ALICE Computing Status
Are we ready?
What about our choices?

Workshop on LHC Computing
26 October

René Brun
CERN

Several slides of this talk are taken from

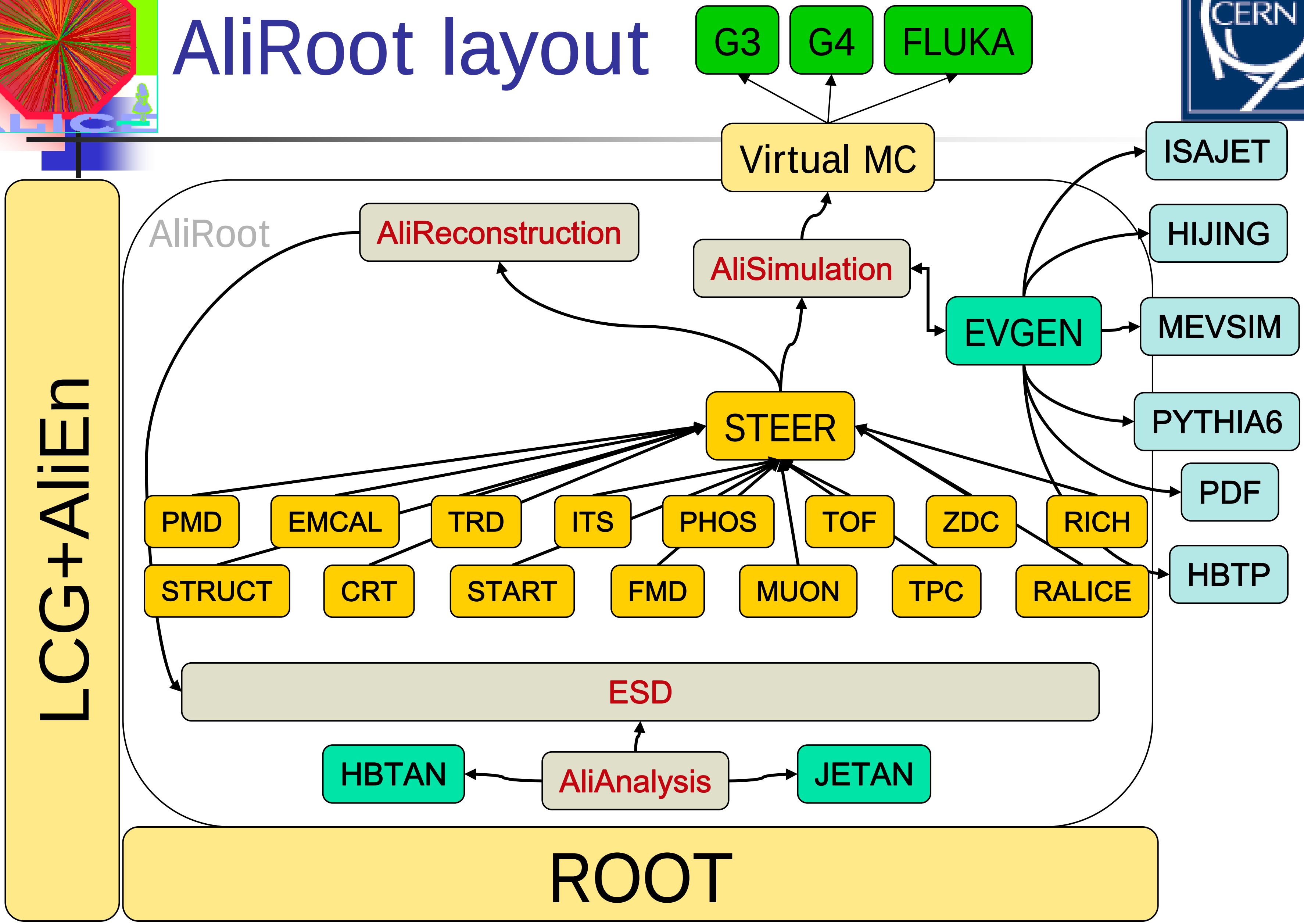
Federico Carminati's presentation at the ROOT workshop

http://root.cern.ch/root/R2005/Welcome.html

Software choices

- 1993->1997 : gAlice (f77 + zebra + G3)
- 1998 : ROOT as a framework
  - All classes have a dictionary
  - Can be made persistent
  - Callable from CINT and ACLIC
- 2001 : Alien for the GRID
- 2001 : Data Challenges DAQ ->MSS
- 2005 : complete chain DAQ/SIMU ->REC->ANA, PROOF for interactive Analysis
- 2006 : consolidation

AliRoot layout



I/O challenge in Alice (1)

- Because of the expected huge data volume, great care has been put in the design of the event model (from RAW to AOD). We had many iterations, eliminating unnecessary layers and optimizing the speed for data analysis.
- We do not use fancy object models. No complex class hierarchy. No templated classes.
- Our classes have all a dictionary and can be made persistent or callable by CINT/ACLIC.
- Object collections designed to minimize memory fractioning with new/delete and be easily usable at the AOD level.

I/O challenge in Alice (2)

- Following the great work in BaBar and Star, we came to the conclusion that load balancing and robustness is vital when processing many jobs in parallel.
- Simply calling rfio, castor, dcache services is just not enough.
- xrootd is a key element in the scenario.
- But still a lot of work to do to have a seamless integration with local services (castor, dcache,..)
- One thing is to optimize I/O within one job, another most important is to optimize the global performance for 1000 jobs (batch or interactive) running in parallel.

I/O challenge in Alice (3)

- We have excluded RDBMS from the picture for bulk data processing. RDBMS are only used where they make sense (online and locking services).
- RDBMS are far too slow, subject to bottlenecks and totally inappropriate to manage Terabytes in a distributed environment.
- We exploit read only data sets as much as possible using load-balanced xrootd servers. This is much better and scalable than adding more processors to a saturated DB server.
- Currently we concentrate our work in speeding up the queries to Trees for interactive analysis with PROOF.
- Our analysis code is gradually upgraded to run with PROOF selectors. Parallelism will play an increasing role.

ALICE event model

- RAW, ESD, AOD, CONDB are all ROOT files managed by a well tuned Alien FC.
- Bulk data in ROOT Trees
- Other data in keyed objects
- AA mode : 1 or few events per file (1.5 GBytes)
- pp mode : several events (>100) per file
- Tree split mode used as much as possible
  - Improve read speed
  - Improve compression

Important rules

- All Alice files can be read via:
- ROOT alone (useful to do quick histogramming and selections)
- ROOT + a small subset of AliRoot shared libs (eg libAliESD). This is the privileged model when running interactive analysis with PROOF.
- The full AliRoot

AliRoot: Current Status

- Up-to-date description of ALICE detectors
  - TGeo
- Rich set of event generators, easily extensible
- Possibility to use different transport packages
  - VMC
- User friendly steering classes for simulation and reconstruction
- Efficient track reconstruction
- Combined PID based on Bayesian approach
- ESD classes for analysis and fine-tune calibration
- Analysis examples to explore wide spectrum of heavy-ion and pp physics

Software summary table

Address: http://agenda.cern.ch/askArchive.php?base=agenda&categ=a056774&id=a056774s1t2/moreinfo

Detector's status: CVS HEAD 20/10/2005

	Geom.	Mat.	Hits	RAW	SDigits	Digits	Rec. points	Final rec.	PID	ESD	Alignment	Calibration	IO	Libs	Doc	Coding
ITS	84 overlaps >0.1, new with TGeo?	fract. Z	OK	OK	OK	OK	clusters V2 vs rec. points	Check tracker SA	OK	OK	No	Only SPD	OK	OK		
TPC	2 overlaps >0.1	OK	Fluka?	OK	OK	OK	OK	OK	OK	OK	first attemt	No	OK	OK		
TRD	OK	OK	Fluka?	OK	OK	OK	OK	OK	OK	OK	No	first classes	OK	OK		
TOF	4 overlaps >0.1	OK	OK	OK	OK	OK	OK	OK	OK	OK	No	No	OK	OK		
RICH	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK?	No	No	OK	OK		
PHOS	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	No	first classes	Getter	no split		
EMCAL	not initialized parameters in rec.	OK	OK	No rec.	OK	OK	OK	OK?	OK?	OK?	No	No	Getter	OK		
FMD	OK	OK	Too long E-tail?	OK	OK	OK	OK	OK	-	-	No	No	OK	OK		
PMD	OK	OK	OK	OK	OK	OK	OK	OK	OK?	OK?	No	No	OK	OK		
START	6 overlaps >0.1	OK	OK	OK	OK	OK	OK	OK	-	OK	No	No	OK	OK		
VZERO	3 overlaps >0.1	OK	OK	OK	OK	OK	OK	OK	-	-	No	No	OK	OK		
MUON	222 overlaps >0.1 St2 to be fixed	OK	G3 info	OK	OK	OK	OK	OK	OK	OK	first classes	No	OK	OK		
ZDC	OK	OK	OK	OK	OK	OK	OK	OK	-	OK	No	OK	OK	OK		
CRT	OK	OK	OK	-	-	-	-	-	-	-	No	No	-	no split		

Alignment requirements:

1. Global access to the alignment parameters (rotations and translations in TGeo form), and alignment functionality (global to local, local to global transformations)
2. Unified class for stotrage of "spatial" alignment parameters

ESD requirements:

1. Meaningful run and event number
2. Error matrix for the impact parameters
3. Storage of track parameters in AliExternalTrackParameter

Many analysis examples

- **HBT analysis**
- **Jet analysis**
- **V0's and cascades**
- **D0->Kpi analysis**
- **Balance function analysis**
- **Tag DB classes**
- **Flow analysis**
- **Analysis of resonances**
- **Multiplicity analysis**
- **D+ in PbPb, B->eX in pp, etc..**

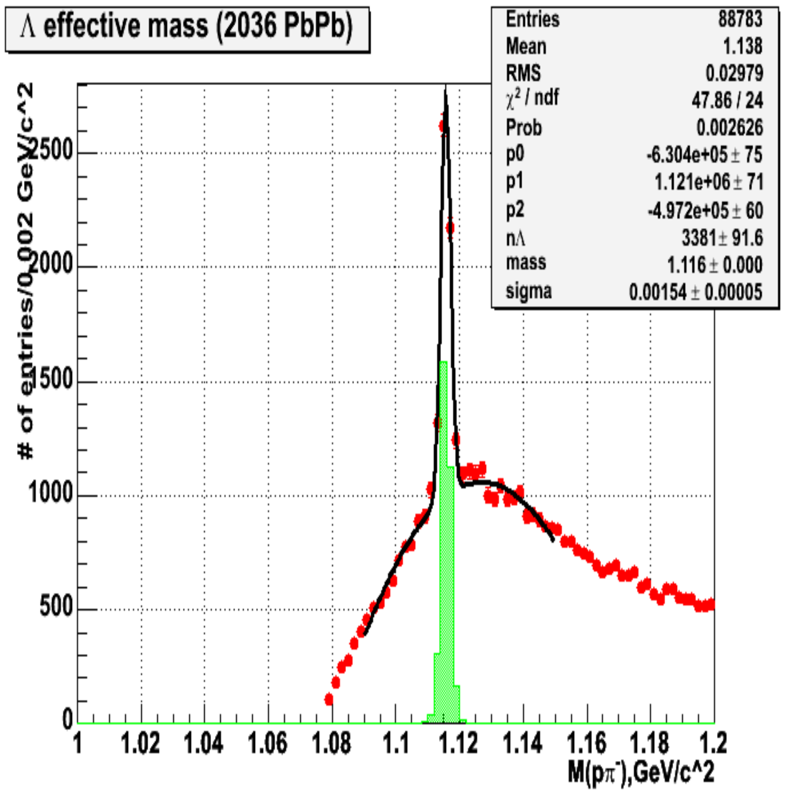
Most examples are simple (< 500 lines)
A few are complex (HBT,Jetan)
They produces histograms and nice plots

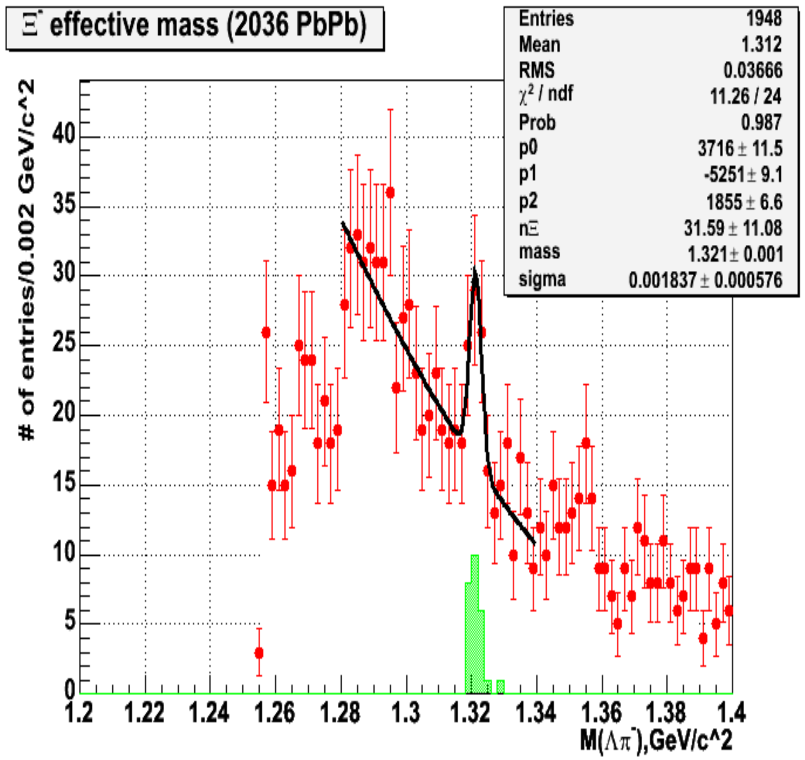
CODE EXAMPLE

```
void demo(const char* user = "pchrista")
{
    //Create an AliTagAnalysis object and connect to AliEn's API services
    AliTagAnalysis TagAna ("aliendb4.cern.ch",9000,"pchrista");
    //Create an AliEventTagCuts object and impose some selection criteria
    AliEventTagCuts EvCuts;
    EvCuts.SetMultiplicityRange(0,500);

    //Query tag data from the catalogue
    TGridResult *TagResult =
    gGrid->Query("/alice/cern.ch/user/p/pchrista/PDC05/pp/Tags1","*tag.root","","-l 10 –Opublicaccess=");

    //Chain the tag files
    TagAna.ChainGridTags(TagResult);
    //Get file info list
    Tlist *resultlist = TagAna.QueryTags(EvCuts);
    //Create a chain
    TChain chain("esdTree");
    //append the resultlist to the chain
    chain.AddFileInfoList(resultlist,100);
    //process the selector
    chain.Process("esdTree.C");
}
```

Can be run with normal ROOT or PROOF

$\Xi \rightarrow \pi\Lambda \rightarrow p\pi$

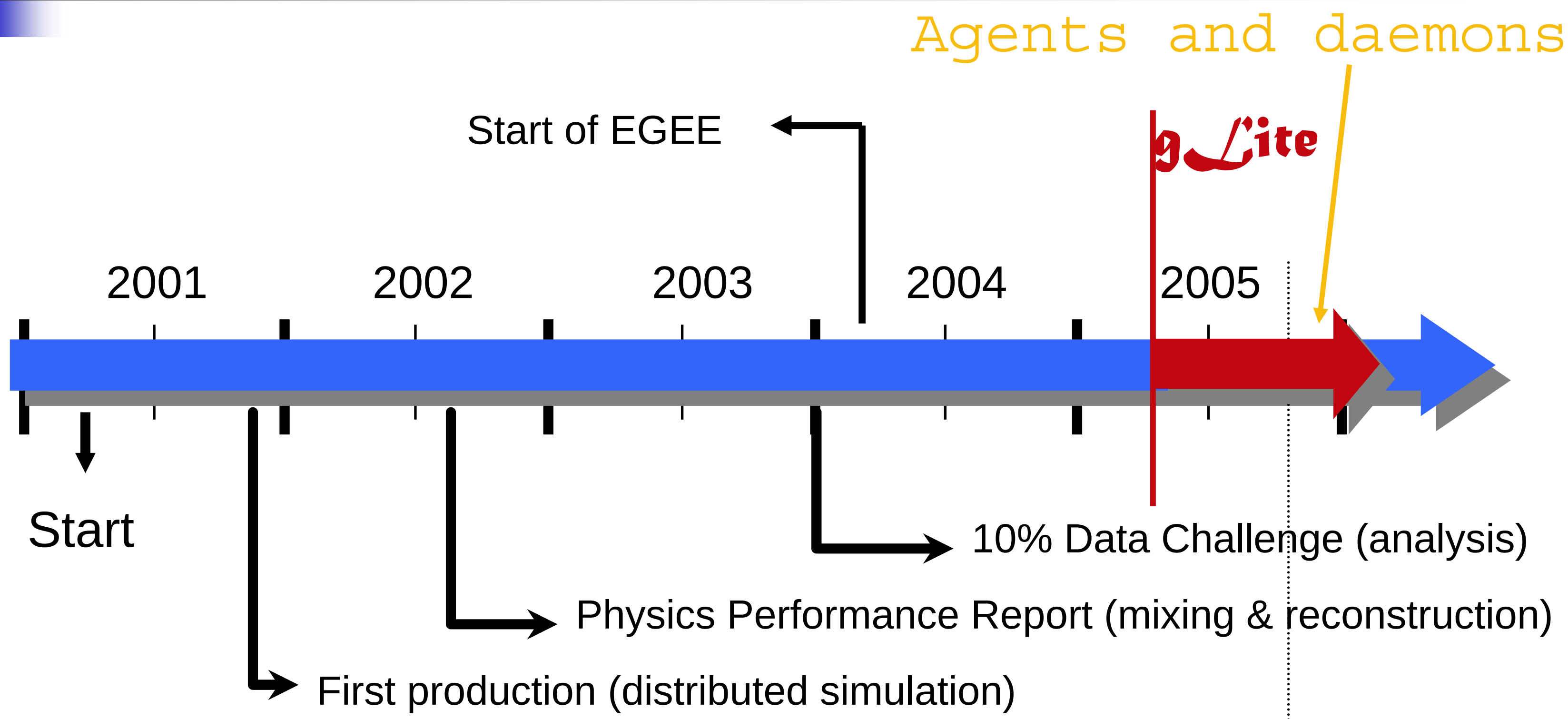




ALICE Analysis Basic Concepts

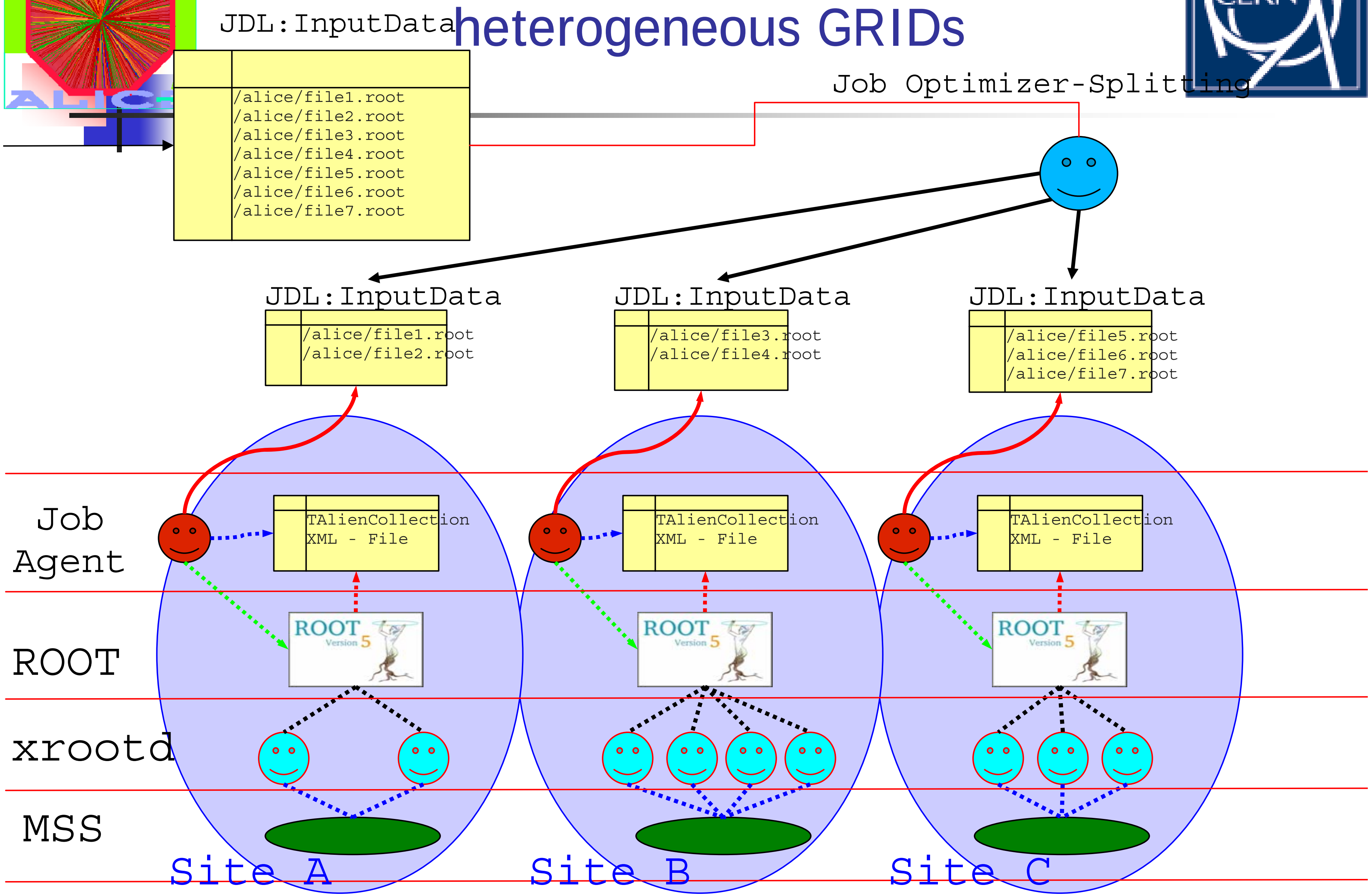
- Analysis Models
  - Prompt analysis at T0 using PROOF(+file catalogue) infrastructure
  - Batch Analysis using GRID infrastructure
  - Interactive Analysis using PROOF(+GRID) infrastructure
- User Interface
  - ALICE User access any GRID Infrastructure via AliEn or ROOT/PROOF UIs
- AliEn
  - Native and “GRID on a GRID” (LCG/EGEE, ARC, OSG)
  - integrate as much as possible common components
    - LFC, FTS, WMS, MonALISA ...
- PROOF/ROOT
  - single- + multitier static and dynamic PROOF cluster
  - GRID API class TGrid(virtual)->TAliEn(real)

The AliEn timeline

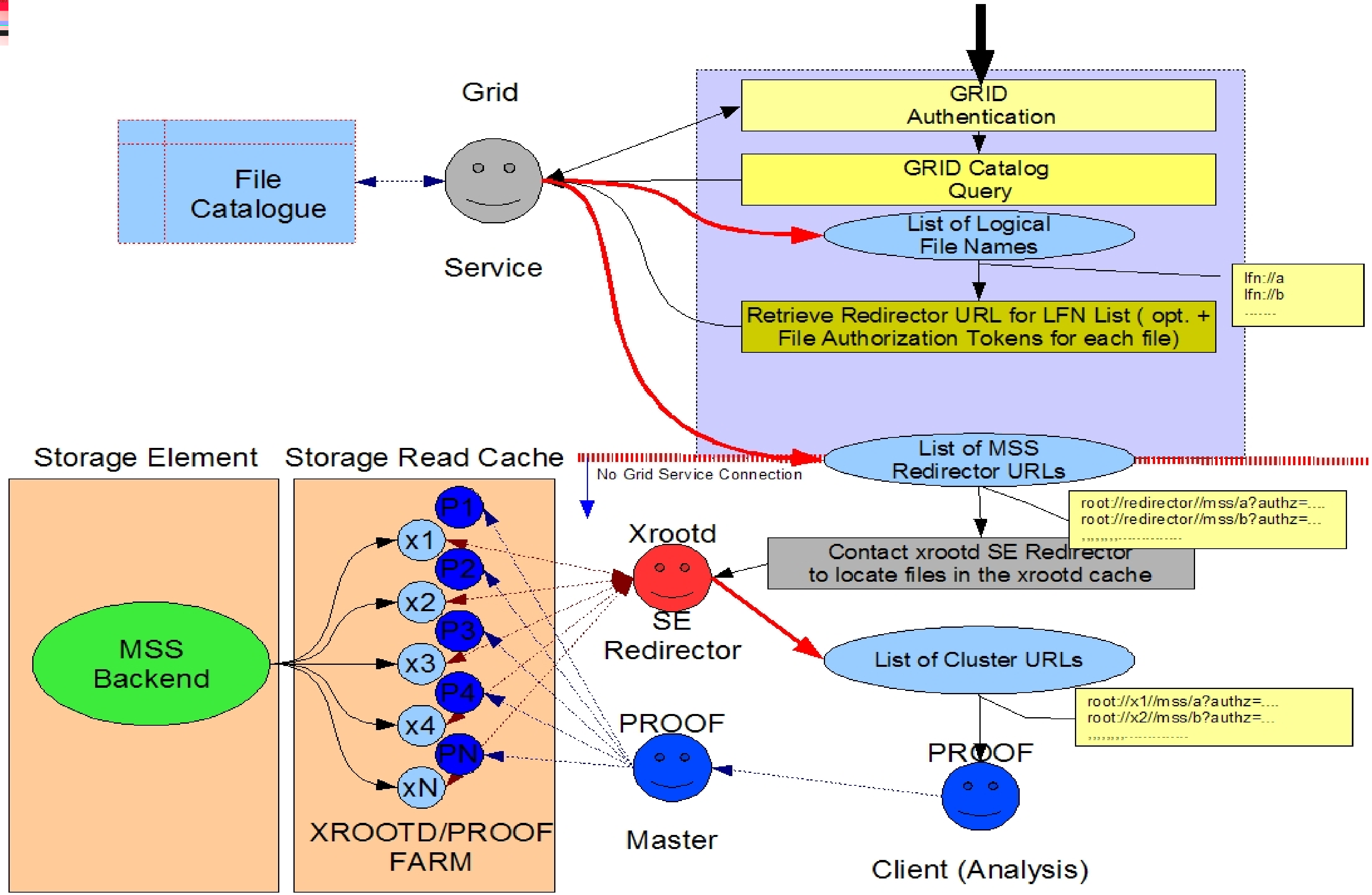


Functionality + Simulation → Interoperability + Reconstruction → Performance, Scalability, Standards + Analysis

ALICE Batch Analysis via agents in heterogeneous GRIDs



PROOF@GRID: 1-Cluster Setup



Our good choices (1)

- Capitalize on ROOT + Alien framework
- We understood early (1997) the importance of dictionaries
  - For I/O AND scripting
- CINT used for configuration scripts since day1
  - Our scripts can be interpreted (CINT) or compiled (ACLIC)
- CINT used for data analysis. No need for Python. Scripts can be interpreted or compiled via ACLIC. Only one efficient language.

Our good choices (2)

- Minimize systems dependencies
  - We do not use SEAL, POOL, COOL, CLHEP, Boost
  - We do not feel the need for stuff like CMT or SCRAM
- Simple build system with Makefiles
- Target portability (AliRoot running on all OS)
- Software and Physics groups are the same
- Coherent pipeline from DAQ to REC and ANA
- Code checking and profiling utilities

What could have been better

- More training with collections and Tree design. Most people learnt by copying examples.
- Better documentation of our system
- More dialog with other experiments in 2000, but probably a common framework would not have been possible.
- Avoid polemics with a some committees