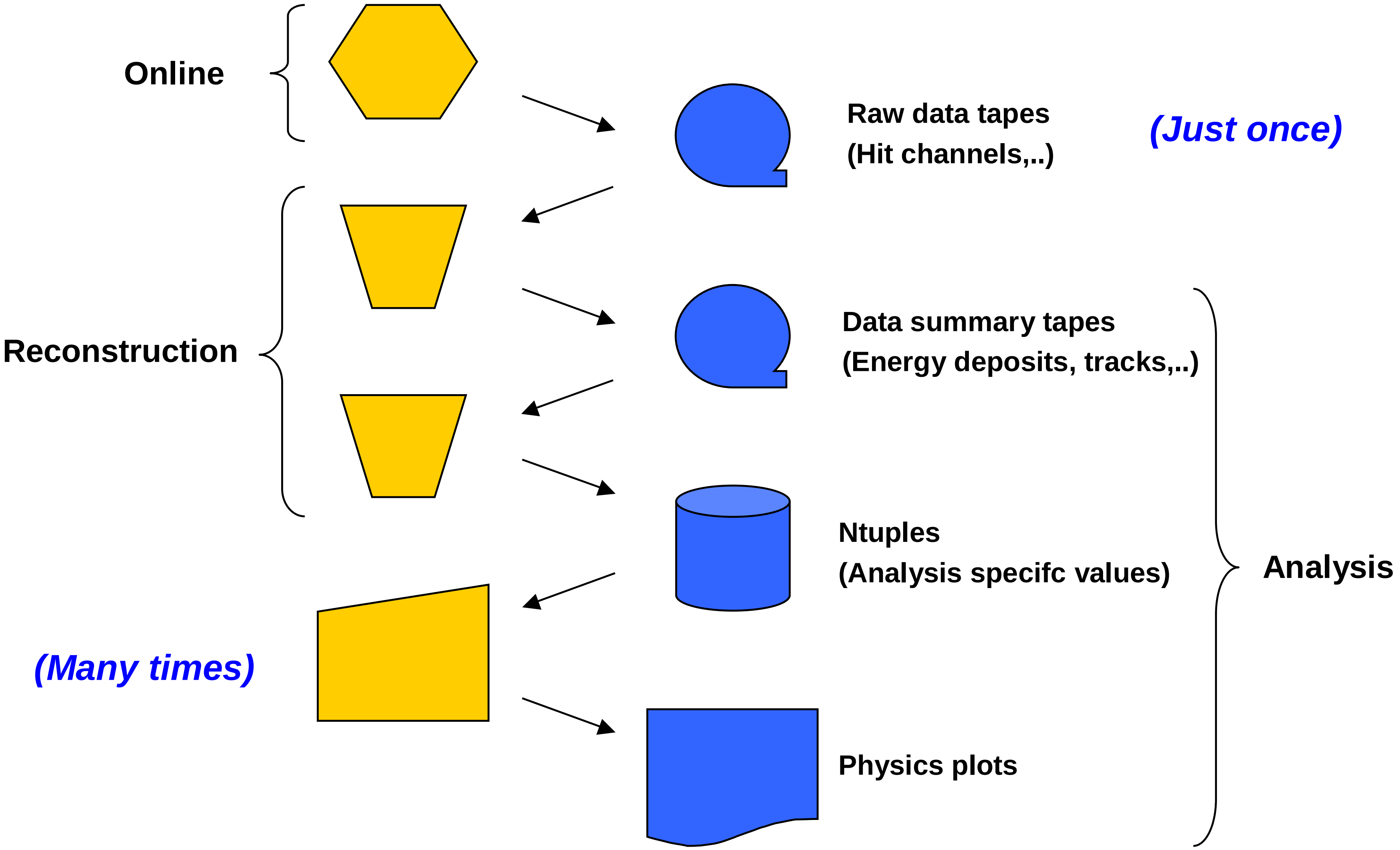
P. Giubellino, S. Kiselev, W. Klempt, A. Morsch, G. Paic, J.-P. Revol and K. Safarik

I. Introduction

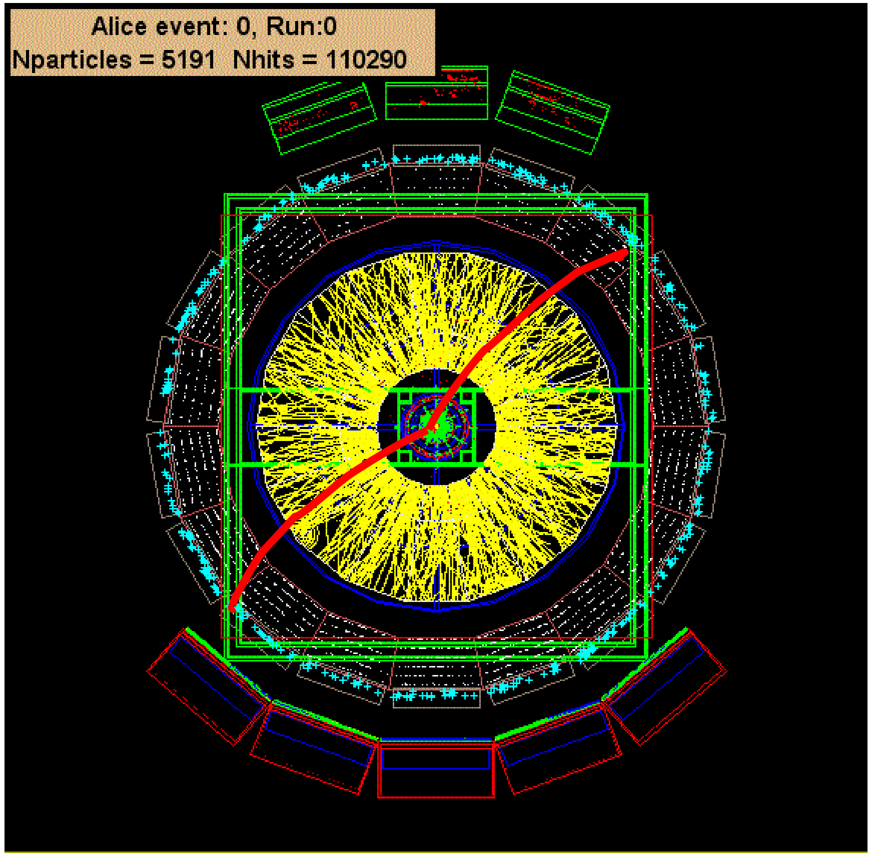
From the Technical Proposal onwards the proton-proton programme was considered an integral part of the ALICE experiment. At the present stage we feel it is important to review the scope of that programme with a specific focus, i.e. we want to discuss the possibilities that ALICE will have in the first few months of LHC running, to make efficient use of the LHC proton beams in order to:

1. provide first insights into pp physics in a new energy domain ($\sqrt{s}$ = 14 TeV) far higher than that available today ($\sqrt{s}$ = 2 TeV at the Tevatron), to study soft hadronic physics and its gradual evolution to the better understood perturbative QCD (pQCD) regime. In this respect it is useful to recall that the important contribution of UA1 to minimum bias physics came mainly from the central detector chamber, see [1–16]

How it is Done



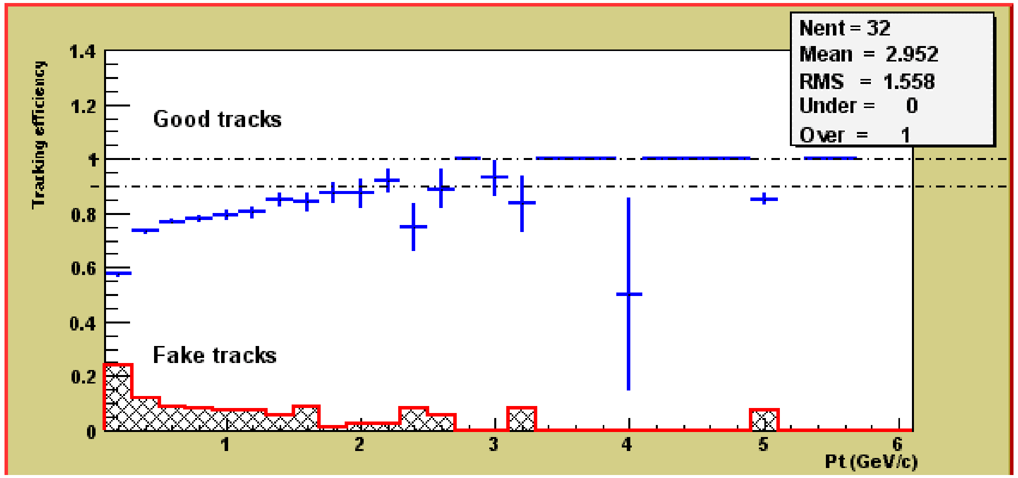
What Needs to be Done



- Calibrate signals
- Find patterns
- Fit for particle measurements
- Recognize underlying event features
- Extract physics quantities

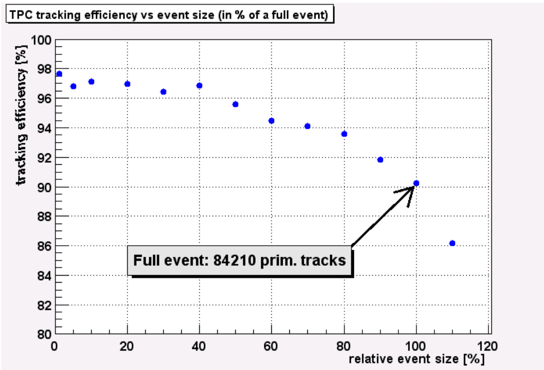
Now do this 10^7 times

What Needs to be Done



Use the event information to study physics questions, make a measurement, publish a paper

Need tools to compare, contrast, understand physics results



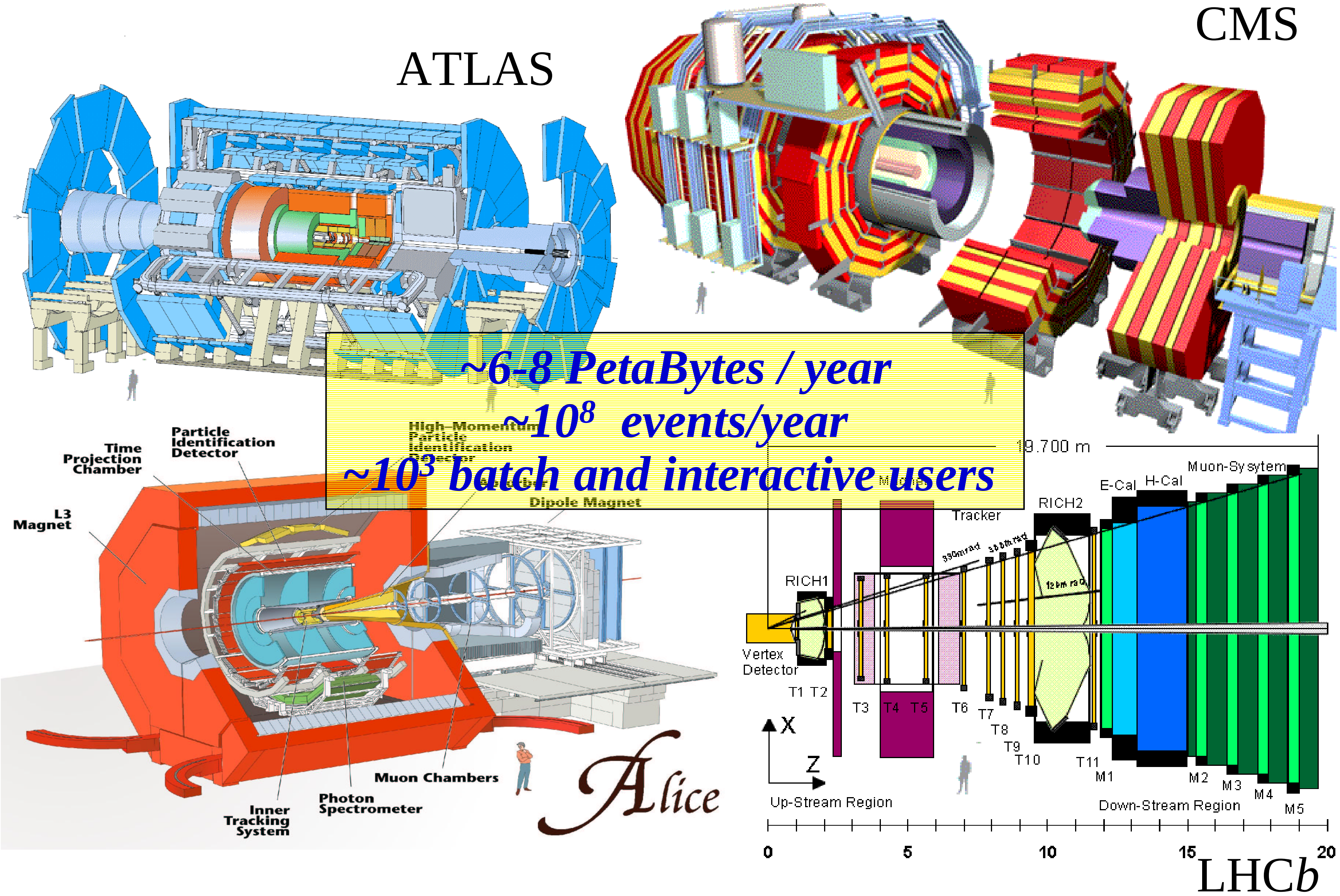
Varieties of Data Analysis

- Large scale production
  - Reconstruction, selections, forming distributions
- Develop and test new algorithms
  - Initially on small samples
- Monitor performance over time and space

- Study distributions in large number of events
- Study one event to death
- Compare measurements and theory

Complexity of the Problem

The LHC Detectors

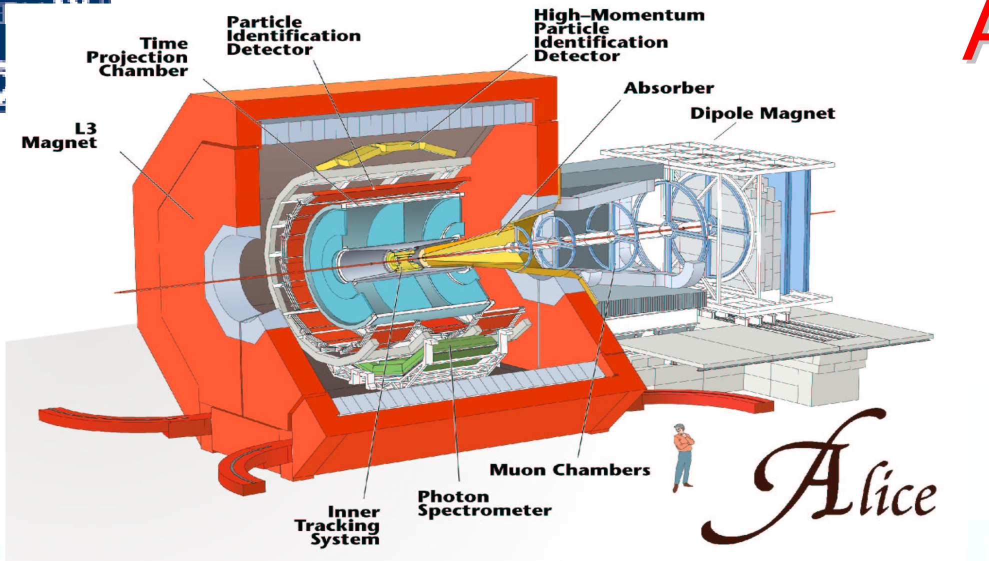


~6-8 PetaBytes / year

~10^8 events/year

~10^3 batch and interactive users

ALICE in Numbers



Independent detectors	**12**
Total weight	**10,000t**
Overall diameter	**16.00m**
Overall length	**25m**
Magnetic Field	**0.4Tesla**

8 kHz (20 GB/sec)

level 0 - *special hardware*

200 Hz (4 GB/sec)

level 1 - *embedded processors*

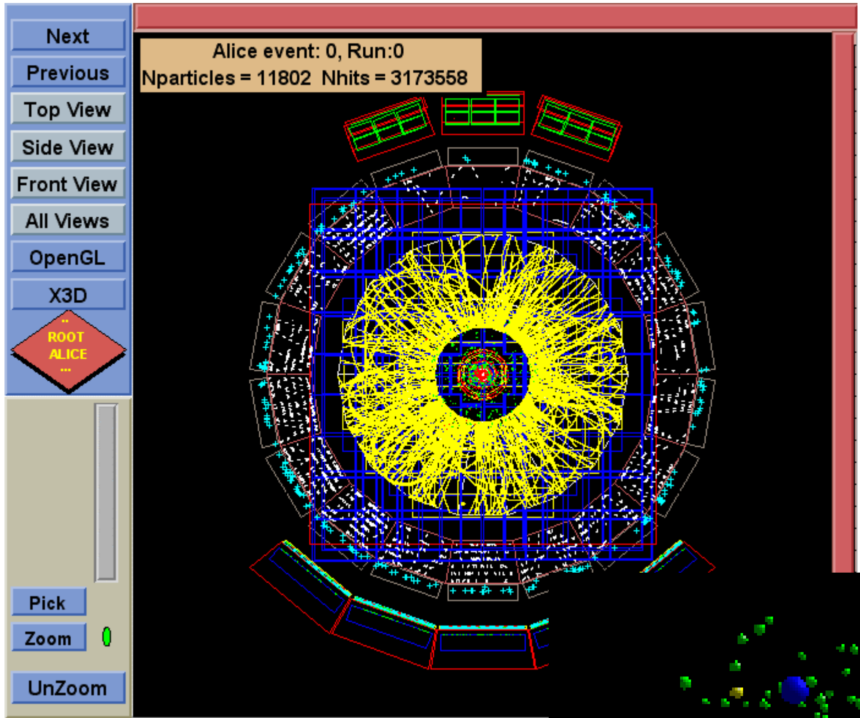
30 Hz (2.5 GB/sec)

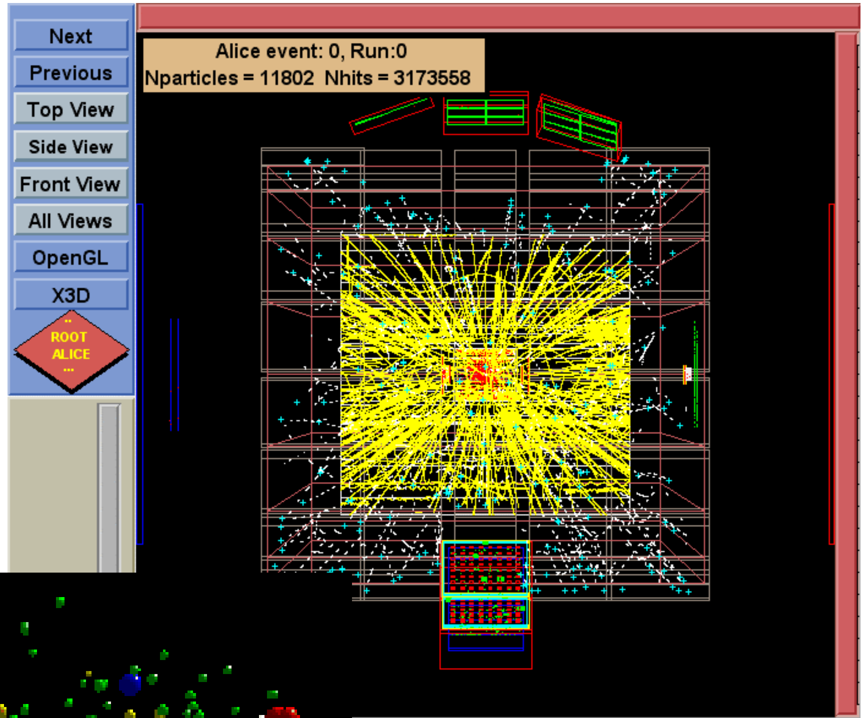
level 2 - PCs

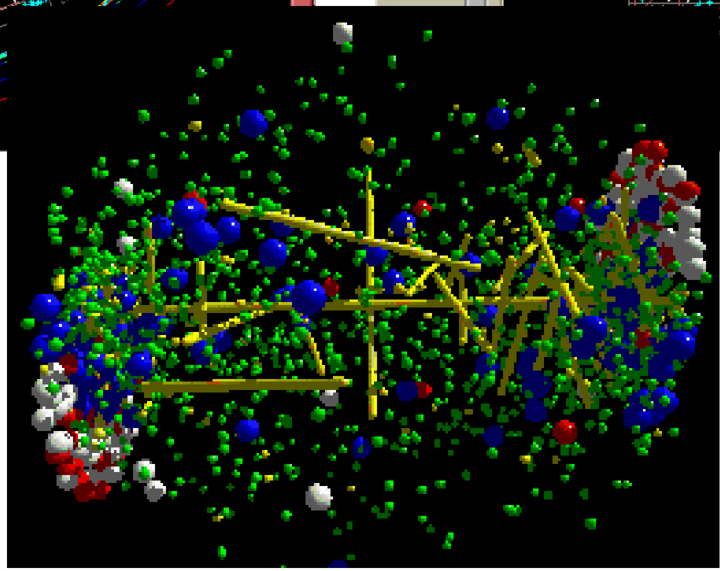
30 Hz (1.25 GB/sec)

data recording & offline analysis

A Pb-Pb Event in ALICE (1/100)







Complexity of the Problem

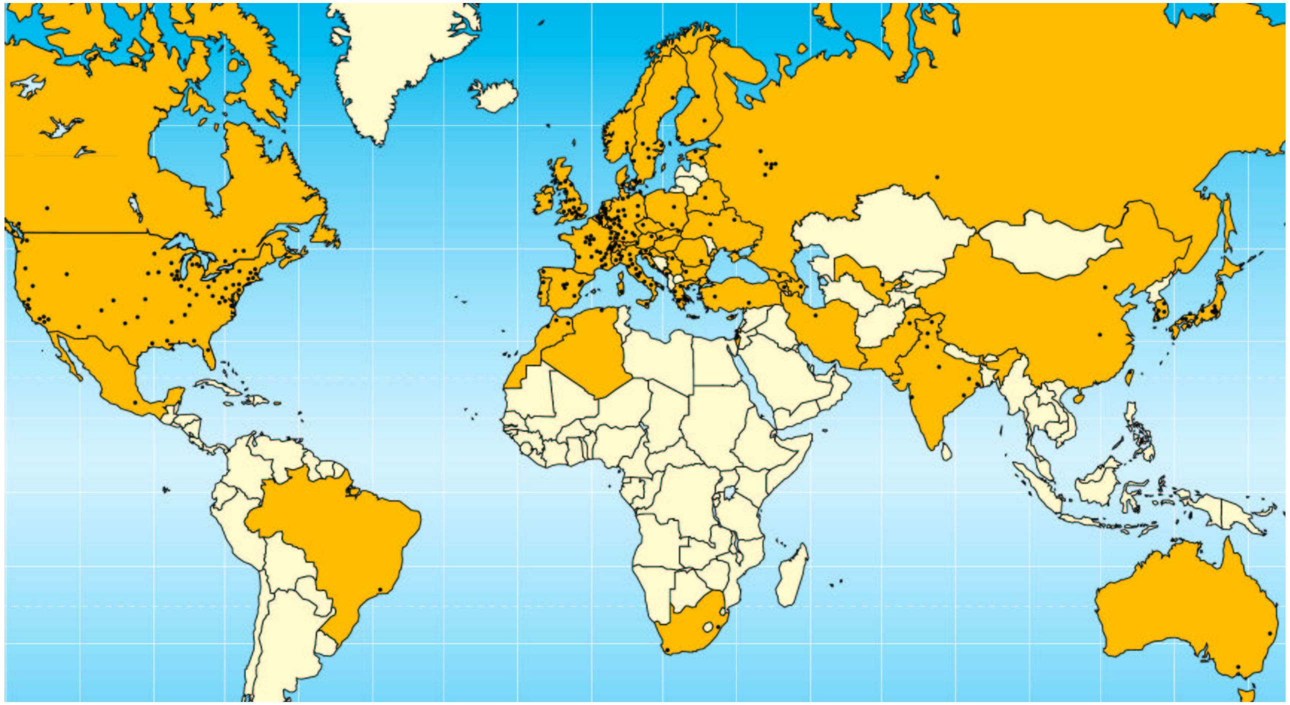
Detectors:

~2 orders of magnitude more channels than today

Triggers must choose correctly only 1 event in every 400,000

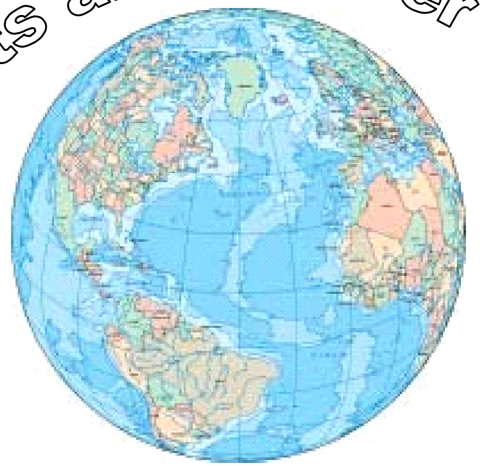
High Level triggers are software-based

Computer resources will not be available in a single location



Complexity of the Problem

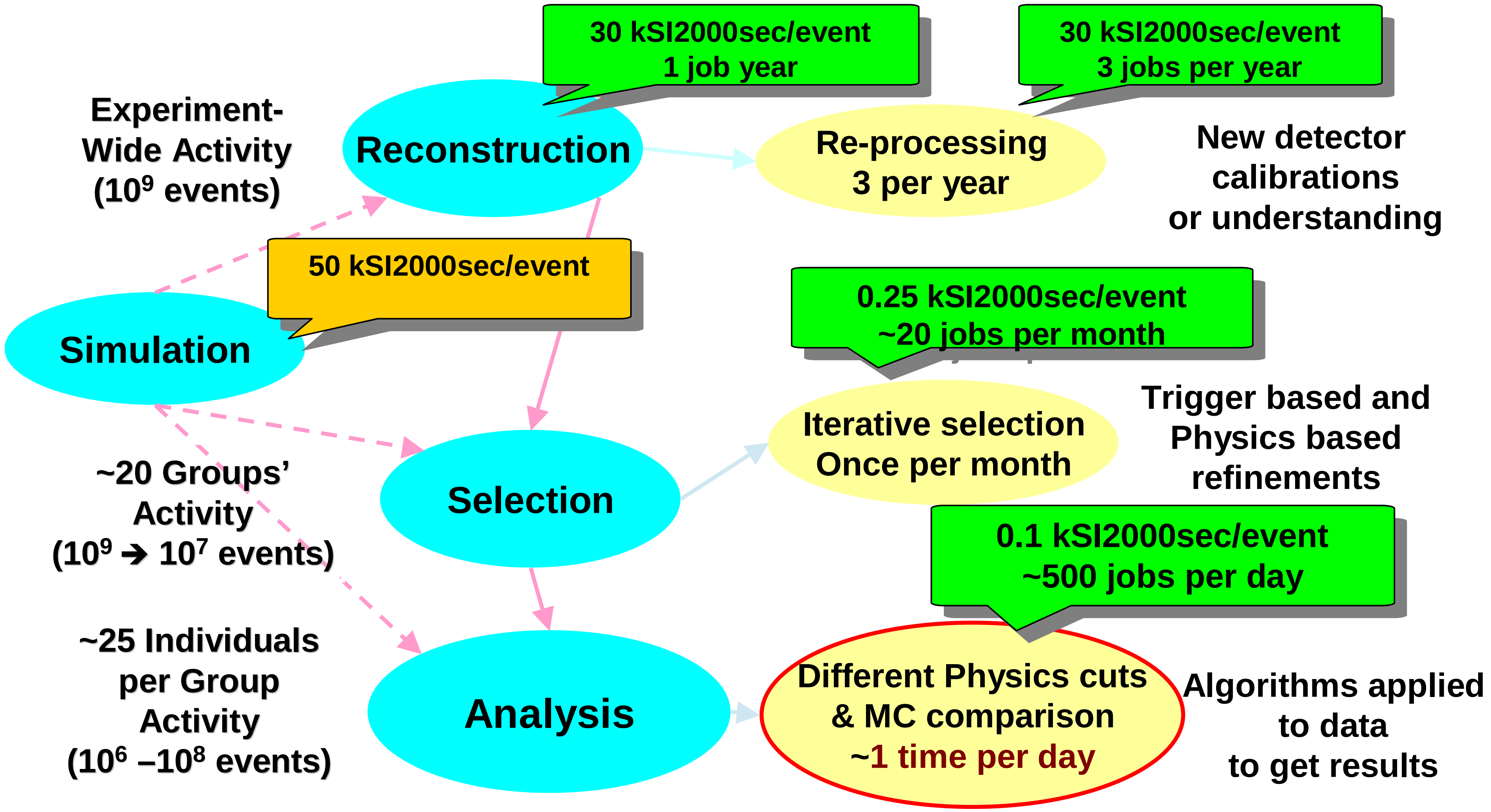
Physicists are all over the world



Major challenges associated with:

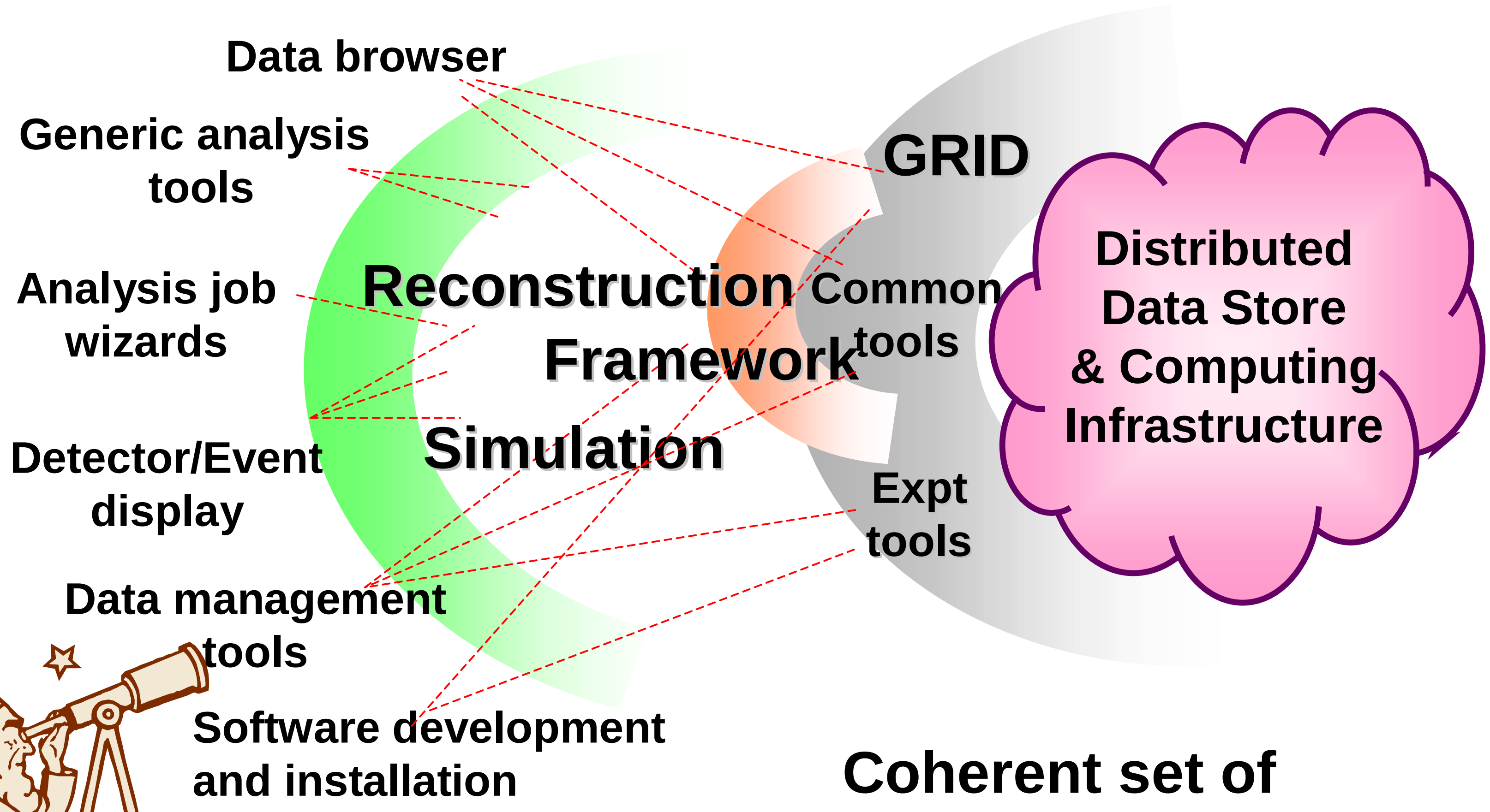
- Communication and collaboration at a distance
- Distributed computing resources
- Remote software development and physics analysis

LHC Analysis Software System



2GHz P4 CPU ~ 700 SI2000

LHC Analysis Software System



Stable User Interface

Coherent set of basic tools and mechanisms

Why We Need Distributed Computing

- The investment for LHC computing is massive
- The numbers for ALICE are
  - 1.25 GB/s in HI mode
  - ~1.5 PB/y of tape
  - ~0.5 PB of disk
  - ~1800 kSI95 (~70,000 PC2000)
  - ~ 8MEuro of hardware
    - **Without personnel + infrastructure and networking**
  - Millions lines of code to develop and maintain for 20 years
- Politically, technically and sociologically it cannot be concentrated in a single location
  - Countries will resist massive investments at CERN
  - Competence is naturally distributed
  - Cannot ask to people to travel to CERN so often

The Distributed Challenge

- Managing a large distributed software project is a challenge
- We are missing off-the-shelf tools and technologies
- But most importantly we are missing the models
  - For developing the software
  - For managing distributed resources
- In the following we will see what we are doing to face this challenge

Software Methodologies

Software Development

- In the LEP era the code was 90% written in FORTRAN
  - ~10 instructions, 50 pages!
- In the LHC era the code is in many cooperating languages, mainly C++
  - ~ 50 instructions, 700 pages – nobody understands it completely (B.Stroustrup)
  - But also C#, Java, Perl, Python, php…, Web and GRID
- Users are heterogeneous, sparse and without hierarchical structure
  - From very expert analysts to users, from 5% to 100% of time devoted to computing
- People come and go with a very high rate
  - Programs have to be maintained by people who did not develop them
  - Young physicists need knowledge they can use also outside physics
- And yet HEP software has been largely successful!
  - Experiments have not been hindered by software in their scientific goals
  - GEANT3, PAW and ROOT: in use since 20 years on all architectures and OS
- And yet we (as a community) have not used traditional SE
  - Did we do something right?

Agile Methodologies (aka SE Catching Up)

- SE response to HEP are the “Agile Methodologies”
  - Adaptive rather than predictive
  - People-oriented rather than process-oriented
  - As simple as possible to be able to react quickly
  - Incremental and iterative, short iterations (weeks)
  - Based on testing and coding rather than on analysis and design

- Uncovering better ways of developing software by valuing:

Individuals and interactions		*processes and tools*
Working software	**OVER**	*huge documentation*
Customer collaboration		*contract negotiation*
Responding to change		*following a plan*

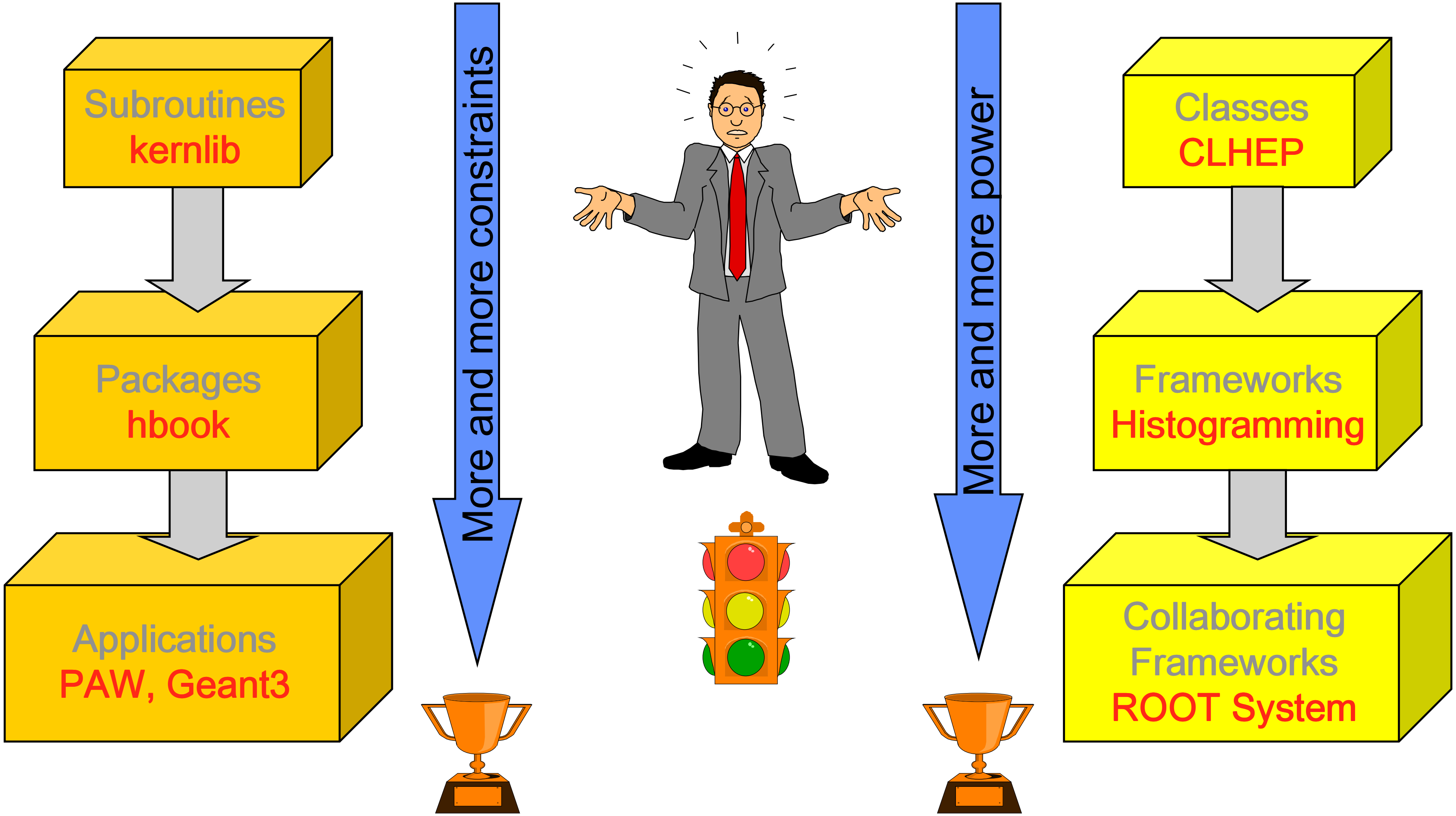
That is, while there is value in the items on the right,
We value the items on the left more

Software Development Process

- ALICE opted for a light core CERN offline team…
  - Concentrate on framework, software distribution and maintenance
- …plus 10-15 people from the collaboration
  - GRID coordination (Torino), World Computing Model (Nantes), Detector Database (Warsaw)
- A development cycle adapted to ALICE has been elaborated
  - Developers work on the most important feature at any moment
  - A stable production version exists
  - Collective ownership of the code
  - Flexible release cycle and simple packaging and installation
- Micro-cycles happen continuously
- 2-3 macro-cycles per year
  - Discussed & implemented at Offline meetings and code reviews
  - Corresponding to major code releases
- We have high-level milestones for technology and physics
  - Computing Data Challenges test technology and integration DAQ – Offline
  - Physics Data Challenges test the Offline from the physics viewpoint

Data Analysis Frameworks

HEP Software Evolution



Software Frameworks

- A framework is a collection of cooperating classes that implement a (reusable) solution for a given problem domain
- Differences between frameworks and class libraries:
  - **Behavior versus protocol:** Class libraries are collections of behaviors that you can call when you need them. A framework provides also the protocol or set of rules that govern the ways in which behaviors can be combined
  - **Don't call us, we'll call you:** With a class library, the programmer writes objects and calls their member functions. With a framework a programmer writes code that overrides and is called by the framework. The framework manages the flow of control among its objects
  - **Implementation versus design:** With class libraries programmers reuse only implementations, with frameworks they reuse design
- The scope of the framework evolves with the evolving needs and understanding of the users

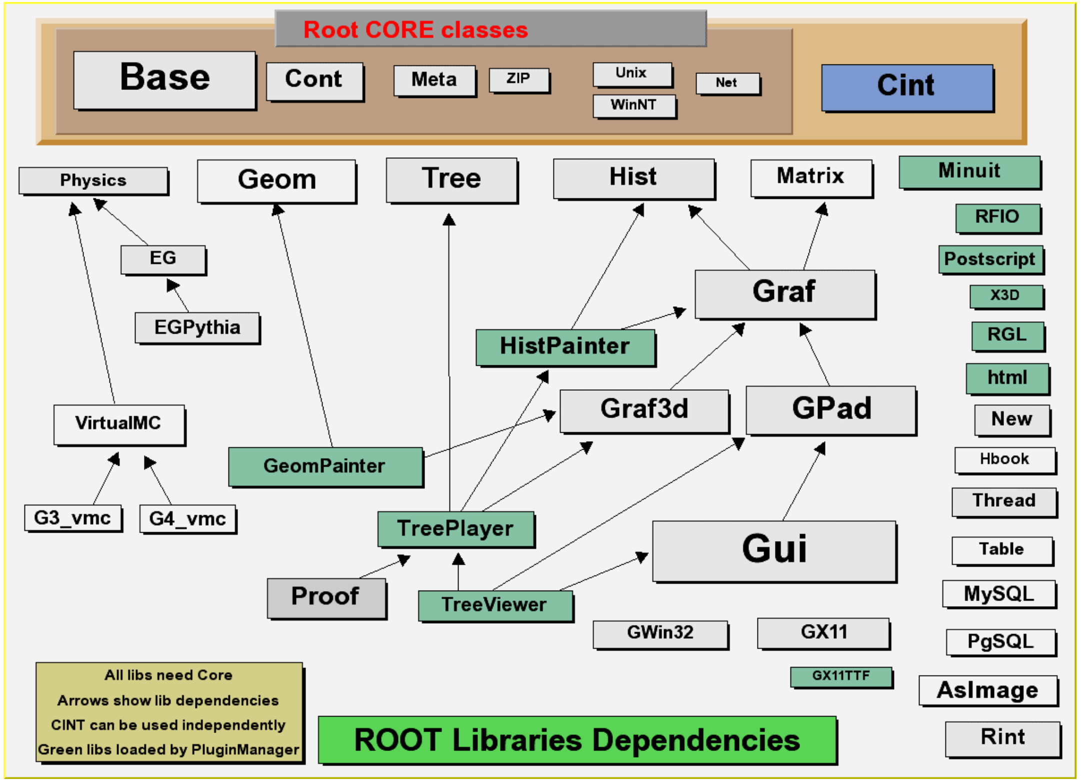
Features of a Data Analysis Framework

- Being able to support the full data analysis chain
  - Raw data, DSTs, mini-DSTs, micro-DSTs
- Being able to handle complex structures
  - Complete objects
  - Object hierarchies
- Support at least the common (PAW) data analysis functionality
  - Histogramming
  - Fitting
  - Visualization
- GUI, object browsers and inspectors
  - Extensive use of object dictionary and RTTI
- Only one language
  - C++
- Better maintainable
  - Use OOP
- Make the system extensible
  - Use OO framework technology

The ROOT Framework

- The ROOT system is an Object Oriented framework for large scale data handling applications
  - Written in C++
  - Provides, among others,
    - **An efficient hierarchical OO database**
    - **A C++ interpreter**
    - **Advanced statistical analysis (multi dimensional histogramming, fitting, minimization and cluster finding algorithms)**
    - **Visualization tools**
    - **And much, much more**
  - The user interacts with ROOT via a graphical user interface, the command line or batch scripts
  - The command and scripting language is C++, thanks to the embedded CINT C++ interpreter and large scripts can be compiled and dynamically loaded

The ROOT Libraries



- Over 650 classes
- 950,000 lines of code
- CORE (10 Mbytes)
- CINT (3 Mbytes)
- Green libraries linked on demand via plug-in manager (only a subset shown)

ROOT Statistics – Distributions and Number of Users

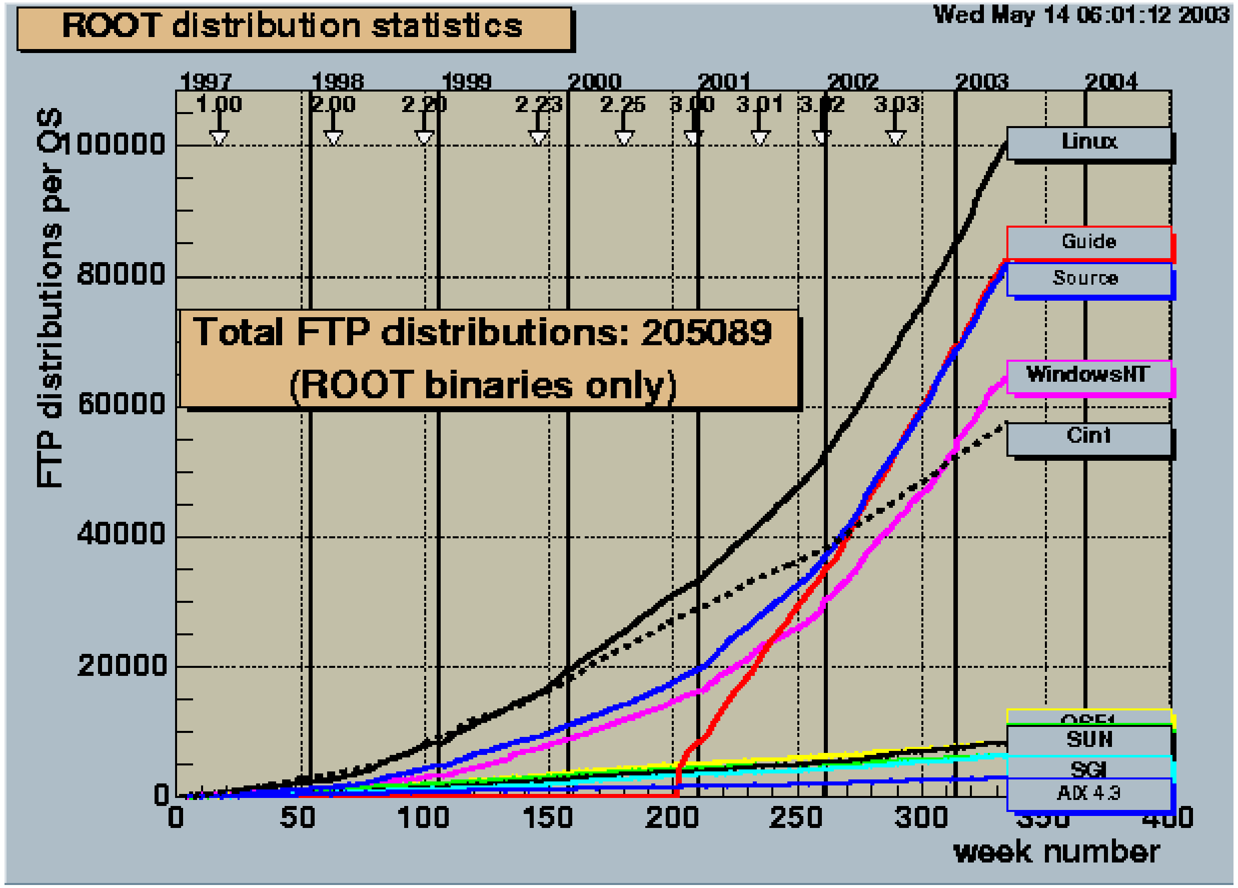
205,000 binaries downloaded

>1,000,000 clicks per month

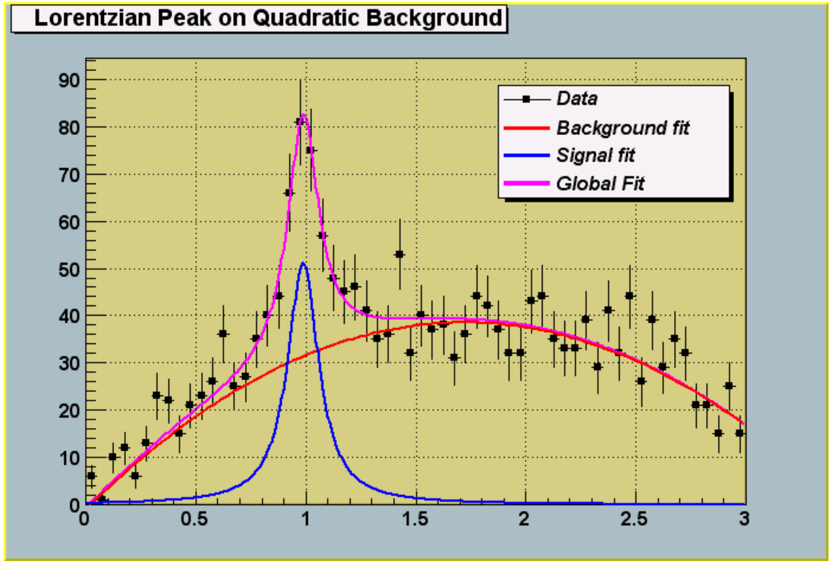
30,000 docs in 12 months

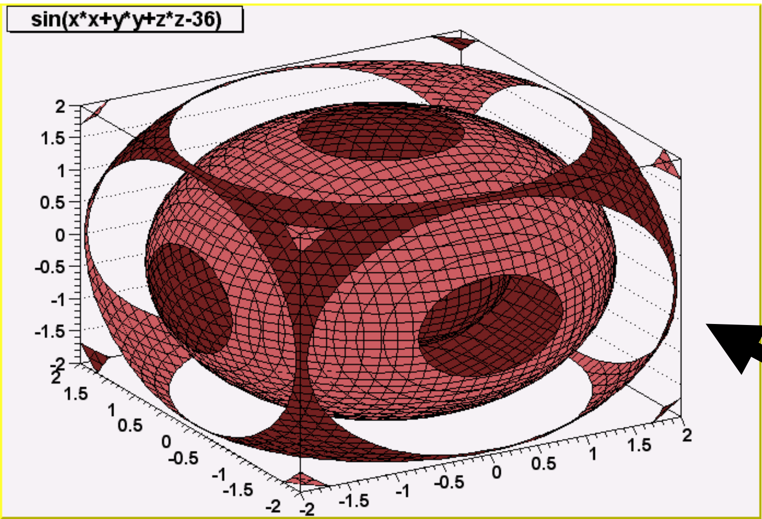
3000 registered users

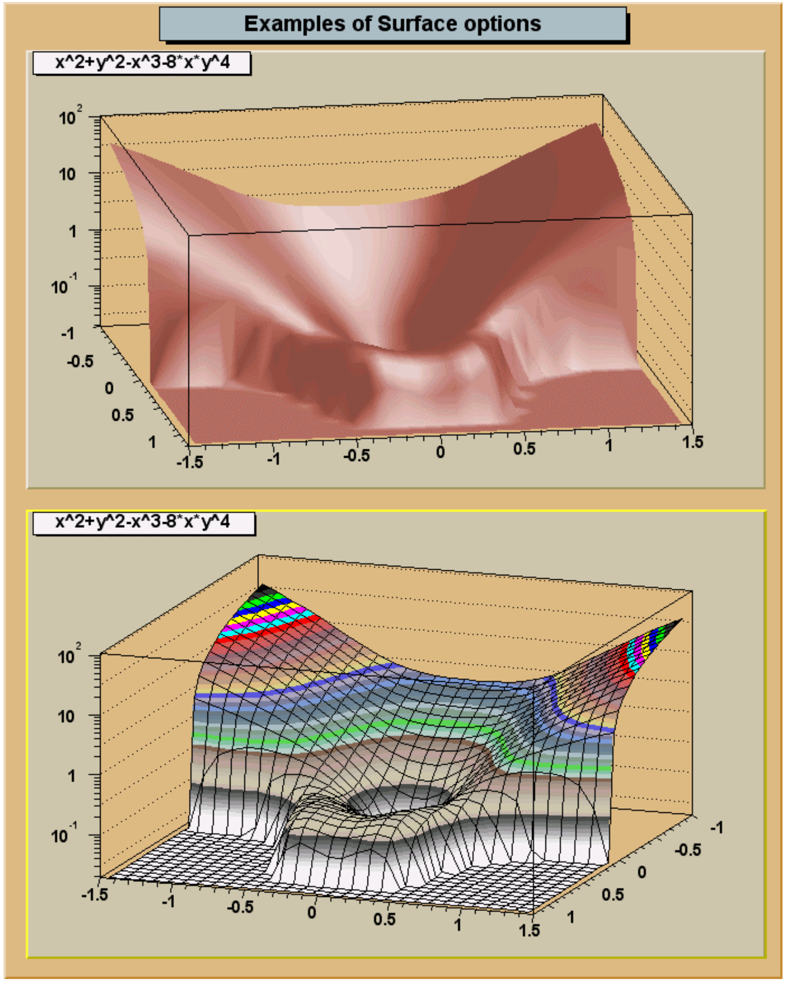
960 users subscribed to roottalk



Examples of ROOT Graphics



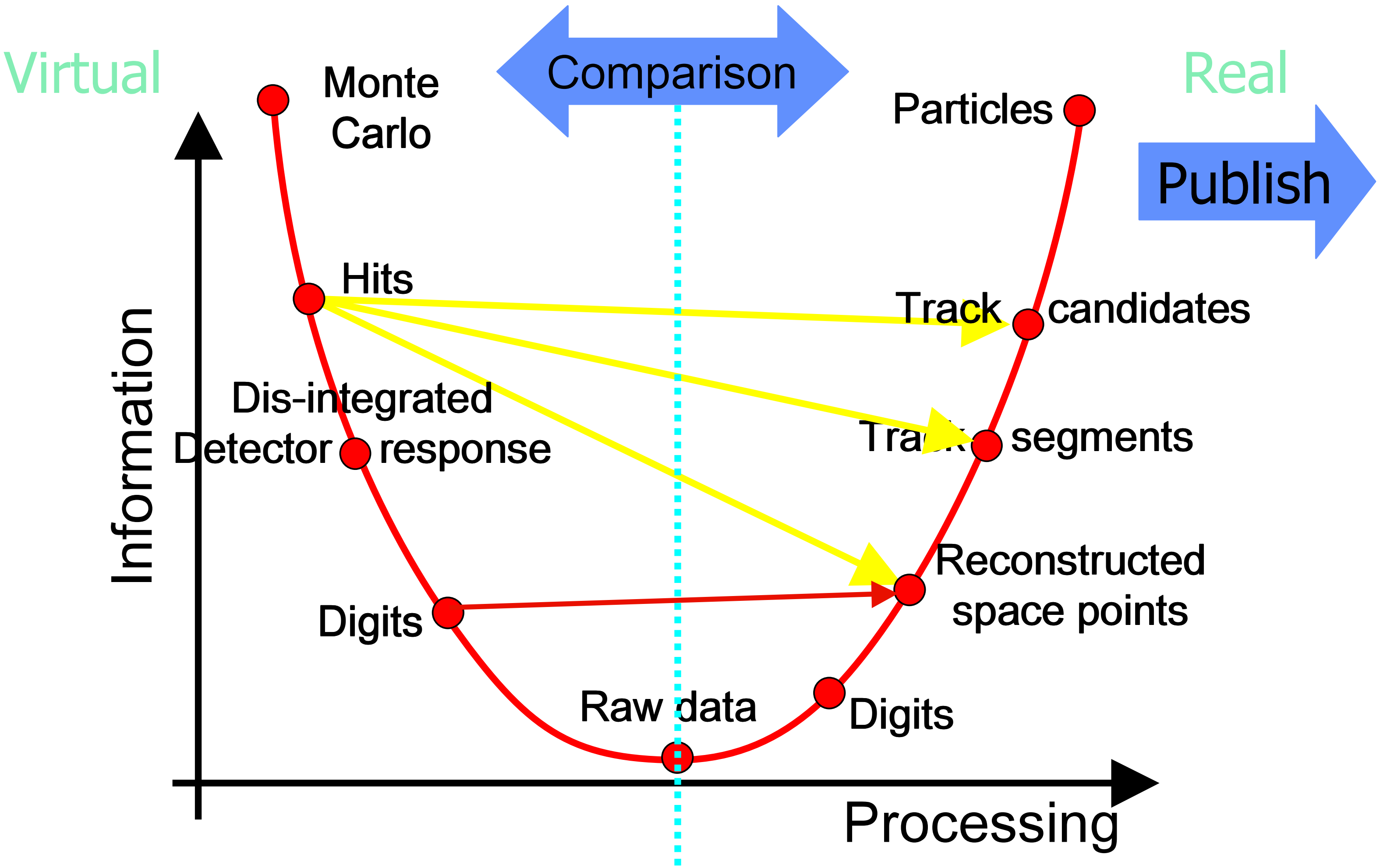





```
TF3 f3("f3","sin(x*x+y*y+z*z-36)",-2,2,-2,2,-2,2);
f3->Draw();
```

The Experiment Frameworks

What Does a HEP Framework Do



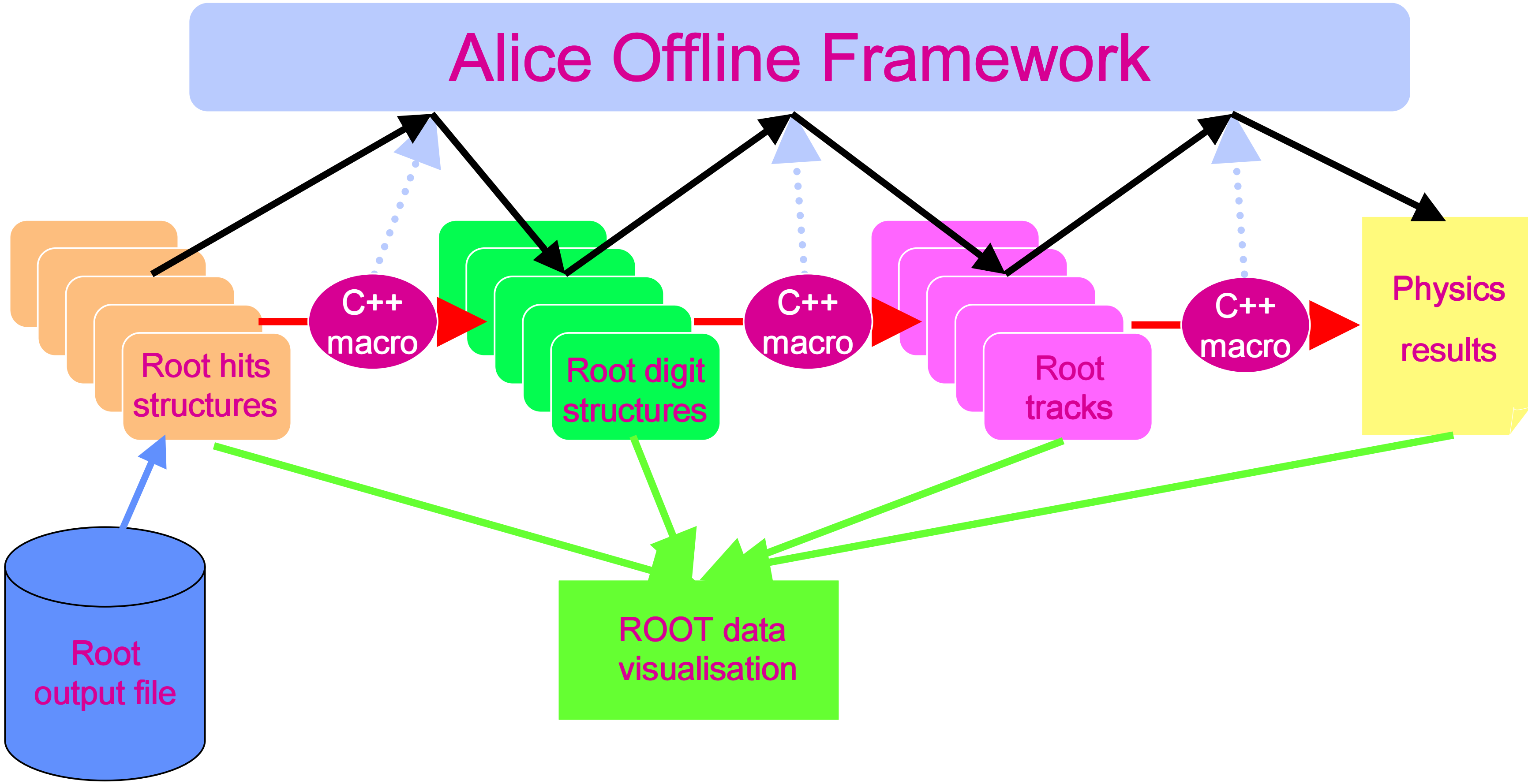
AliRoot – The ALICE Framework

- Development started in 1998
- More than 50 users participate in the development of AliRoot from the detector groups
  - 70% of the code developed outside, 30% by the core Offline team
  - C++: 400kLOC + 225kLOC (generated) + macros: 77kLOC
  - FORTRAN: 13kLOC (ALICE) + 914kLOC (external packages)
  - Maintained on Linux (any version!), HP-UX, DEC Unix, Solaris
- Two packages to install (ROOT+AliRoot)
  - 1-click-away install: download and make
  - Less than 1 second to link (thanks to the use of shared libraries)
- Installed on more than 30 sites
- Fully interfaced to the AliEn grid (see later)
- Single framework for simulation, reconstruction and visualization

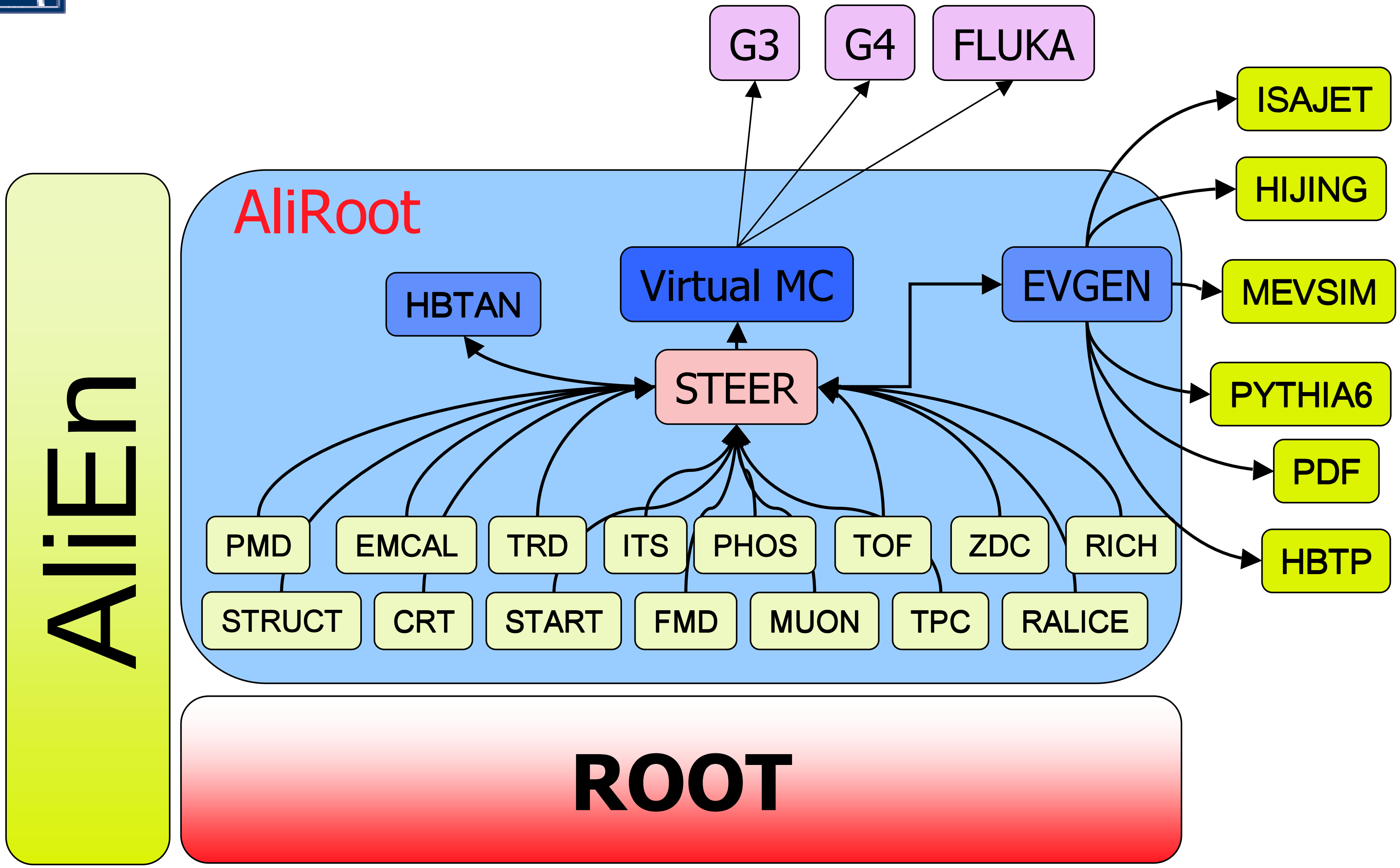
AliRoot Highlights

- AliRoot development highlights
  - Attention to the requirements of large production and analysis
  - Evolution to higher levels of abstraction for objects and procedures
  - Whiteboard data communication
  - Generic task handling
  - Event merging framework
  - New atomic file structure
- Large use of abstract interfaces
  - Independence and modularity of the different detector modules
  - Easy integration of new developments in evolving framework
- Full exploitation of the ROOT integrated framework

AliRoot Evolution Schema



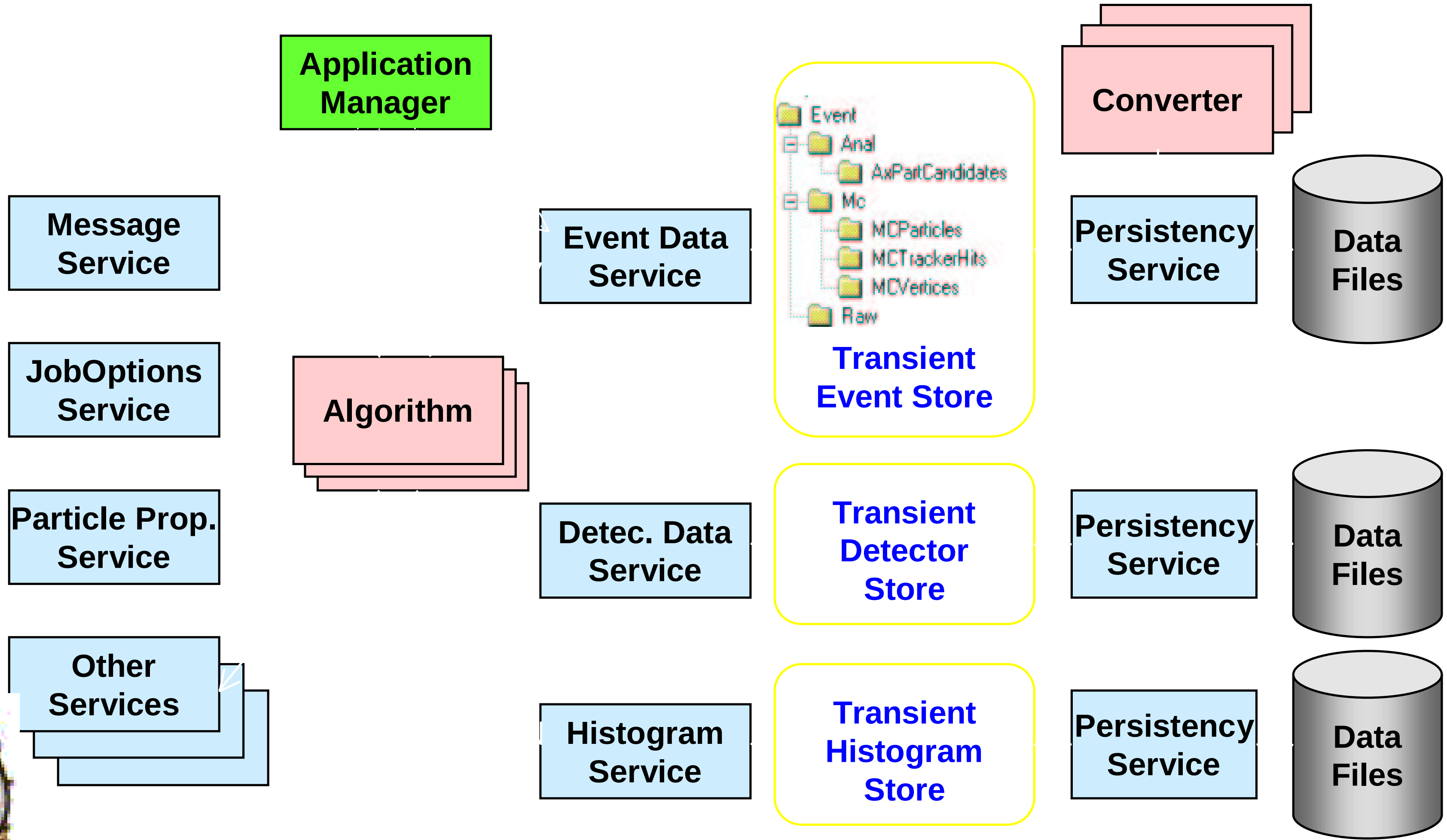
AliRoot Layout

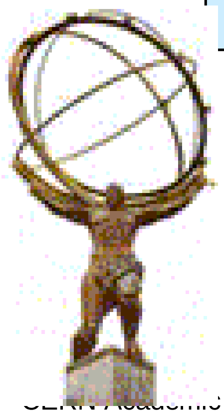


Structure of the framework reflects the ALICE experiment sub-structure

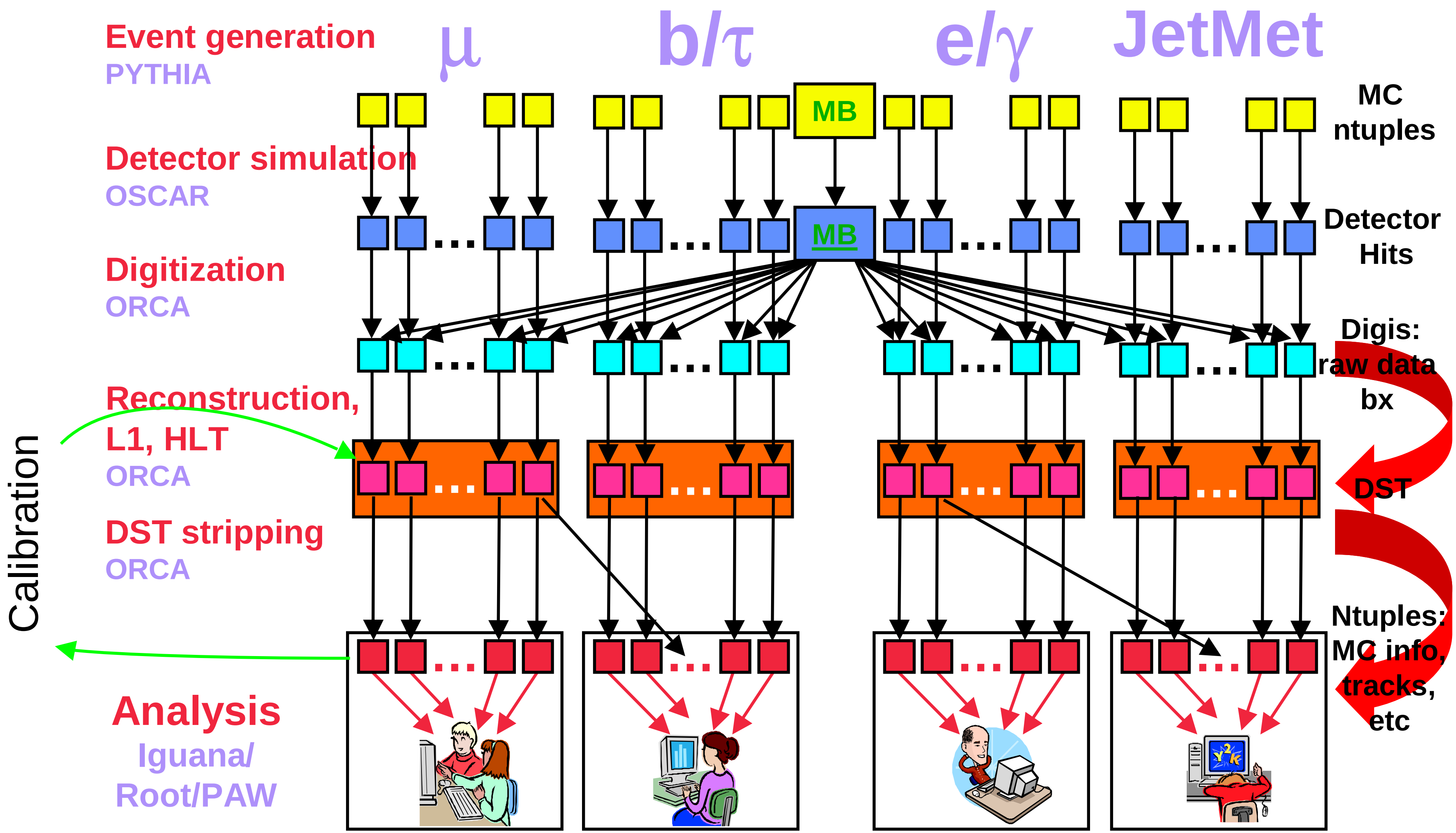
STEER coordinates the cooperation between detectors

Gaudi – ATLAS/LHCb Framework



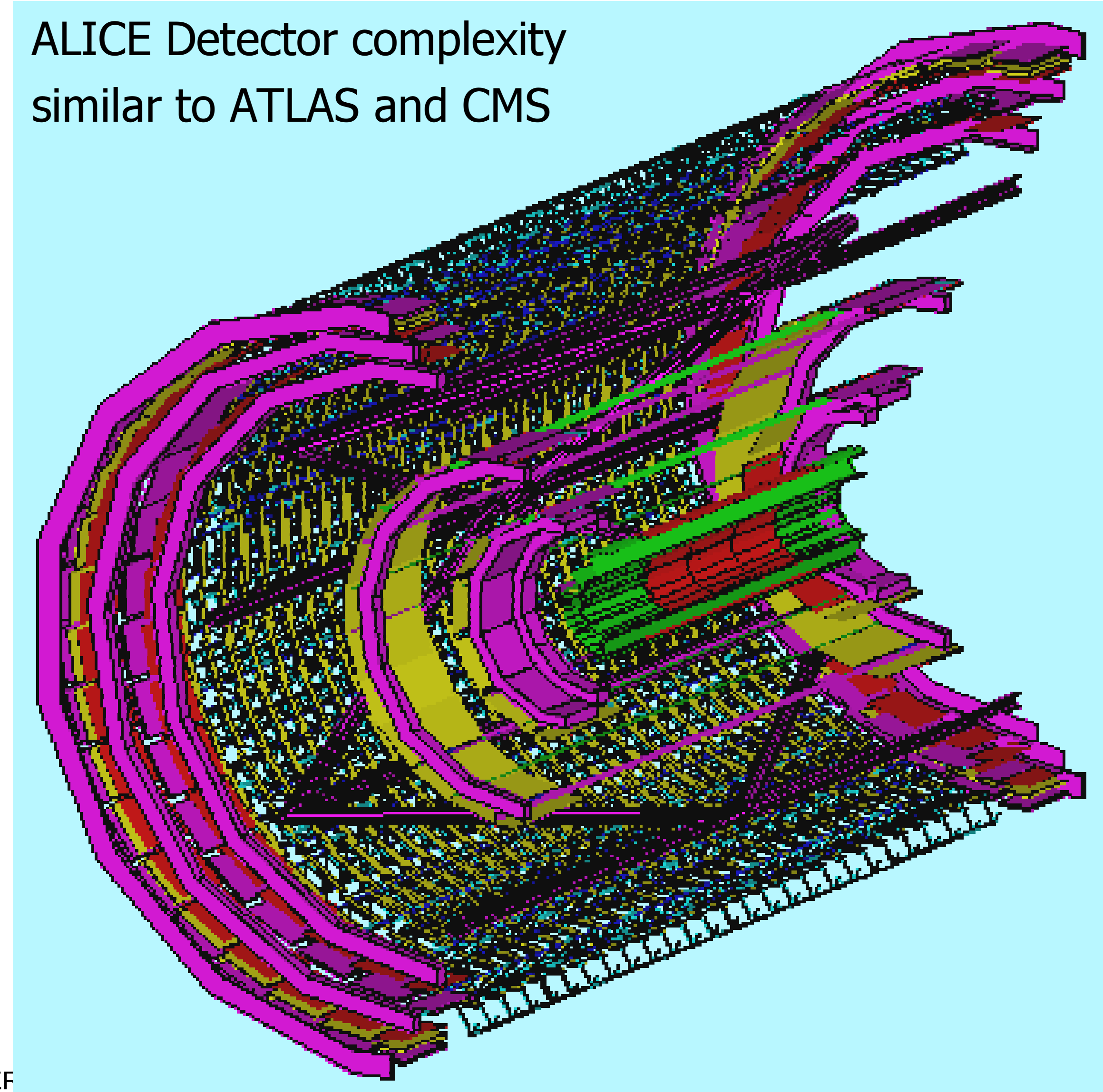


CMS Analysis Production Chain



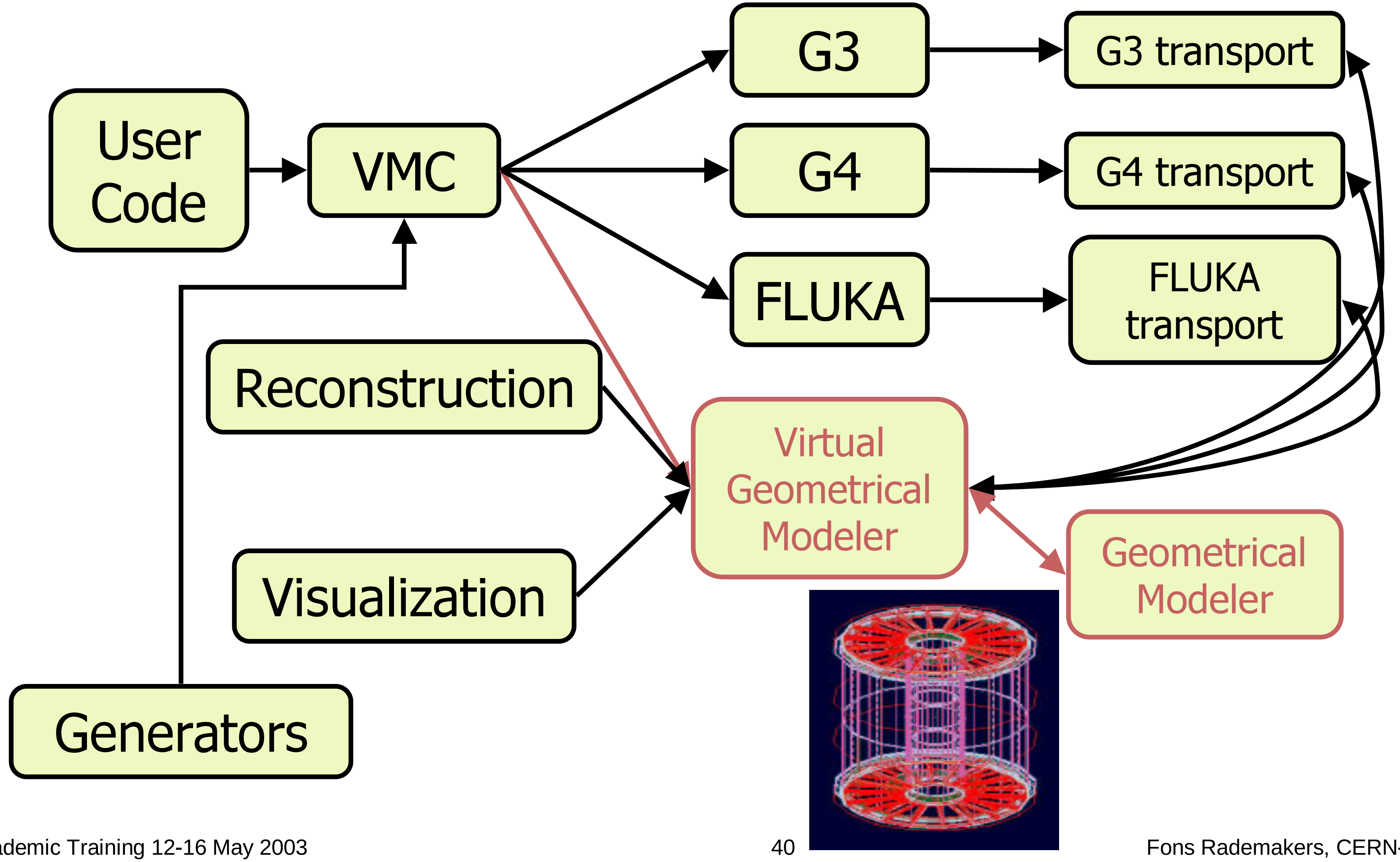
Simulation

ALICE Event/100



- GEANT3
  - Developed in 1981
  - Still used by the majority of experiments
- Geant4
  - A huge investment
  - Slow penetration in HEP experiments
- FLUKA
  - State of the art for hadronic and neutron physics
  - Difficult to use for full detector simulation

The Virtual MC

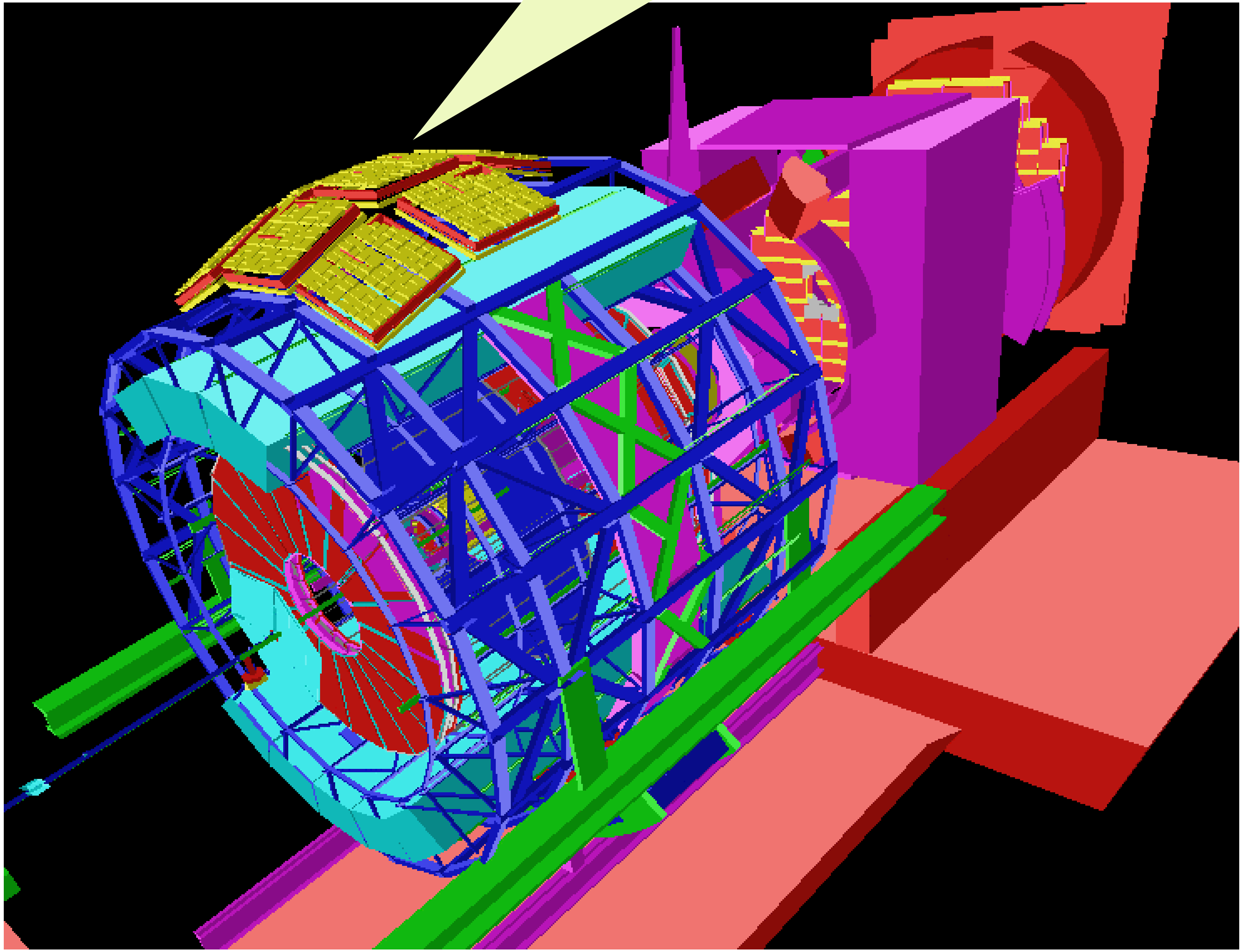


ROOT Geometrical Modeler

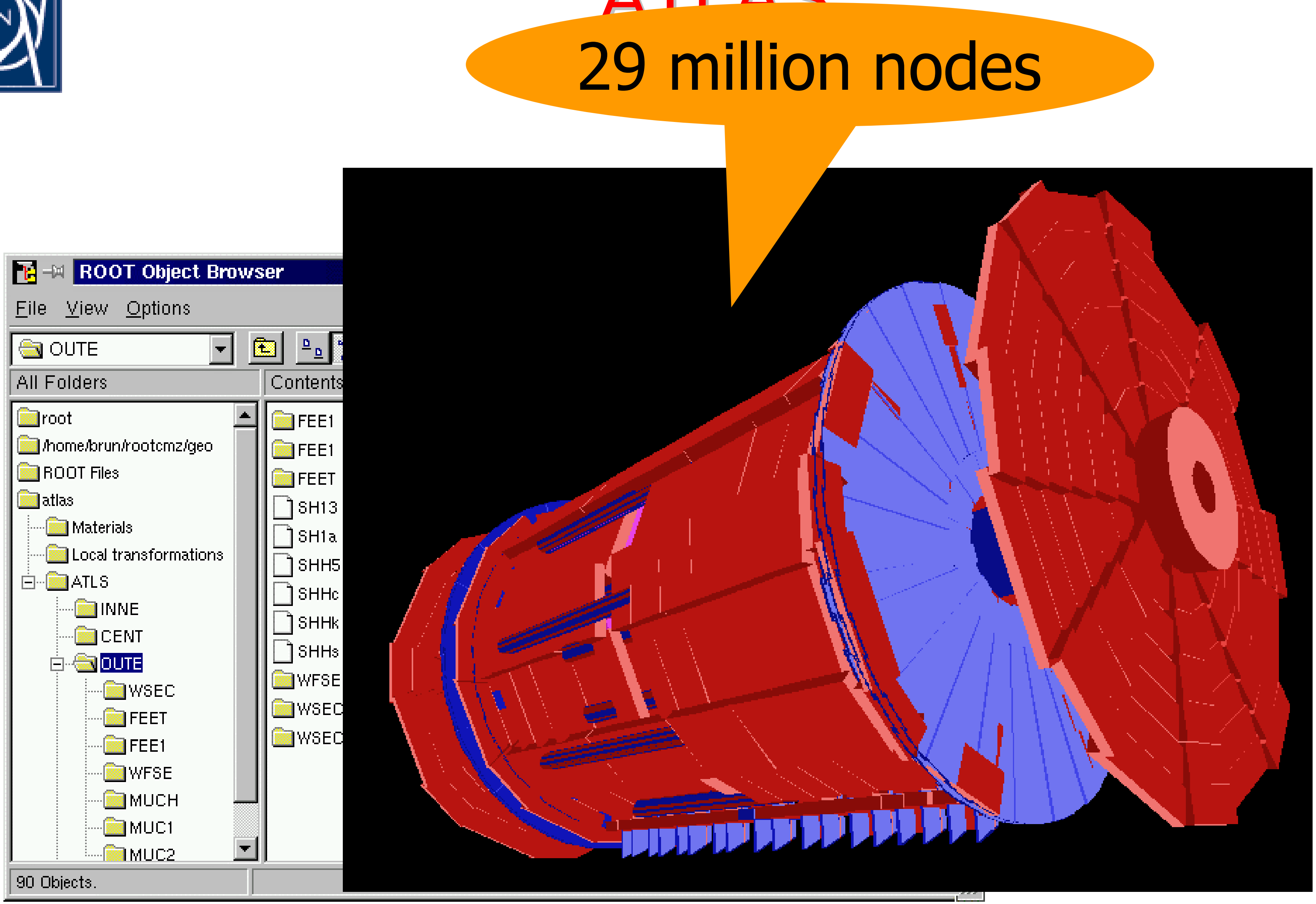
- The ROOT geometry package is intended as a framework to provide the geometrical description of a detector, full tracking functionality and additional tools to ease the building, checking and debugging of a geometry. Main features:
  - Modeling
  - Visualization
  - Interactivity
  - Where am I?
  - Distance to boundary
  - Closest boundary
  - Persistency
- Started ~1 year ago as a common ALICE/ROOT effort having in mind the idea to run several MC's with the same user code

ALICE

3 million volumes



ATLAS

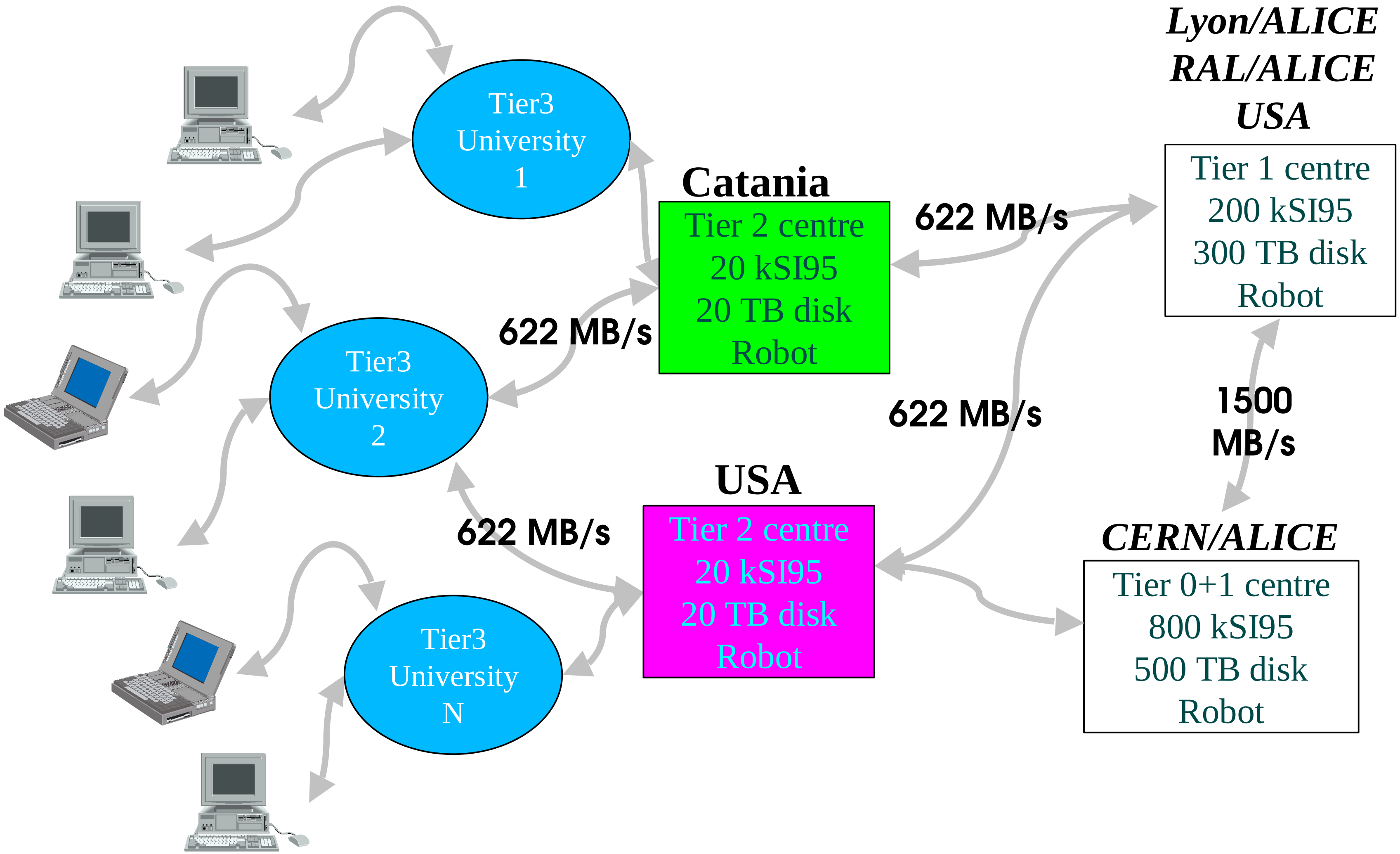

29 million nodes
ROOT Object Browser
File View Options
OUTE
All Folders
Contents
root
/home/brun/rootcmz/geo
ROOT Files
atlas
Materials
Local transformations
ATLS
INNE
CENT
OUTE
WSEC
FEET
FEE1
WFSE
MUCH
MUC1
MUC2
FEE1
FEE1
FEET
SH13
SH1a
SHH5
SHHc
SHHk
SHHs
WFSE
WSEC
WSEC
90 Objects.

Distributed Computing and the GRID

Access to Resources

- Resources for HEP computing (CPU's, disks and Mass Storage) will be distributed
  - They will be concentrated in so-called *regional centres*
- The different centres will have to work as a single integrated system providing
  - Maximisation of the usage of the resources
  - Redundancy and fault tolerance
  - Security
  - Maximum transparency of usage
- Physicists have realised the challenge of this already since few years
  - A study group (MONARC) has been put together already some years ago and a model has been elaborated

The Monarc Model



The Distributed Computing Model

- Basic principle
  - Every physicist should have in principle equal access to the data and to the resources
- Sample assumptions
  - Raw data will be kept at CERN and Tier 1's
  - Reconstruction done at CERN and data shipped to Tier1-2
  - Simulation done in the Tier1-2, data shipped to CERN
  - The Tier1-2 re-process data as many time as necessary
  - Users access data remotely
- The system will be extremely complex
  - Number of components in each site
  - Number of sites
  - Different tasks performed in parallel: simulation, reconstruction, scheduled and unscheduled analysis

The Challenge

- Bad news is that the basic tools are missing
  - Distributed resource management
  - Distributed namespace for files and objects
  - Distributed authentication
  - Local resource management of large clusters
  - Data replication and caching
  - WAN/LAN monitoring and logging
- Good news is that we are not alone
  - All the above issues are central to the new developments going on in the US and now in Europe under the collective name of GRID

The Grid Concept

- Grid R&D has its origins in high-end computing & metacomputing, but…
- In practice, the "Grid problem" is about *resource sharing & coordinated problem solving in dynamic, multi-institutional virtual organizations*
- Data is often the focus
  - As opposed to classical numerically intensive simulations
- The analogy is with the power grid

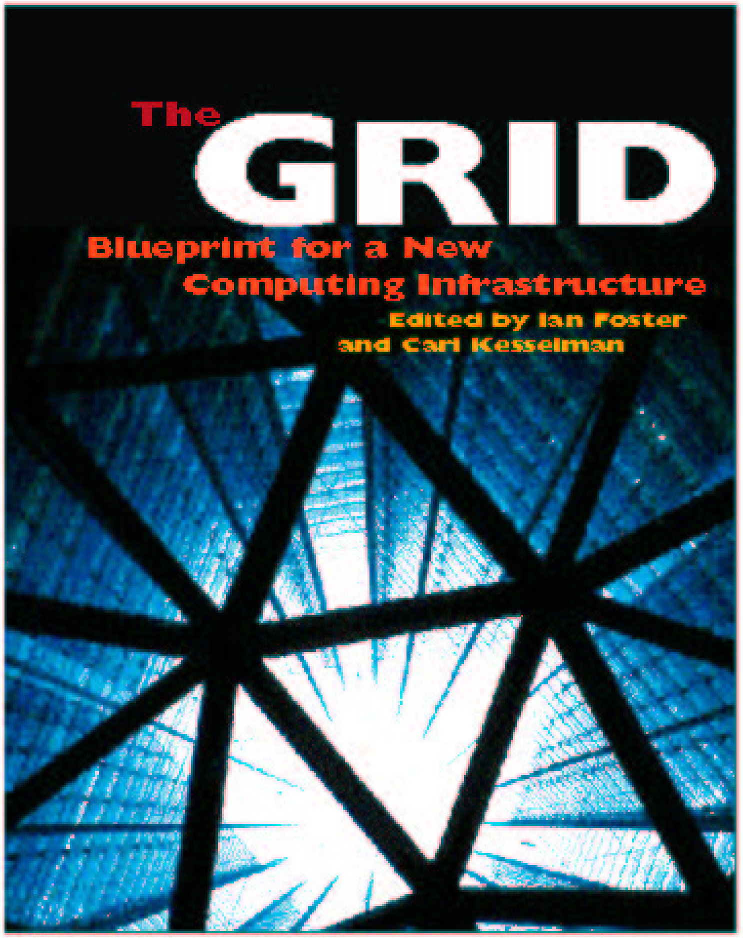
The Grid:
Blueprint for a New Computing Infrastructure

I. Foster, C. Kesselman (Eds), Morgan Kaufmann, 1999

- ✓ Available July 1998;
- ✓ ISBN 1-55860-475-8
- ✓ 22 chapters by expert authors including Andrew Chien, Jack Dongarra, Tom DeFanti, Andrew Grimshaw, Roch Guerin, Ken Kennedy, Paul Messina, Cliff Neuman, Jon Postel, Larry Smarr, Rick Stevens, and many others

“A source book for the history of the future” -- Vint Cerf

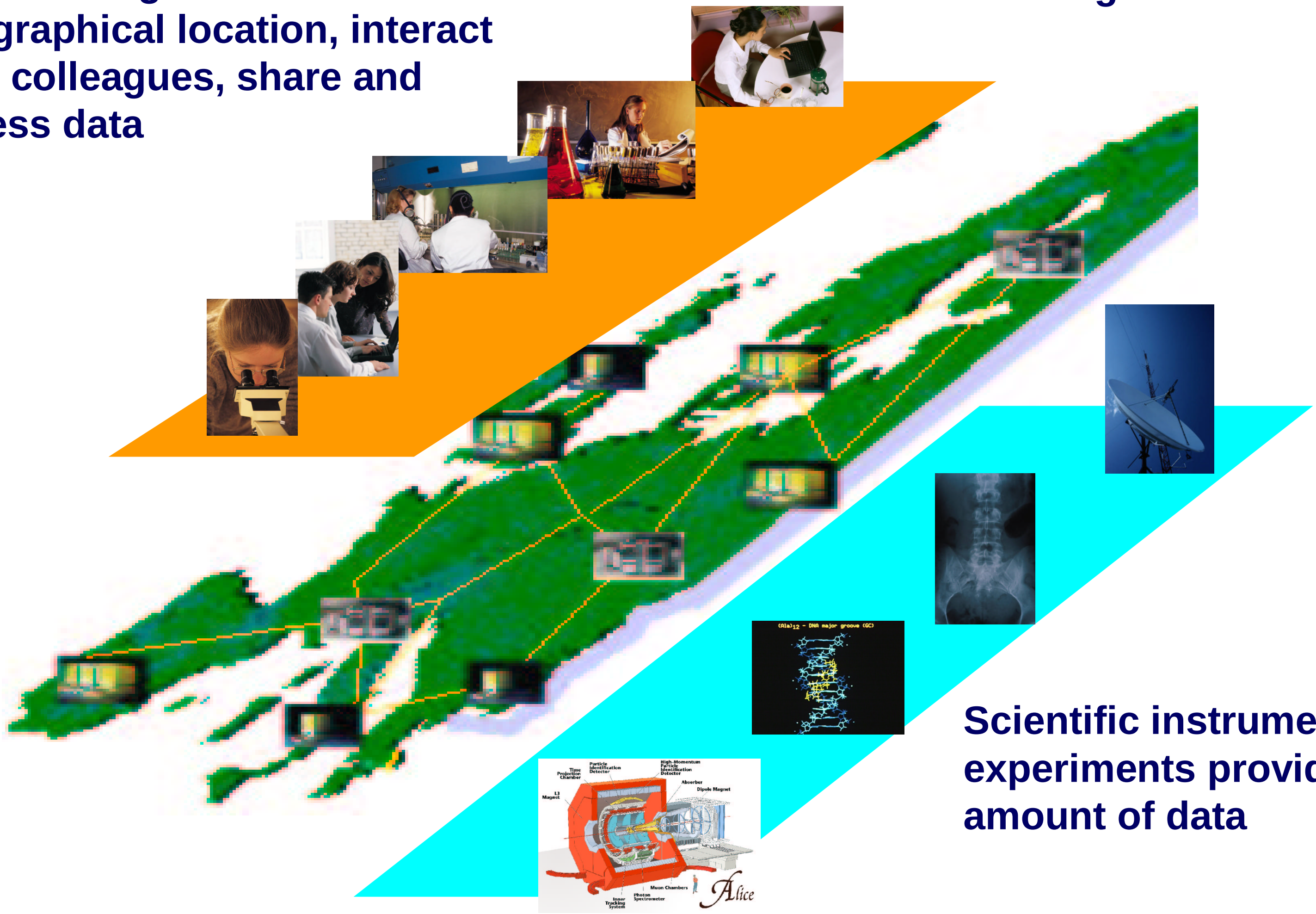
http://www.mkp.com/grids



The Grid Vision

The GRID: networked data processing centres and ”middleware” software as the “glue” of resources.

Researchers perform their activities regardless geographical location, interact with colleagues, share and access data



Scientific instruments and experiments provide huge amount of data

EU Grid Projects

- DataGRID
  - Started on 1-01, 9.8M EU funding, 21 partners
  - Main CERN, PPARC, INFN, CNRS, NIKHEF, ESA
  - Focus on middleware, test beds and applications, 90% HEP/LHC focussed
  - Intensive collaboration with other Grid projects
    - **Europe: GridPP, INFN-Grid, CrossGrid, DataTAG**
    - **US: GriPhyN, PPDG, DTF, iVDGL**
  - Support similar activities in other sciences
- Other EU projects: CrossGrid, DataTAG and, later, GRIDStart

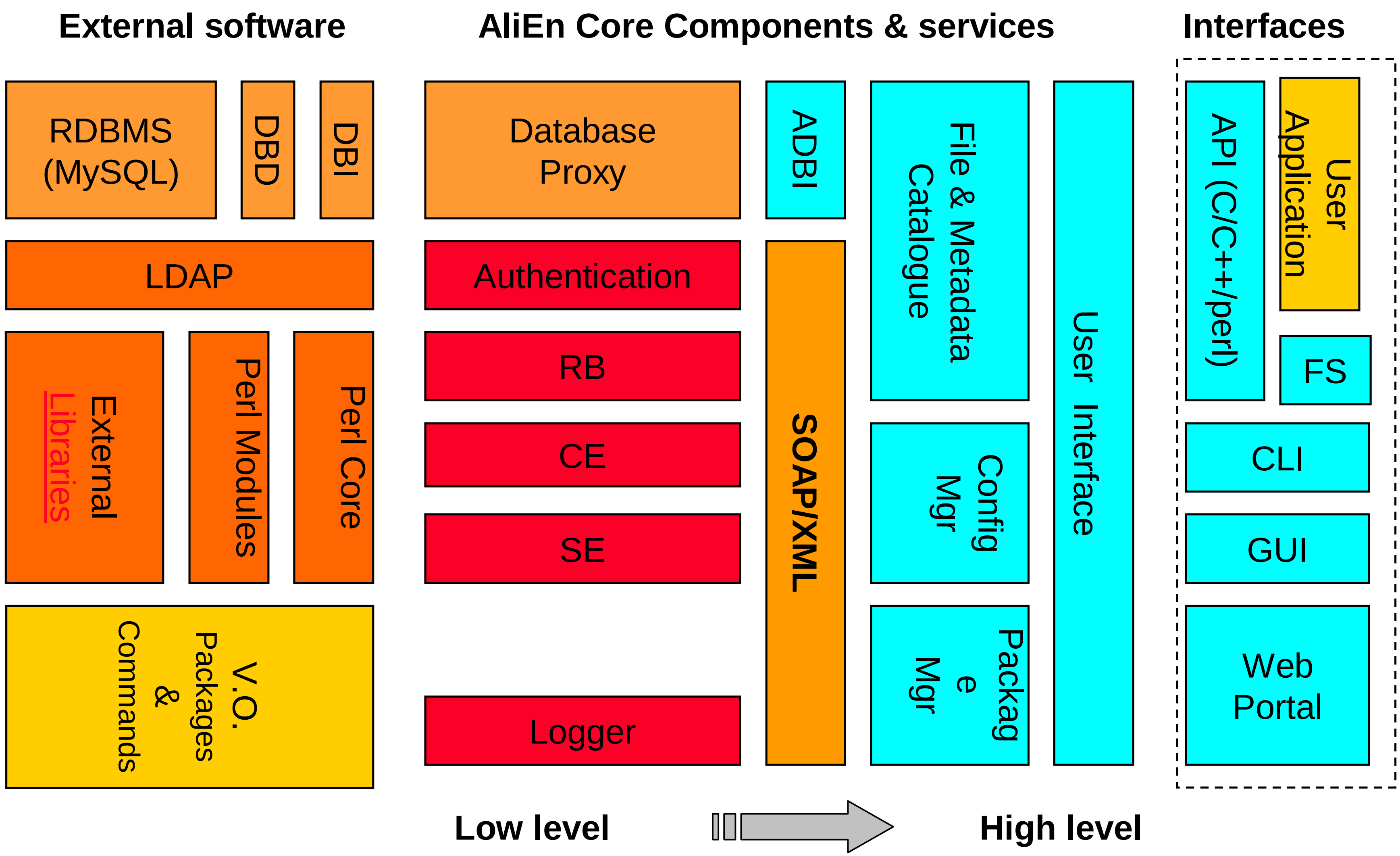
AliEn a Lightweight GRID

- AliEn (http://alien.cern.ch) is a lightweight alternative to full blown GRID based on standard components (SOAP, Web services)
  - Distributed file catalogue as a global file system on a RDBMS
  - TAG catalogue, as extension
  - Secure authentication
  - Central queue manager ("pull" vs "push" model)
  - Monitoring infrastructure
  - C/C++/perl API
  - Automatic software installation with AliKit

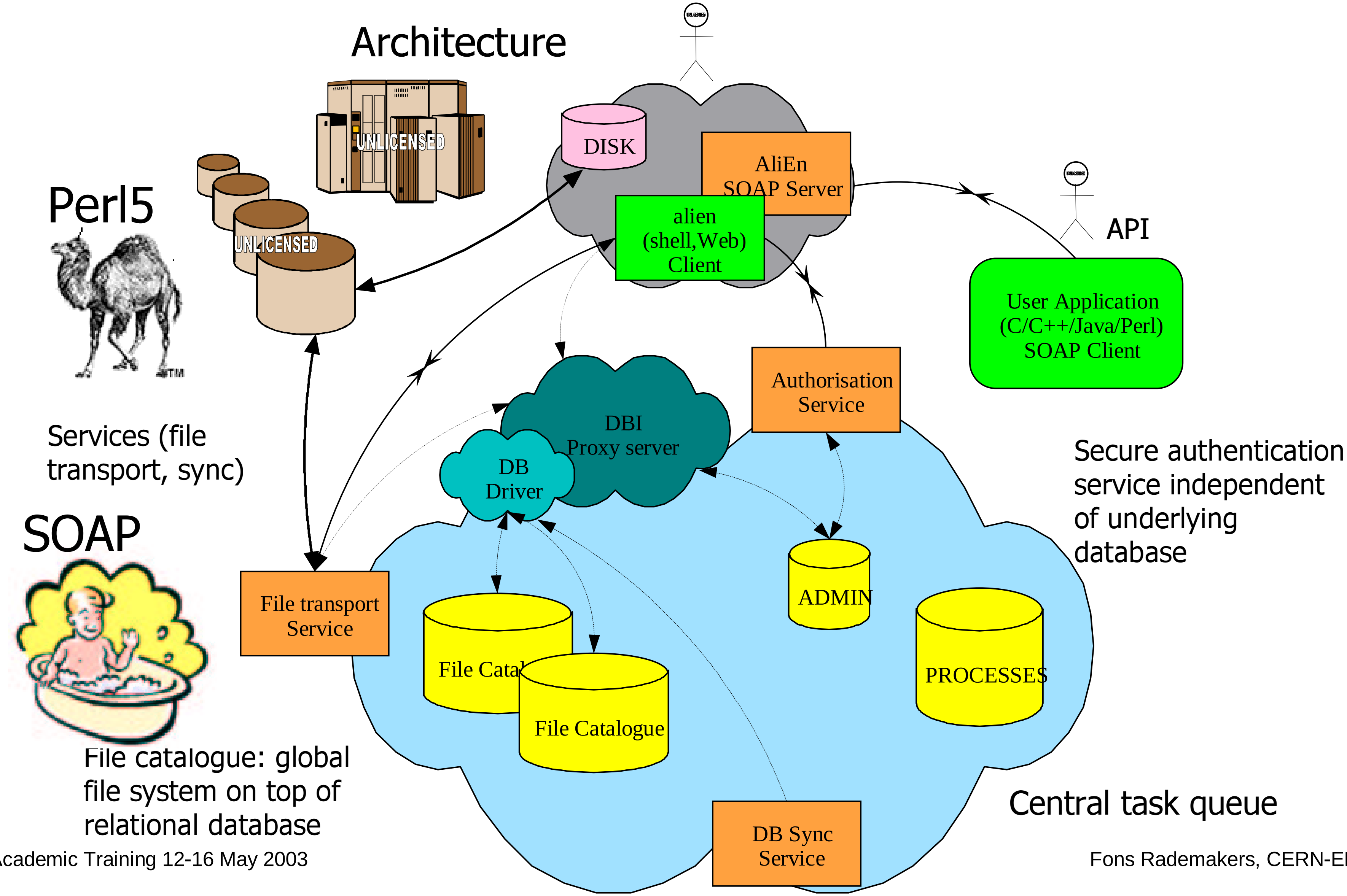
The Core GRID Functionality !!

- AliEn is routinely used in production for ALICE

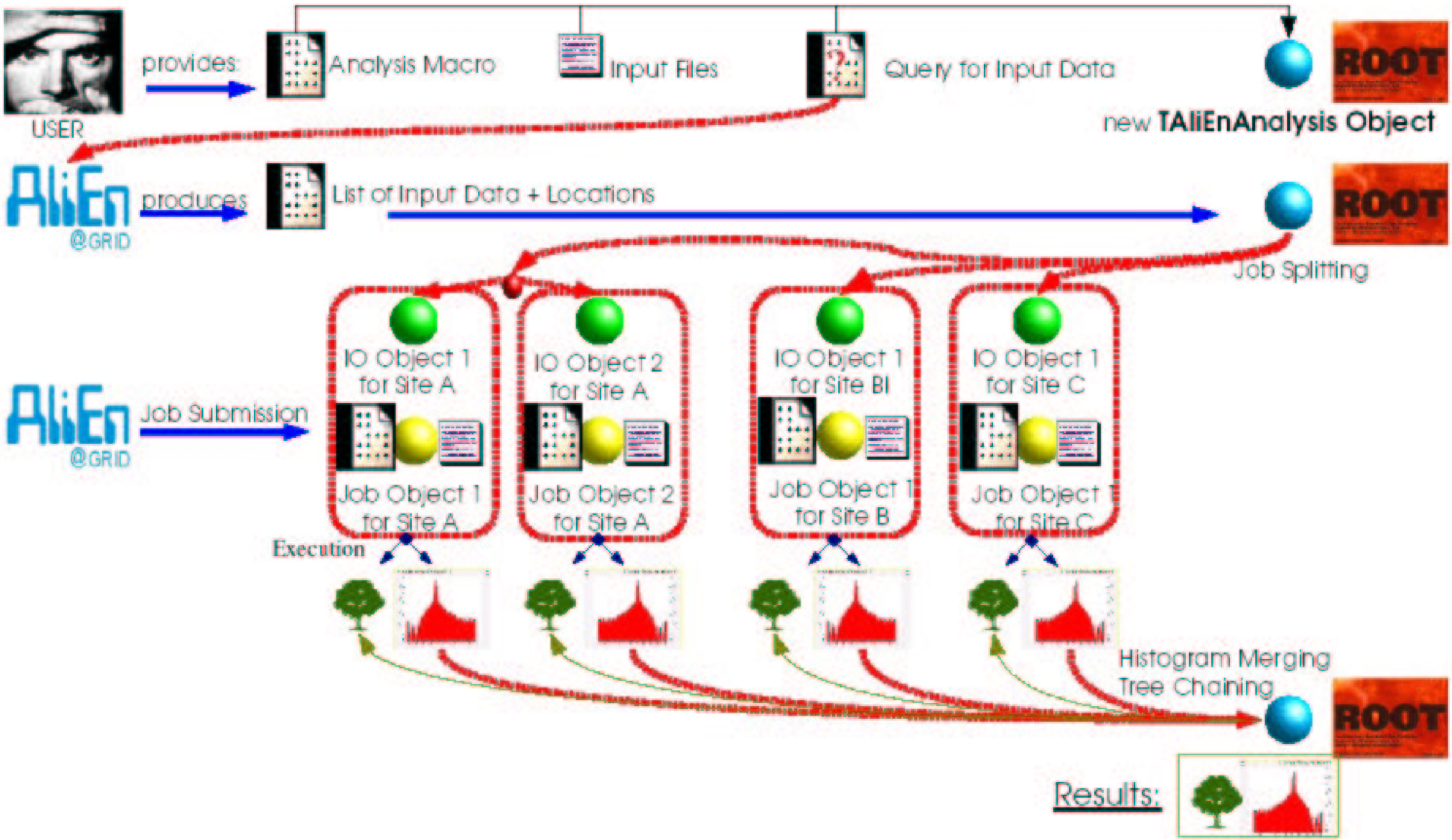
AliEn Architecture



AliEn Components

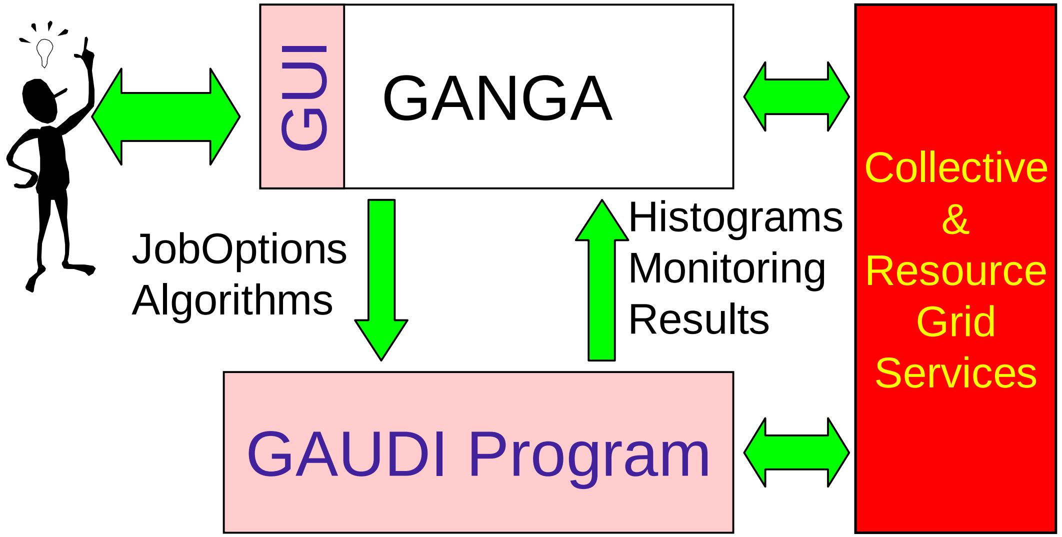


ROOT and the AliEn Grid



GANGA – Gaudi And Grid Alliance

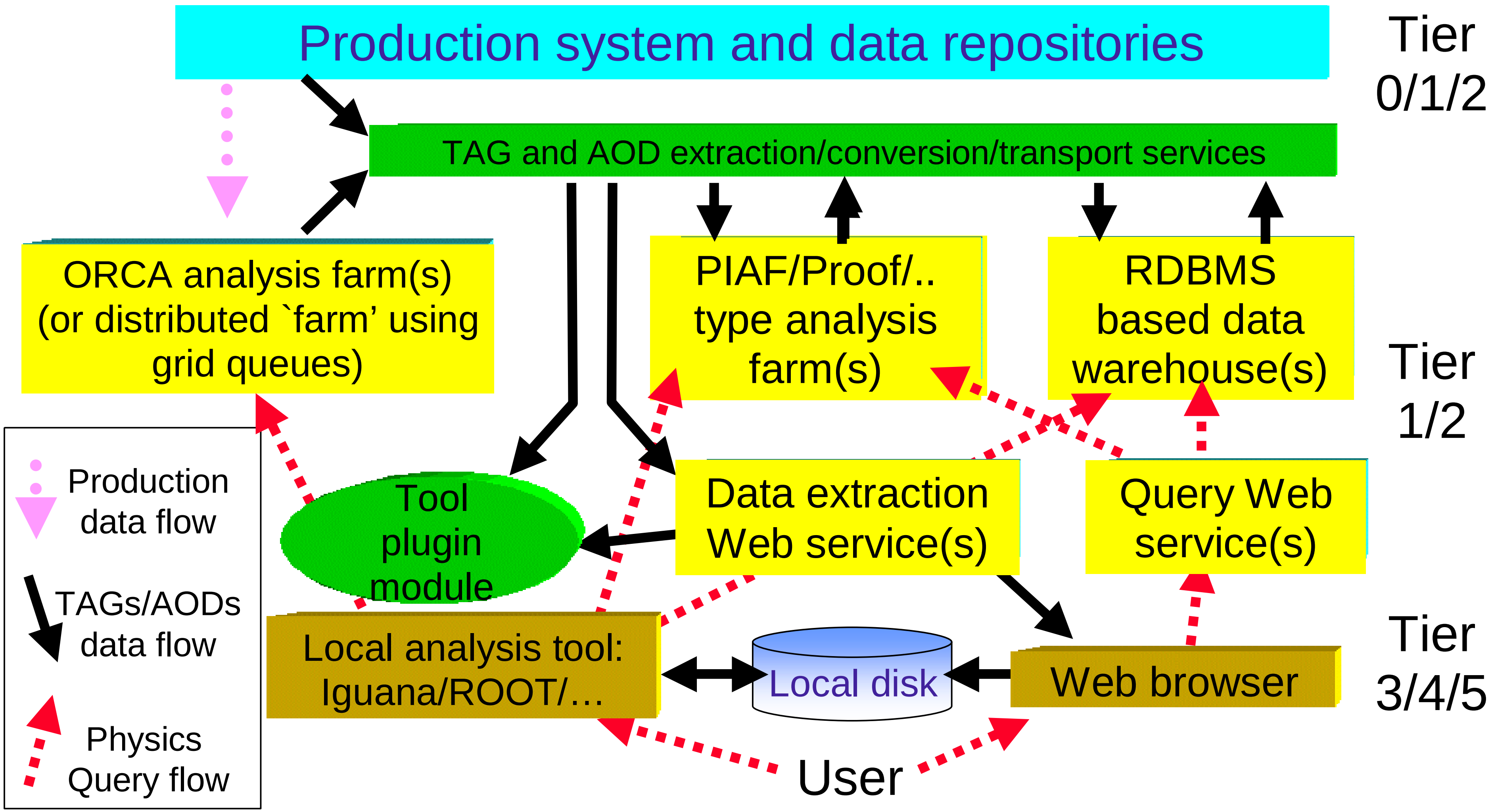
Joint ATLAS and LHCb Project



Based on the concept of Python bus:

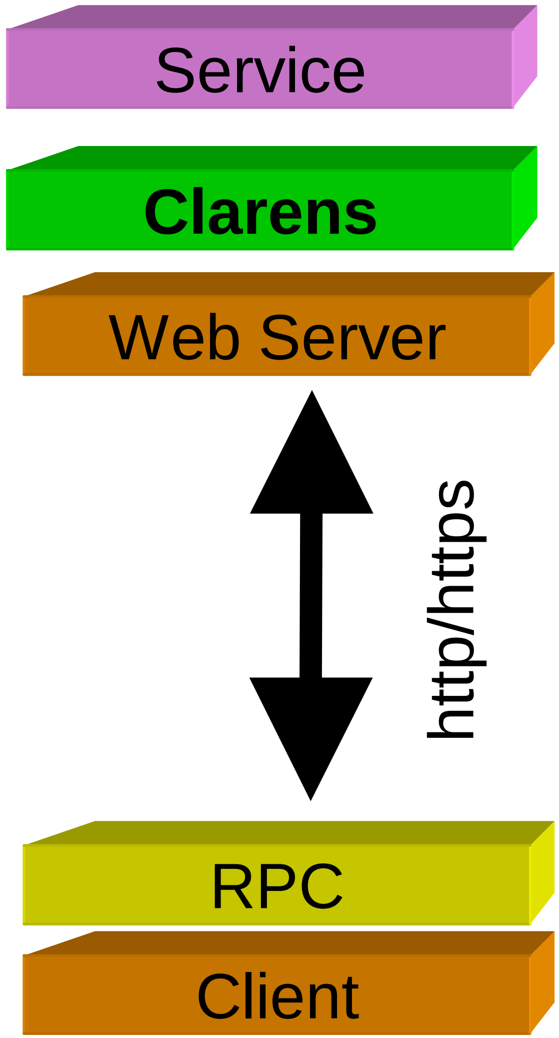
use different modules whichever are required to provide full functionality of the interface use Python to glue this modules, i.e., allow interaction and communication between them

CMS Components and Data Flow



Clarens - A CMS Grid Portal

- Grid-enabling the working environment for physicists' data analysis
- **Clarens** consists of a server communicating with various clients via the commodity XML-RPC protocol. This ensures implementation independence.
- The server will provide a remote API to Grid tools:
  - The Virtual Data Toolkit: Object collection access
  - Data movement between Tier centres using GSI-FTP
  - CMS analysis software (ORCA/COBRA),
  - Security services provided by the Grid (GSI)
  - No Globus needed on client side, only certificate

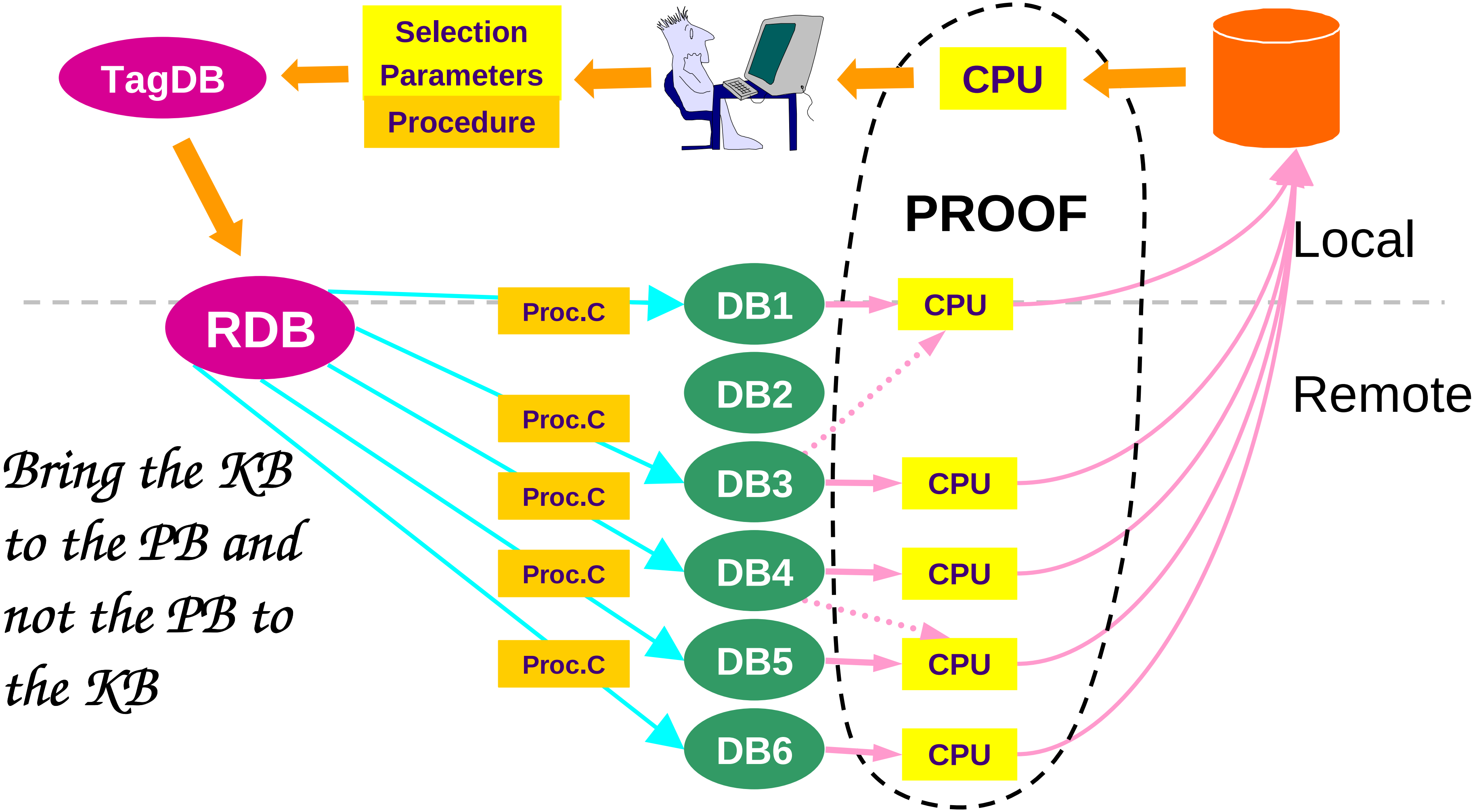


Current prototype is running on the Caltech proto-Tier2

PROOF – Parallel ROOT Facility

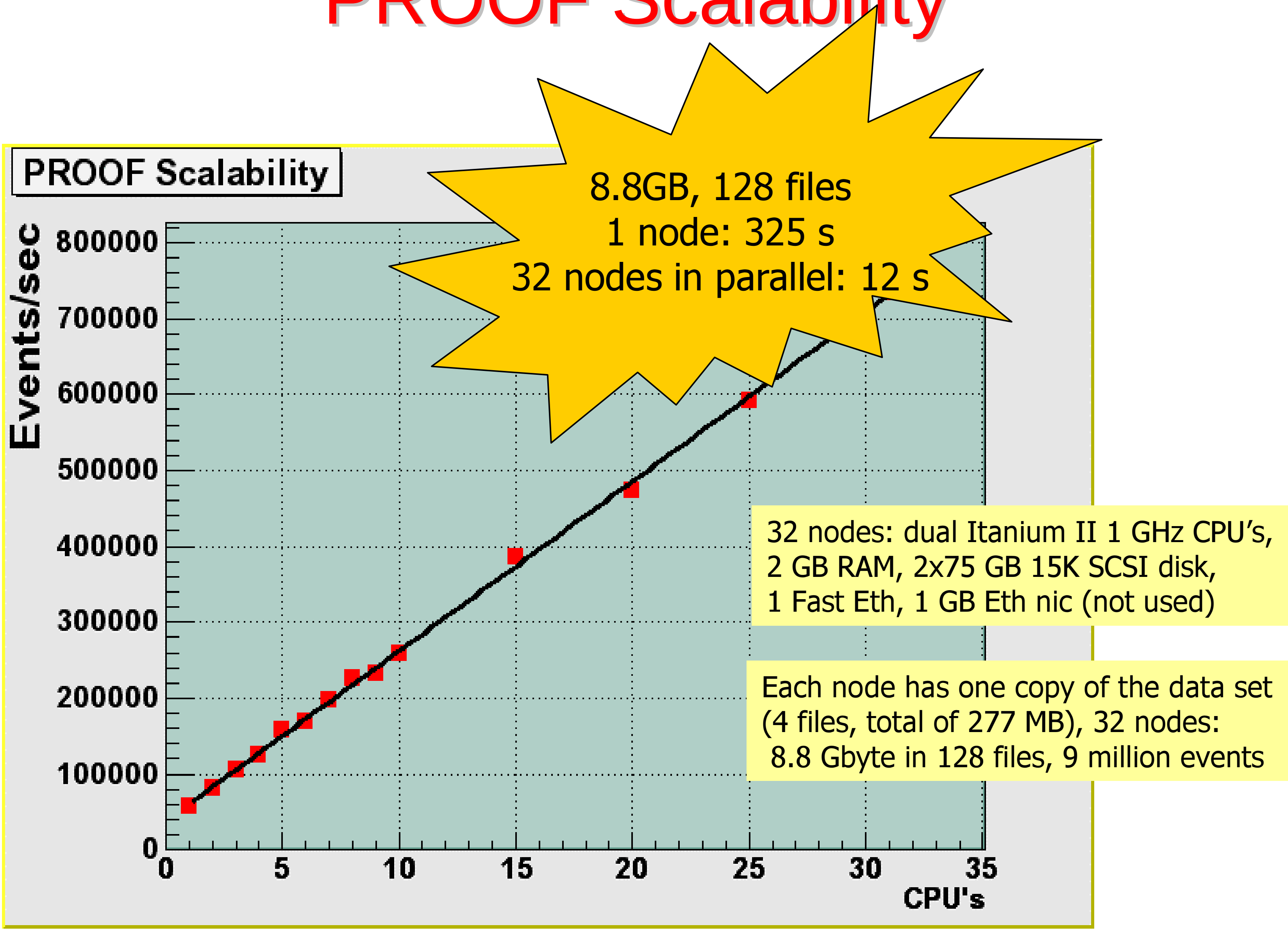
- PROOF is a system for the interactive analysis of very large sets of ROOT data files on a cluster of computers
- The main idea is to speed up the query processing by employing parallelism
- In the GRID context, this model will be extended from a local cluster to a wide area "virtual cluster". The emphasis in that case is not so much on interactive response as on transparency
- With a single query, a user can analyze a globally distributed data set and get back a "single" result
- The main design goals are:
  - transparency, scalability, adaptability

PROOF



Bring the KB to the PB and not the PB to the KB

PROOF Scalability



Conclusions

- Analyzing the Peta Bytes of LHC data is an enormous challenge and requires a large software and hardware infrastructure
- Modern OO software techniques are used to build the offline analysis systems
- Data processing will be distributed world wide, not a single site is large enough to do it alone
- The GRID paradigm fits neatly into what we are trying to do and we plan to make maximum use of it (and help develop it along the way)
- To make sure the analysis chain is up to the task we regularly run Data Challenges to test the capabilities of the system