Introduction to ROOT

Summer Students Lecture

21 July 2004

René.Brun@cern.ch

`ftp://root.cern.ch/root/SummerStudents2004.ppt`

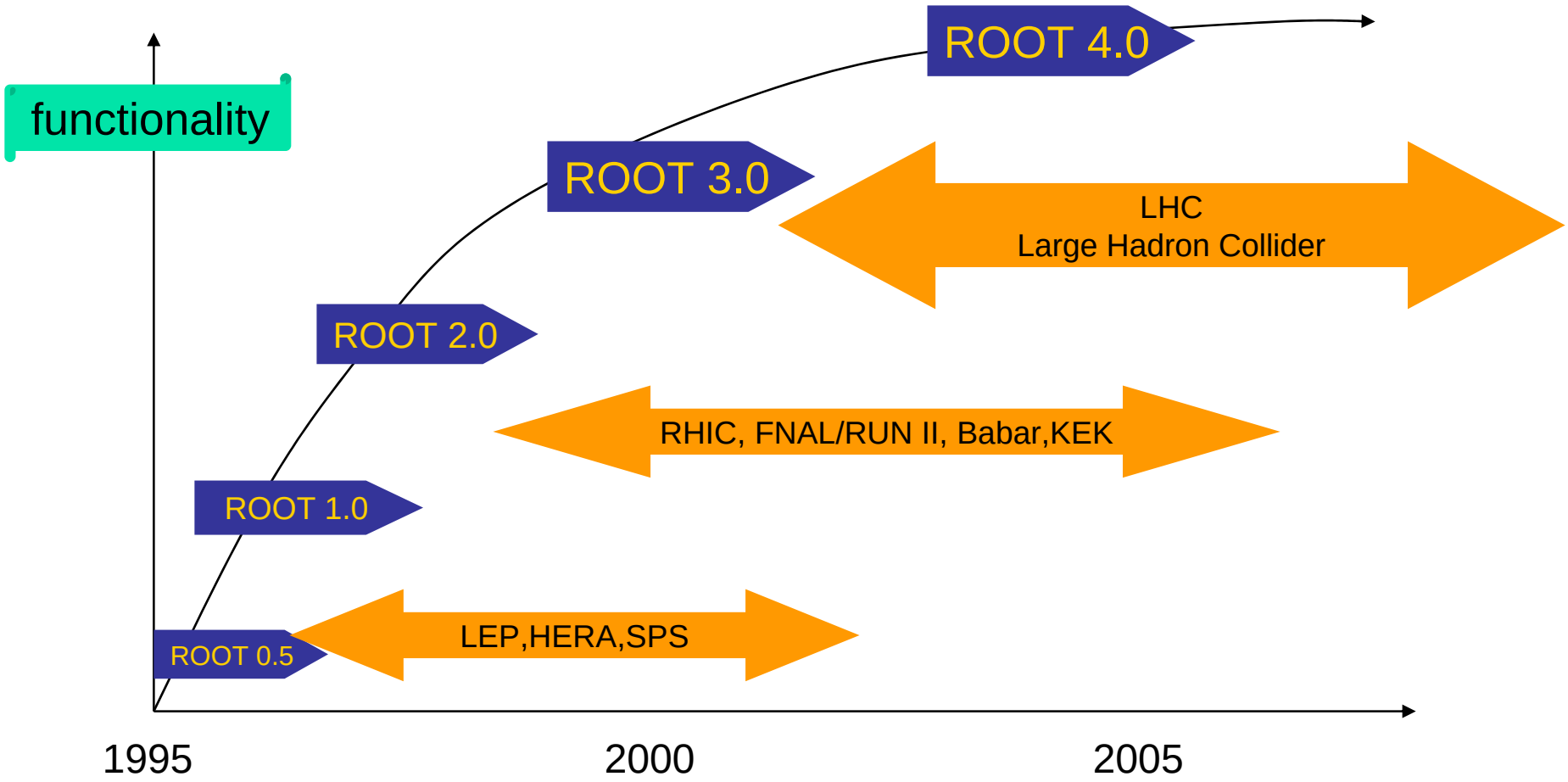
ROOT in a nutshell

- An efficient data storage and access system designed to support structured data sets in very large distributed data bases (Petabytes).
- A query system to extract information from these distributed data sets.
- The query system is able to use transparently parallel systems on the GRID (PROOF).
- A scientific visualisation system with 2-D and 3-D graphics.
- An advanced Graphical User Interface
- A C++ interpreter allowing calls to user defined classes.
- An Open Source Project

ROOT: An Open Source Project

- The project is developed as a collaboration between :
- Full time developers:
  - 6 people full time at CERN
  - 1 key developer at FermiLab
  - 1 key developer in Japan (Agilent Technologies)
  - 1 key developer at MIT
  - 1 mathematician at CERN sponsored by a US Finance Company
- Many contributors spending a substantial fraction of their time in specific areas (> 50).
- Key developers in large experiments using ROOT as a framework.
- Several thousand users given feedback and a very long list of small contributions.

The ROOT Project

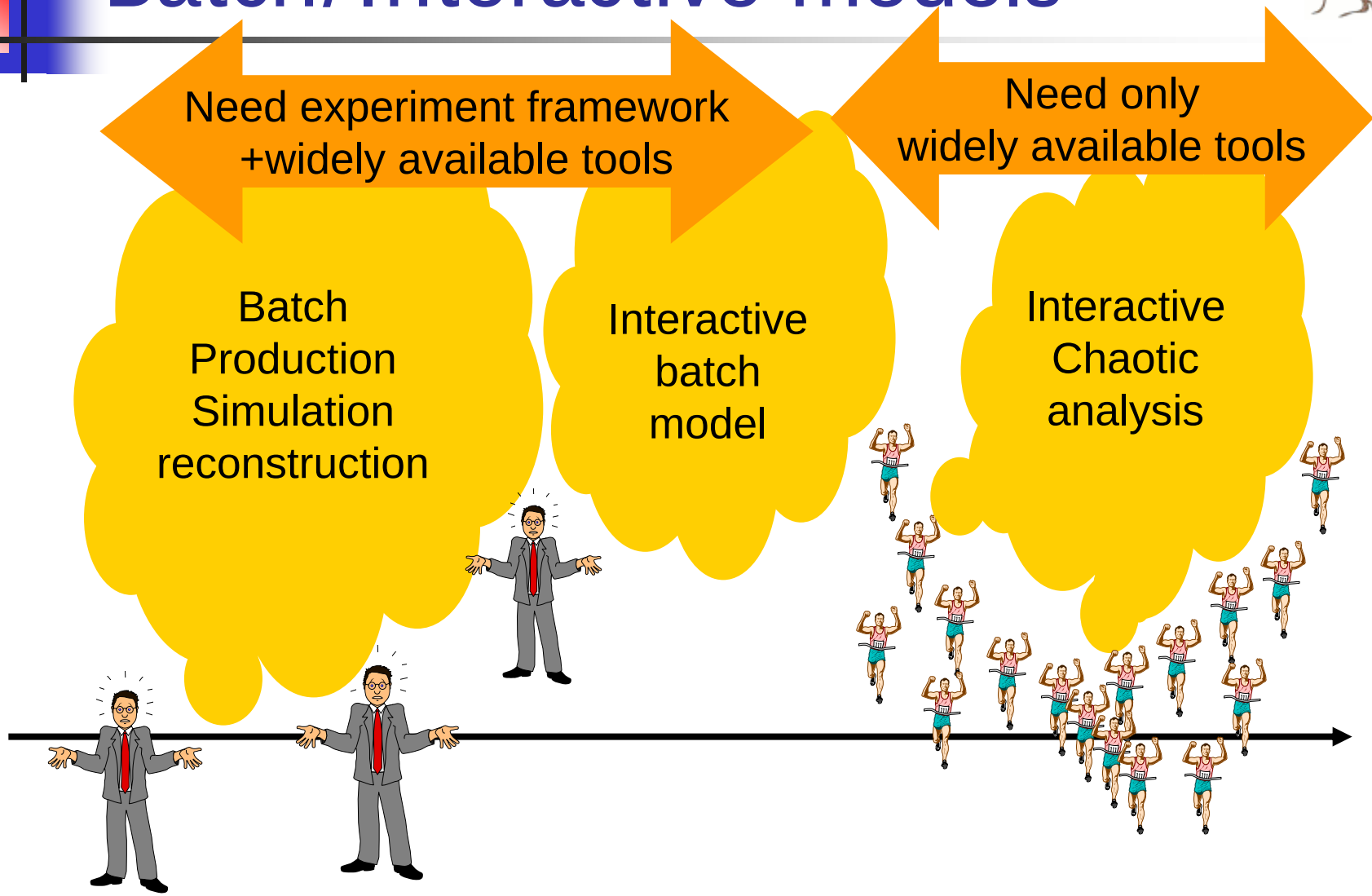


The ROOT web pages

http://root.cern.ch

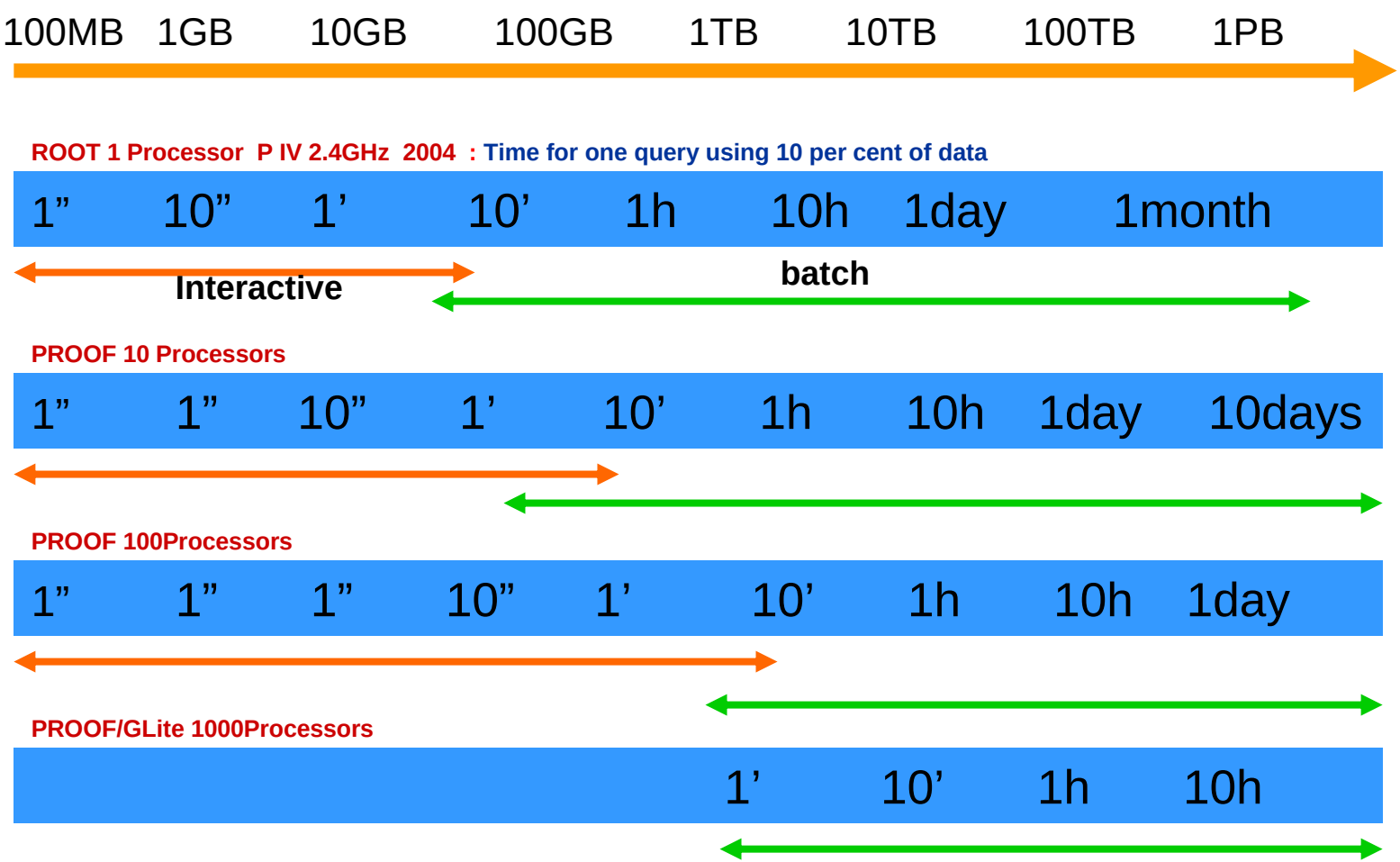
- General Information and News
- Download source and binaries
- Howto & tutorials
- User Guide & Reference Guides
- Roottalk Digest & Forum

Batch/Interactive models



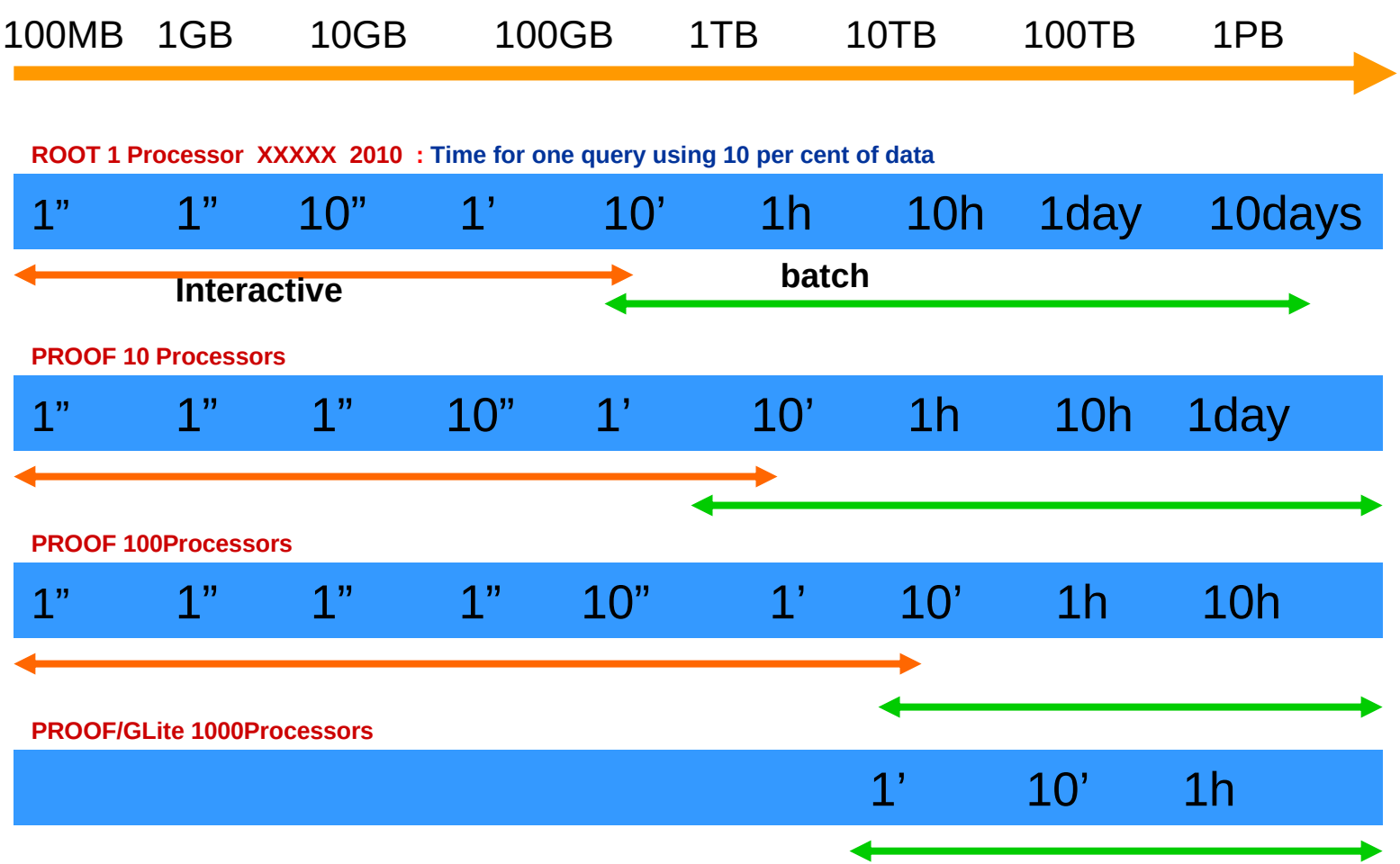
Data Volume & Processing Time

Using technology available in 2004



Data Volume & Processing Time

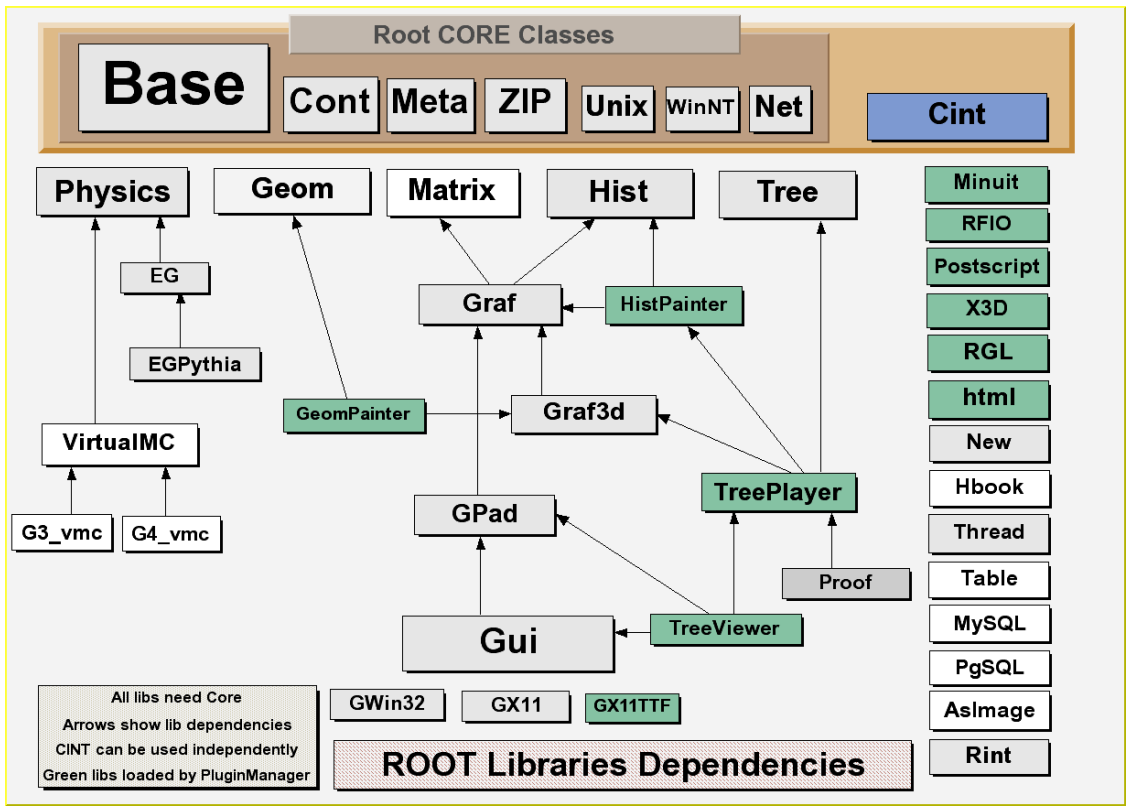
Using technology available in 2010



ROOT Library Structure

- ROOT libraries are a layered structure
- The CORE classes are always required (support for RTTI, basic I/O and interpreter)
- The optional libraries (you load only what you use) Separation between data objects and the high level classes acting on these objects. Example, a batch job uses only the histogram library, no need to link histogram painter library.
- Shared libraries reduce the application link time
- Shared libraries reduce the application size
- ROOT shared libraries can be used with other class libraries

The Libraries



- Over 1000 classes
- 1250,000 lines of code
- CORE (12 Mbytes)
- CINT (3 Mbytes)
- Green libraries linked on demand via plug-in manager (only a subset shown)

ROOT: a Framework and a Library

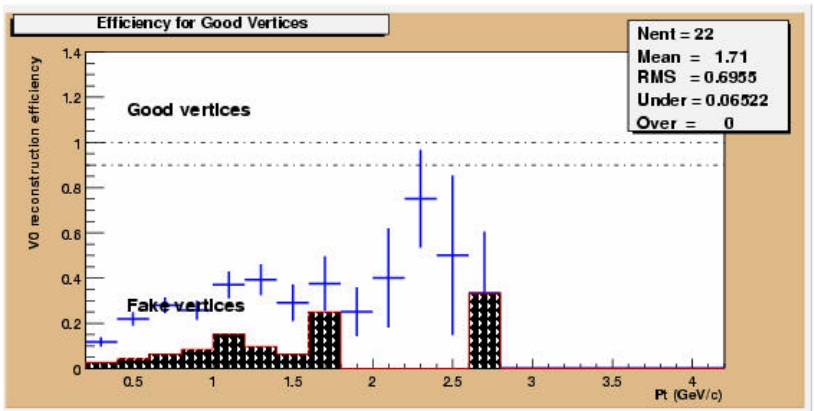
- User classes
  - User can define new classes interactively
  - Either using calling API or sub-classing API
  - These classes can inherit from ROOT classes
- Dynamic linking
  - Interpreted code can call compiled code
  - Compiled code can call interpreted code
  - Macros can be dynamically compiled & linked

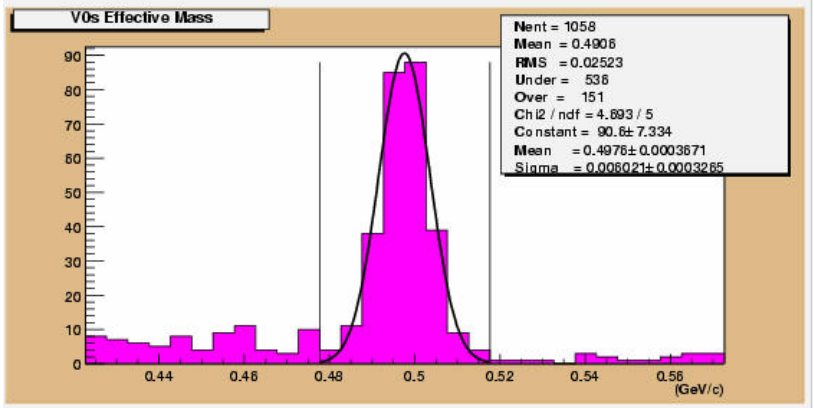
This is the normal operation mode

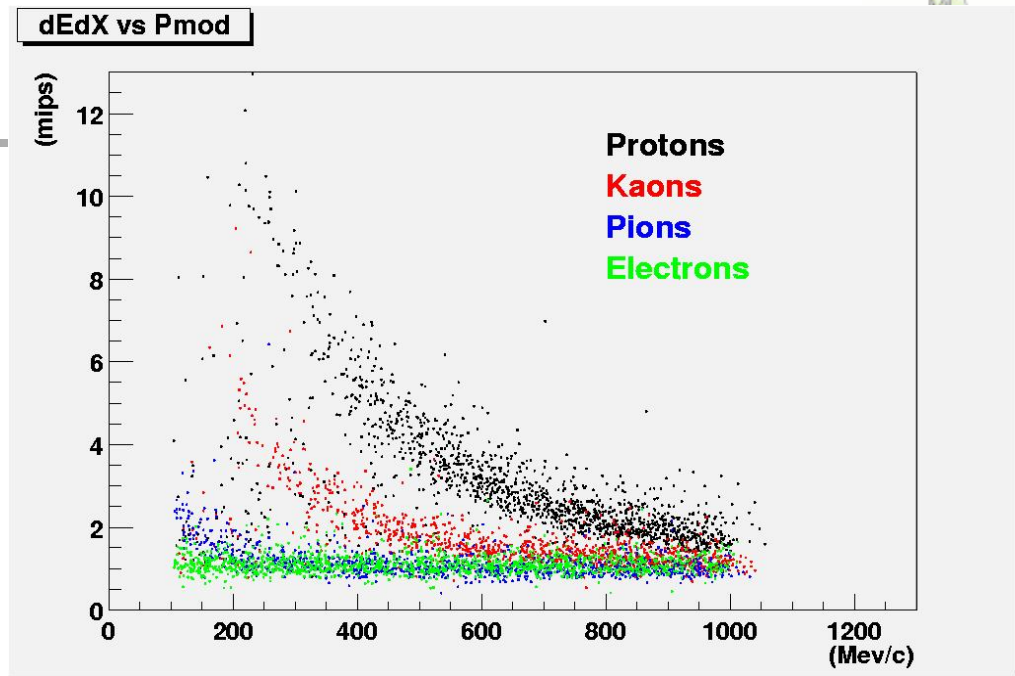
Interesting feature for GUIs & event displays

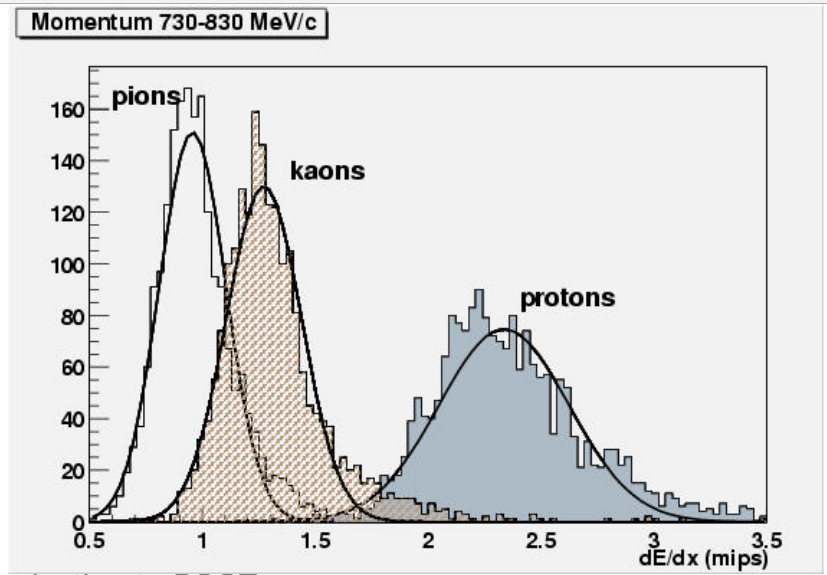
Script Compiler
root > .x file.C++

A Data Analysis & Visualisation tool

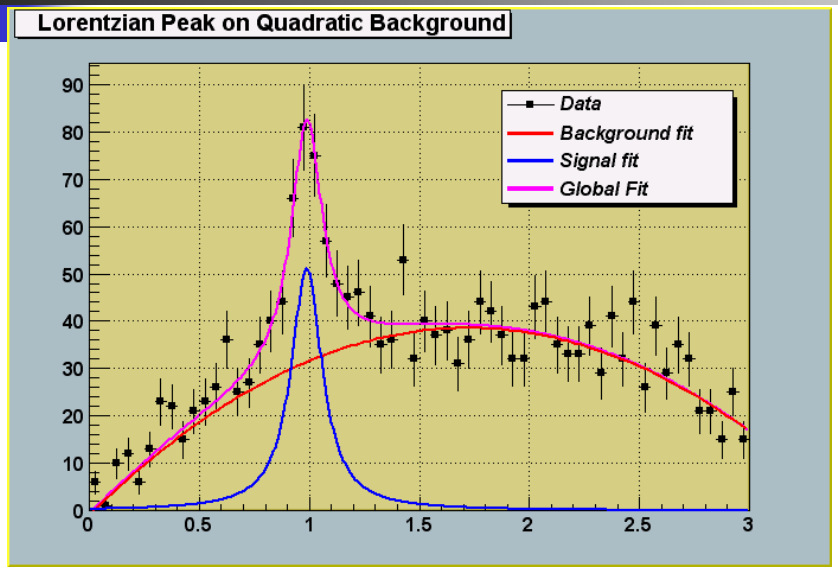


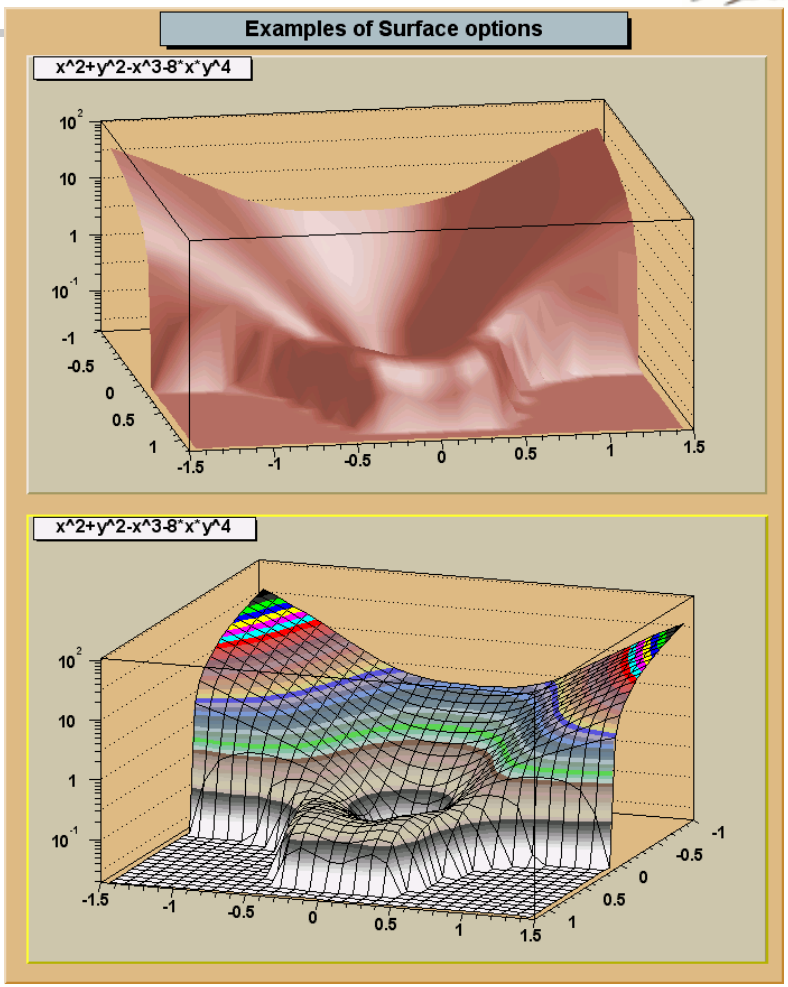


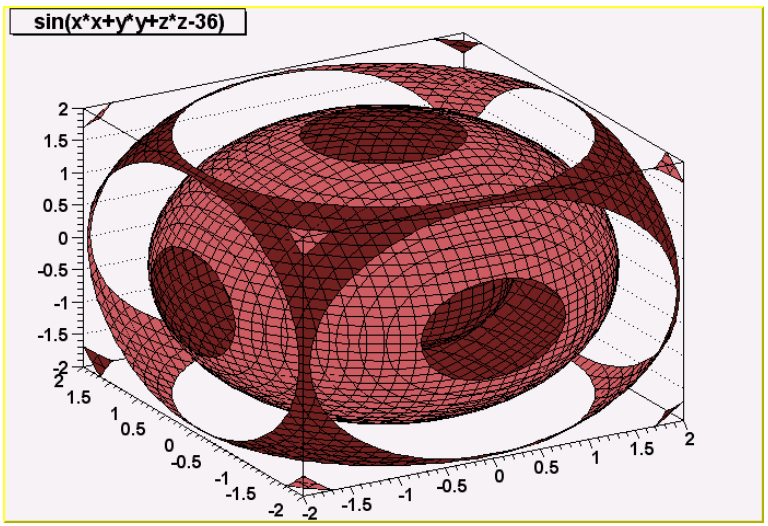




Graphics : 1,2,3-D functions







Full LateX support on screen and postscript

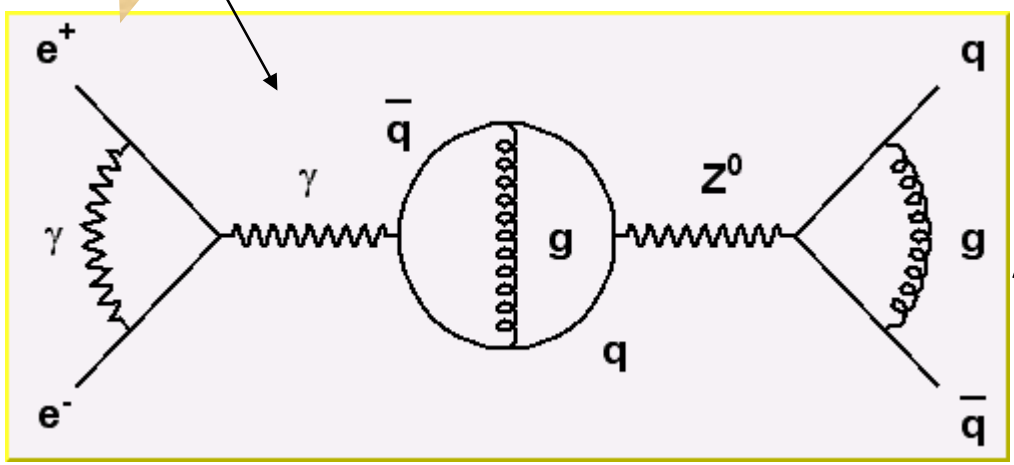
Born equation

$$\frac{2s}{\pi\alpha^2}\frac{d\sigma}{d\cos\theta}(e^+e^- \to f\bar{f}) = \left|\frac{1}{1-\Delta\alpha}\right|^2 (1+\cos^2\theta)$$

$$+ 4\,\mathrm{Re}\left\{\frac{2}{1-\Delta\alpha}\chi(s)\left[\tilde{g}_v^e\tilde{g}_v^f(1+\cos^2\theta) + 2\,\tilde{g}_a^e\tilde{g}_a^f\cos\theta\right]\right\}$$

$$+ 16\left|\chi(s)\right|^2\left[(\tilde{g}_a^{e\,2} + \tilde{g}_v^{e\,2})(\tilde{g}_a^{f\,2} + \tilde{g}_v^{f\,2})(1+\cos^2\theta) + 8\,\tilde{g}_a^e\tilde{g}_a^f\tilde{g}_v^e\tilde{g}_v^f\cos\theta\right]$$

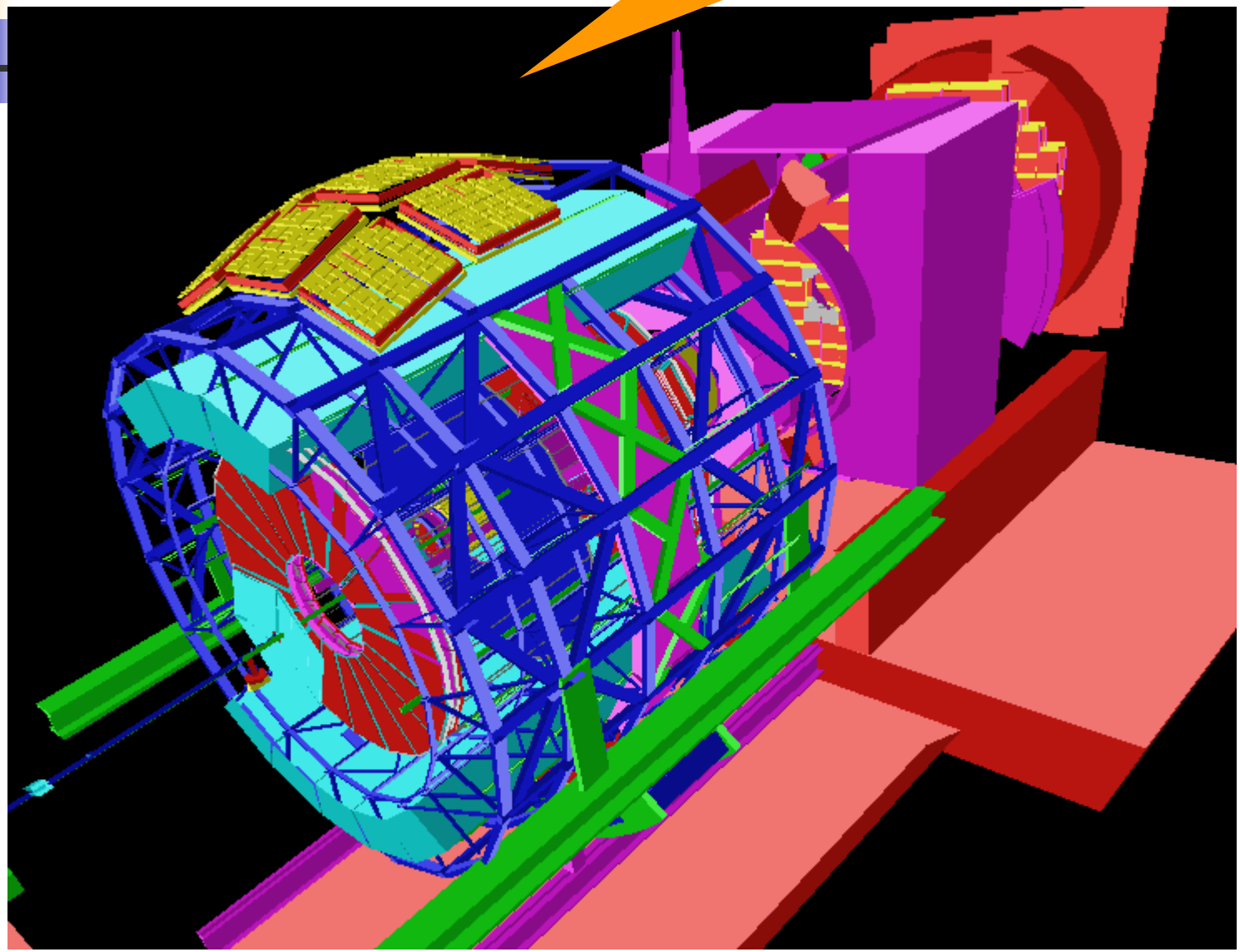
Formula or diagrams can be edited with the mouse



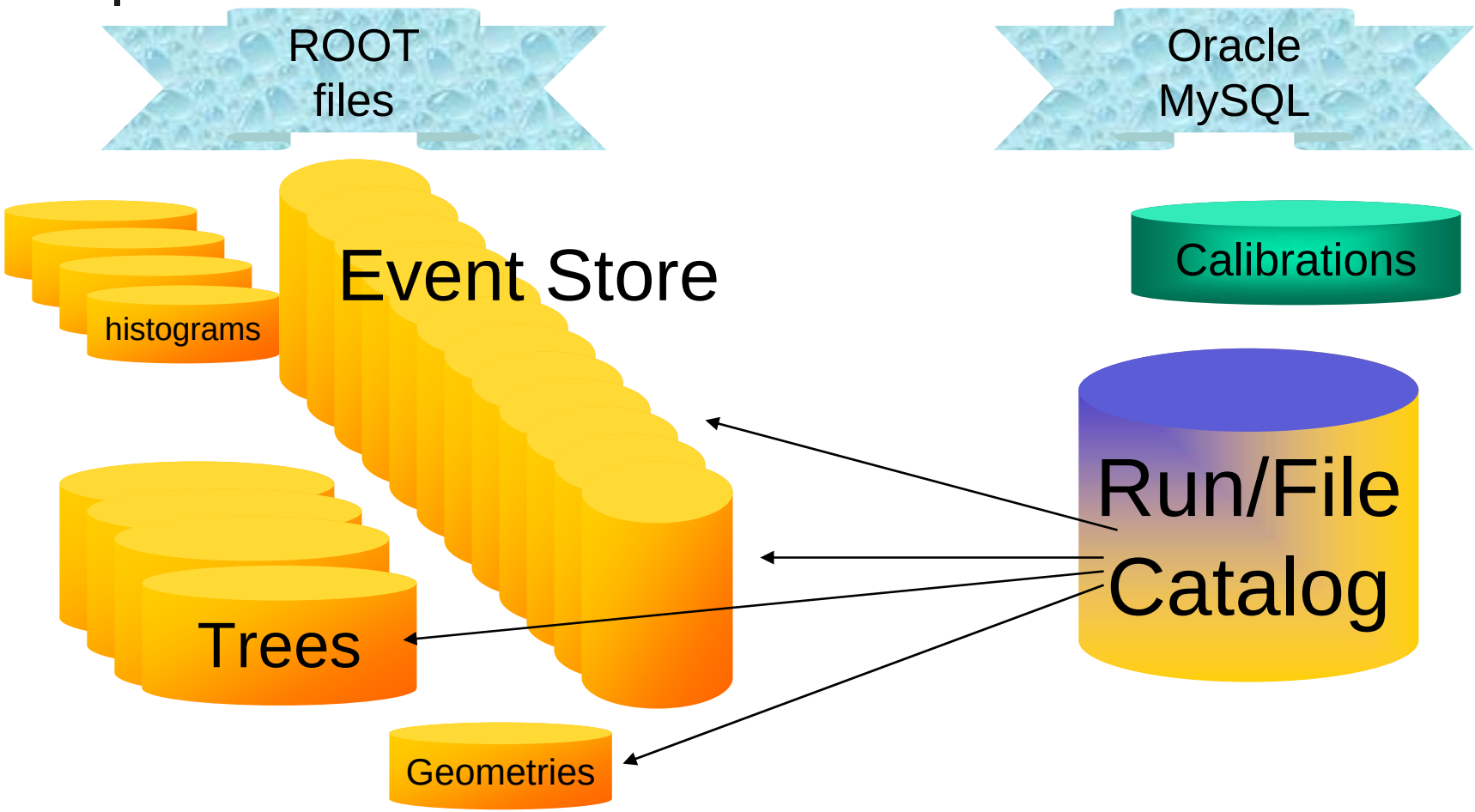
TCurlyArc
TCurlyLine
TWavyLine
and other building blocks for Feynmann diagrams

Alice

3 million nodes



ROOT + RDBMS Model



ROOT I/O : An Example

Program Writing

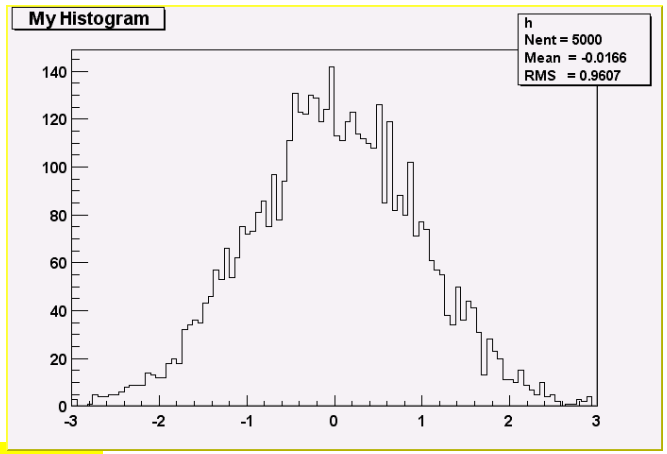
demoh.C

```
TFile f("example.root","new");
TH1F h("h","My histogram",100,-3,3);
h.FillRandom("gaus",5000);
h.Write();
```

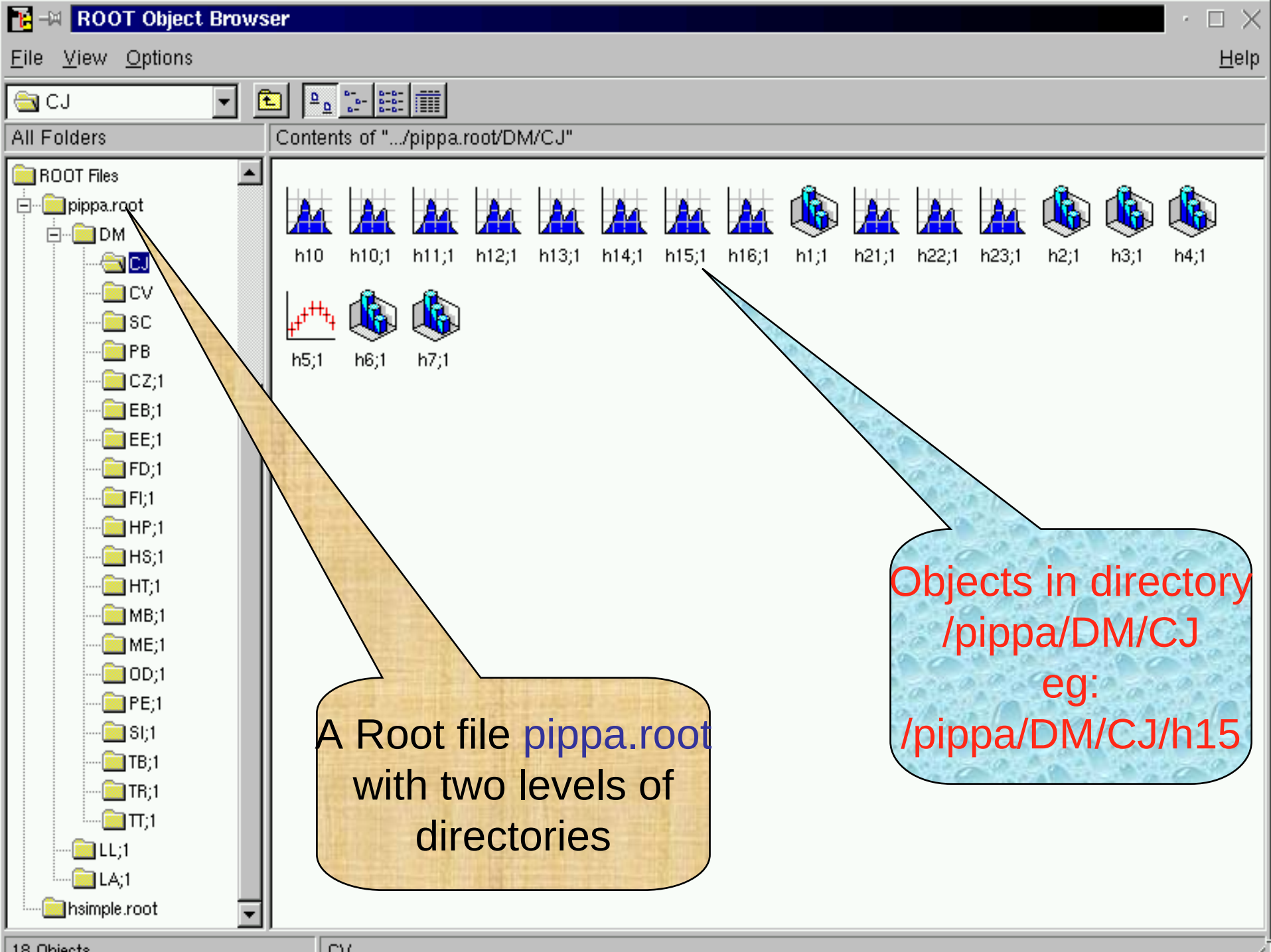
Program Reading

demohr.C

```
TFile f("example.root");
TH1F *h = (TH1F*)f.Get("h"):
h->Draw();
f.Map();
```

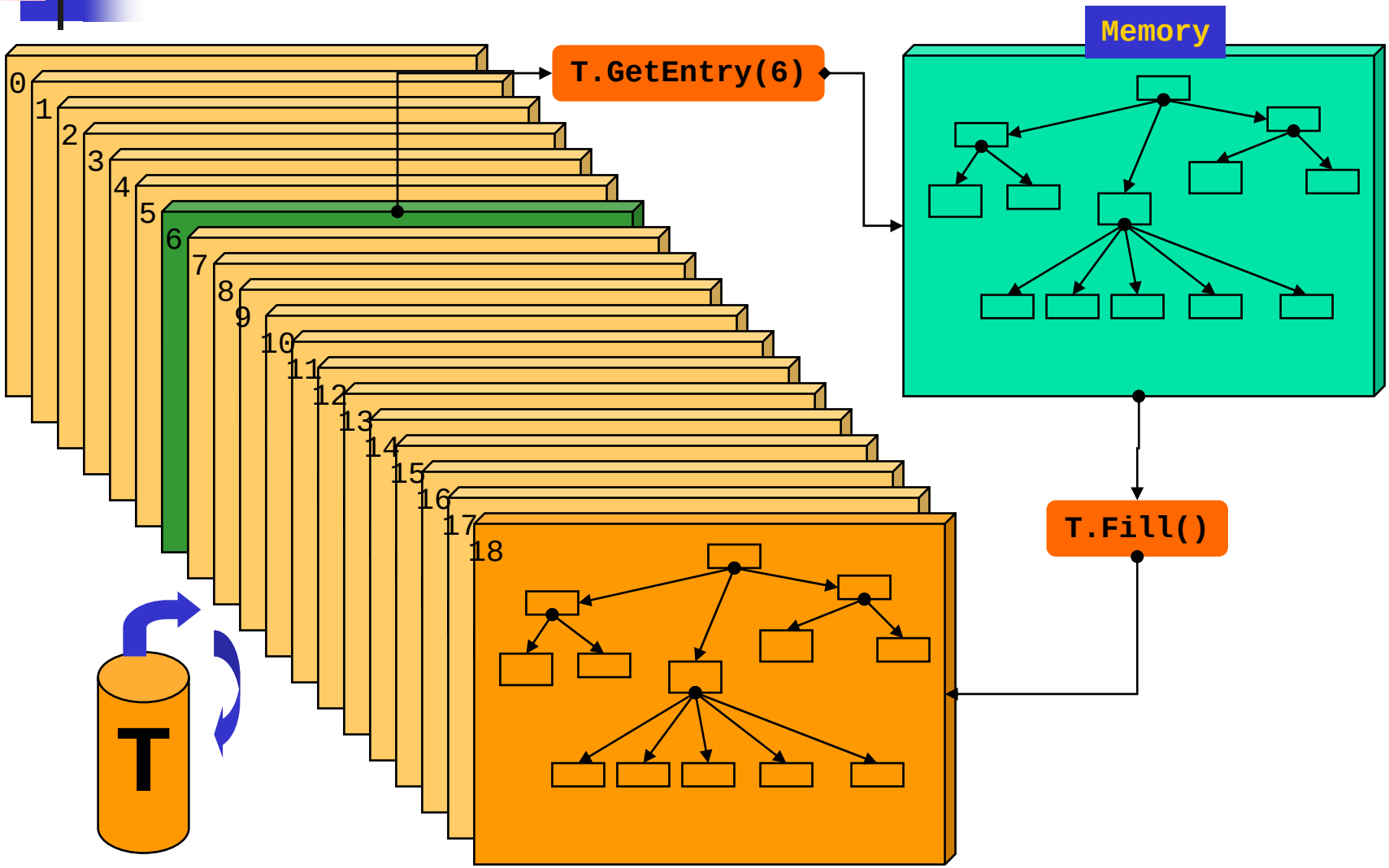



```
20010831/171903  At:64      N=90       TFile
20010831/171941  At:154     N=453      TH1F          CX =  2.09
20010831/171946  At:607     N=2364     StreamerInfo  CX =  3.25
20010831/171946  At:2971    N=96       KeysList
20010831/171946  At:3067    N=56       FreeSegments
20010831/171946  At:3123    N=1        END
```

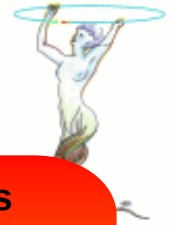

ROOT Object Browser
File View Options Help
CJ
All Folders
Contents of ".../pippa.root/DM/CJ"
ROOT Files
pippa.root
DM
CJ
CV
SC
PB
CZ;1
EB;1
EE;1
FD;1
FI;1
HP;1
HS;1
HT;1
MB;1
ME;1
OD;1
PE;1
SI;1
TB;1
TR;1
TT;1
LL;1
LA;1
hsimple.root
h10
h10;1
h11;1
h12;1
h13;1
h14;1
h15;1
h16;1
h1;1
h21;1
h22;1
h23;1
h2;1
h3;1
h4;1
h5;1
h6;1
h7;1
A Root file pippa.root with two levels of directories
Objects in directory /pippa/DM/CJ eg: /pippa/DM/CJ/h15

Memory <--> Tree

Each Node is a branch in the Tree



Tree example Event (write)



```
void demoe(int nevents) {
   //load shared lib with the Event class
   gSystem->Load("$ROOTSYS/test/libEvent");

   //create a new ROOT file
   TFile f("demoe.root","new");

   //Create a ROOT Tree with one single top level branch
   int split = 99;  //try also split=1 and split=0
   int bufsize = 16000;
   Event *event = new Event;
   TTree T("T","Event demo tree");
   T.Branch("event","Event",&event,bufsize,split);

   //Build Event in a loop and fill the Tree
   for (int i=0;i<nevents;i++) {
      event->Build(i);
      T.Fill();
   }

   T.Print();  //Print Tree statistics
   T.Write();  //Write Tree header to the file
}
```

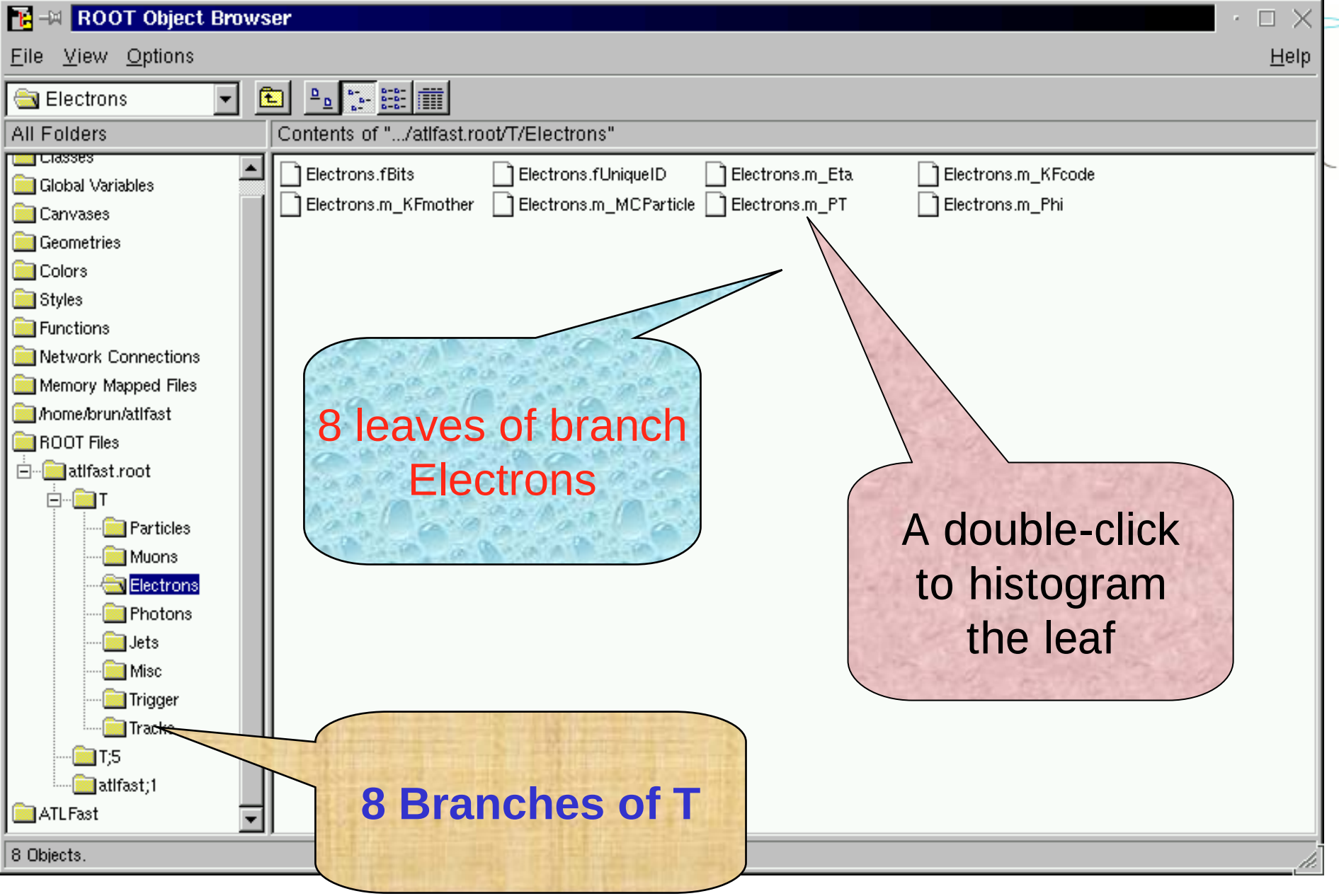
All the examples can be executed with CINT or the compiler

```
root > .x demoe.C
root > .x demoe.C++
```

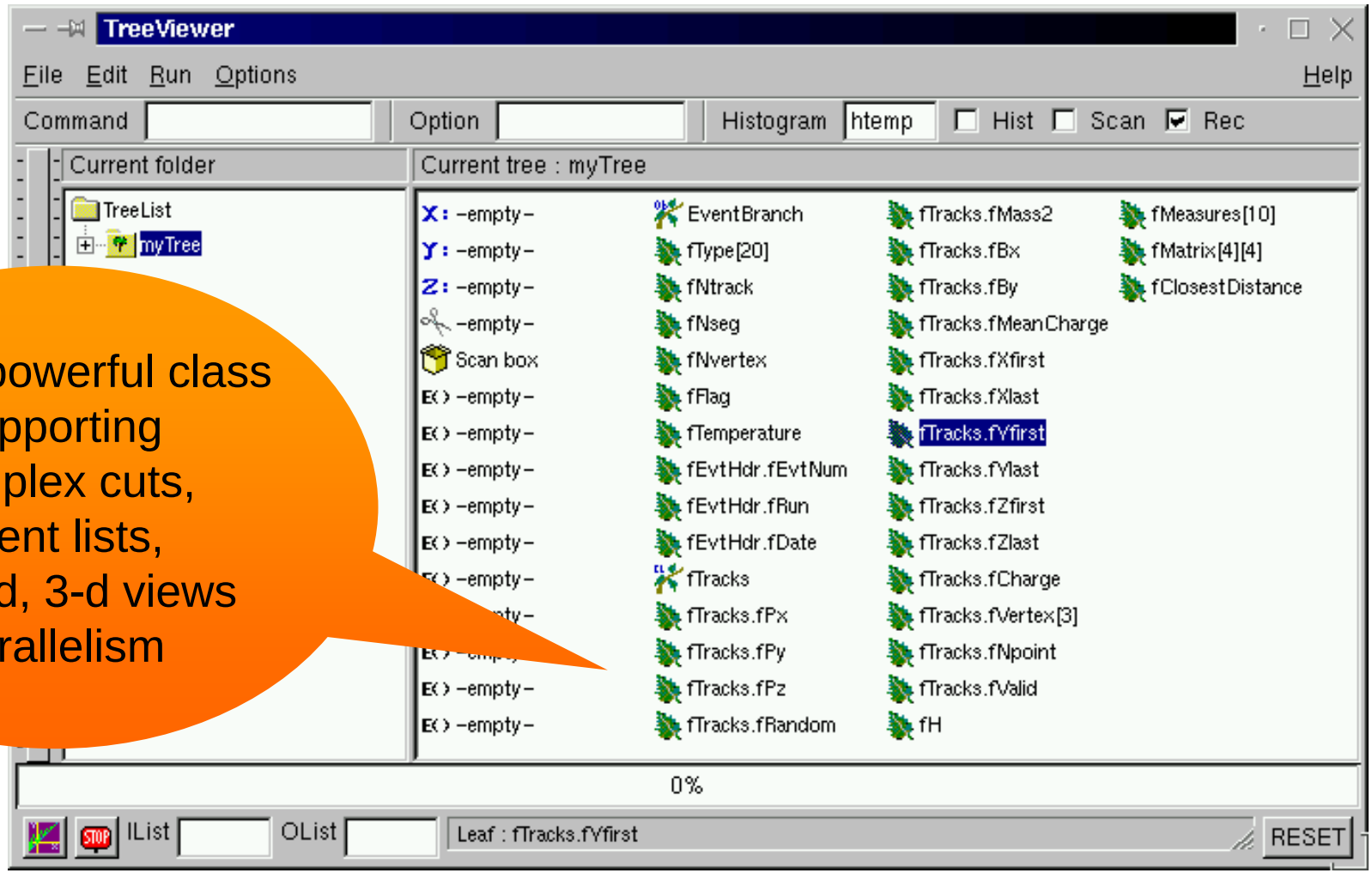
Tree example Event (read 1)

```
void demoer() {
   //load shared lib with the Event class
   gSystem->Load("$ROOTSYS/test/libEvent");

   //connect ROOT file
   TFile *f = new TFile("demoe.root");

   //Read Tree header and set top branch address
   Event *event = 0;
   TTree *T = (TTree*)f->Get("T");
   T->SetBranchAddress("event",&event);

   //Loop on events and fill an histogram
   TH1F *h = new TH1F("hntrack","Number of tracks",100,580,620);
   int nevents = (int)T->GetEntries();
   for (int i=0;i<nevents;i++) {
      T->GetEntry(i);
      h->Fill(event->GetNtrack());
   }

   h->Draw();
}
```

Rebuild the full event in memory

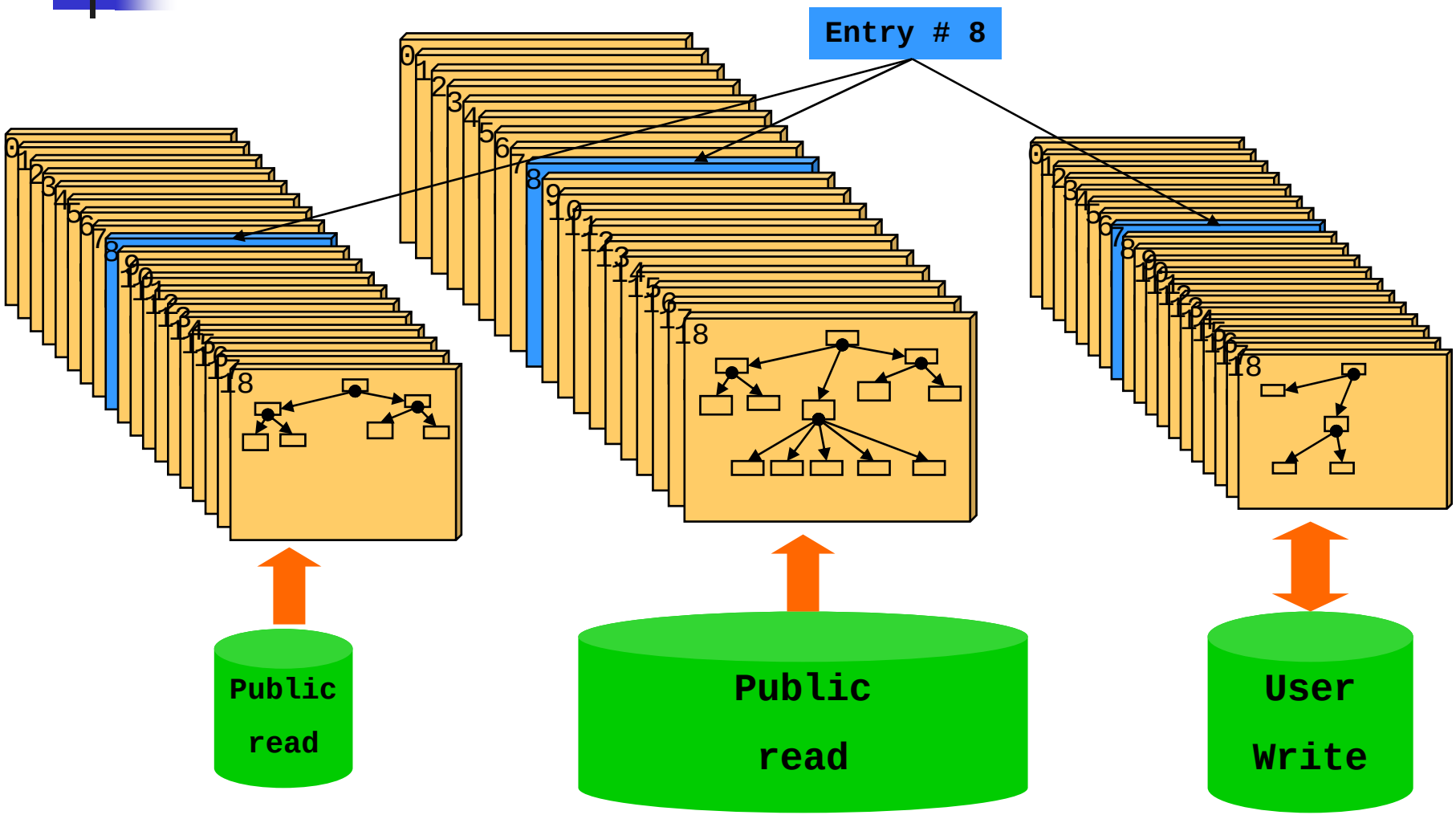

ROOT Object Browser
File View Options Help
Electrons
All Folders
Contents of ".../atlfast.root/T/Electrons"
Global Variables
Canvases
Geometries
Colors
Styles
Functions
Network Connections
Memory Mapped Files
/home/brun/atlfast
ROOT Files
atlfast.root
T
Particles
Muons
Electrons
Photons
Jets
Misc
Trigger
Tracks
T;5
atlfast;1
ATLFast
Electrons.fBits
Electrons.fUniqueID
Electrons.m_Eta
Electrons.m_KFcode
Electrons.m_KFmother
Electrons.m_MCParticle
Electrons.m_PT
Electrons.m_Phi
8 leaves of branch Electrons
A double-click to histogram the leaf
8 Branches of T
8 Objects.

The Tree Viewer & Analyzer

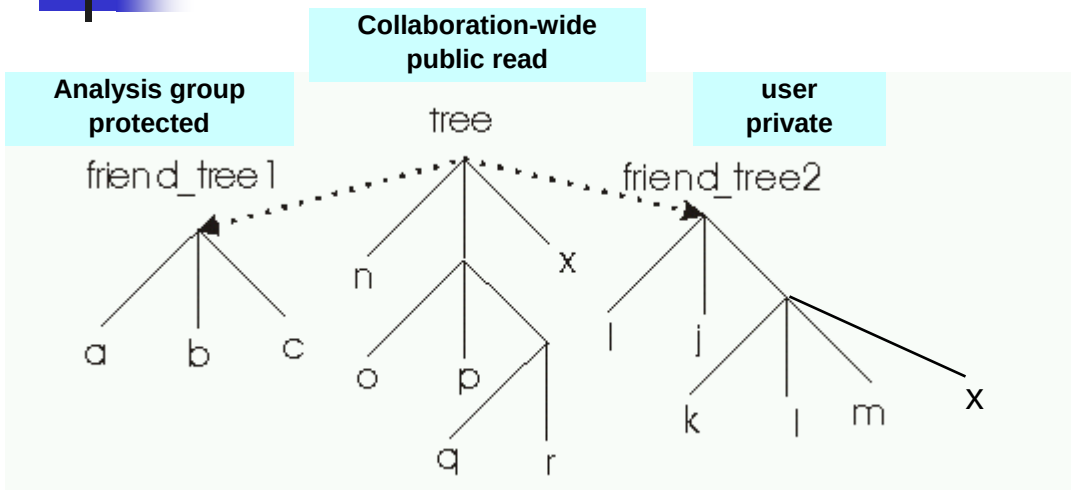


A very powerful class supporting complex cuts, event lists, 1-d,2-d, 3-d views parallelism

Tree Friends



Tree Friends

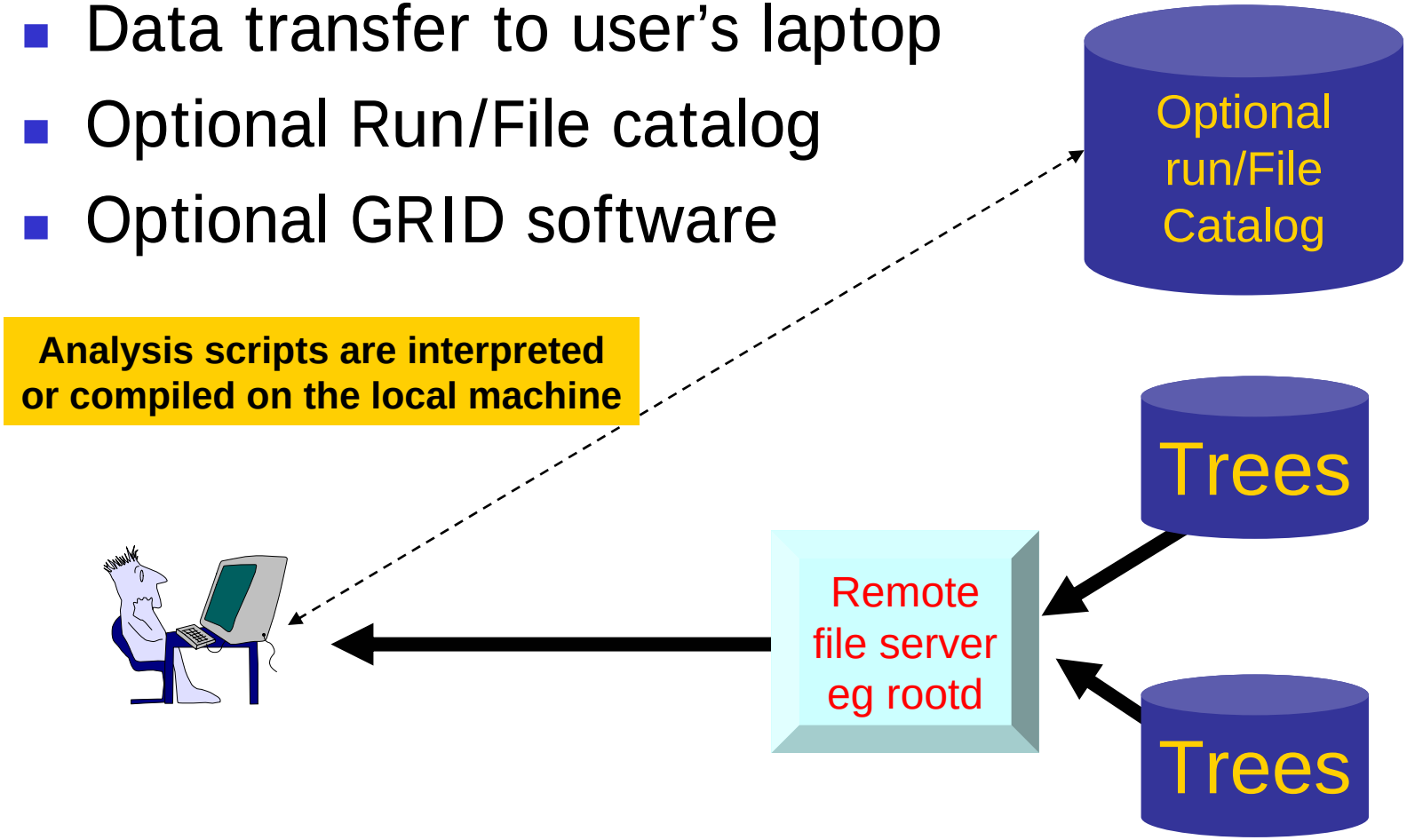


Processing time independent of the number of friends unlike table joins in RDBMS

```
Root > TFile f1("tree1.root");
Root > tree.AddFriend("tree2","tree2.root")
Root > tree.AddFriend("tree3","tree3.root");
Root > tree.Draw("x:a","k<c");
Root > tree.Draw("x:tree2.x","sqrt(p)<b");
```

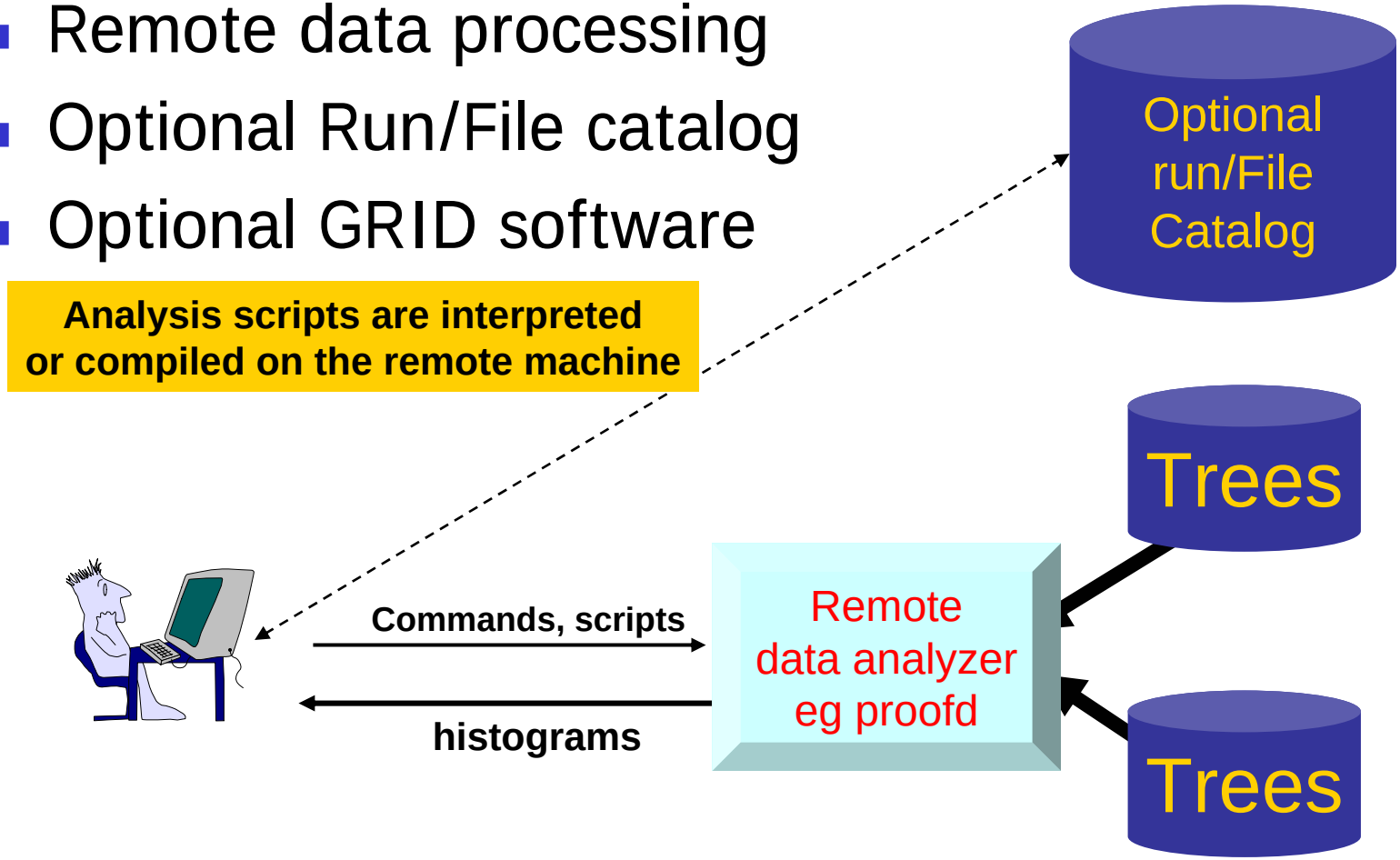
GRID: Interactive Analysis Case 1

- Data transfer to user's laptop
- Optional Run/File catalog
- Optional GRID software



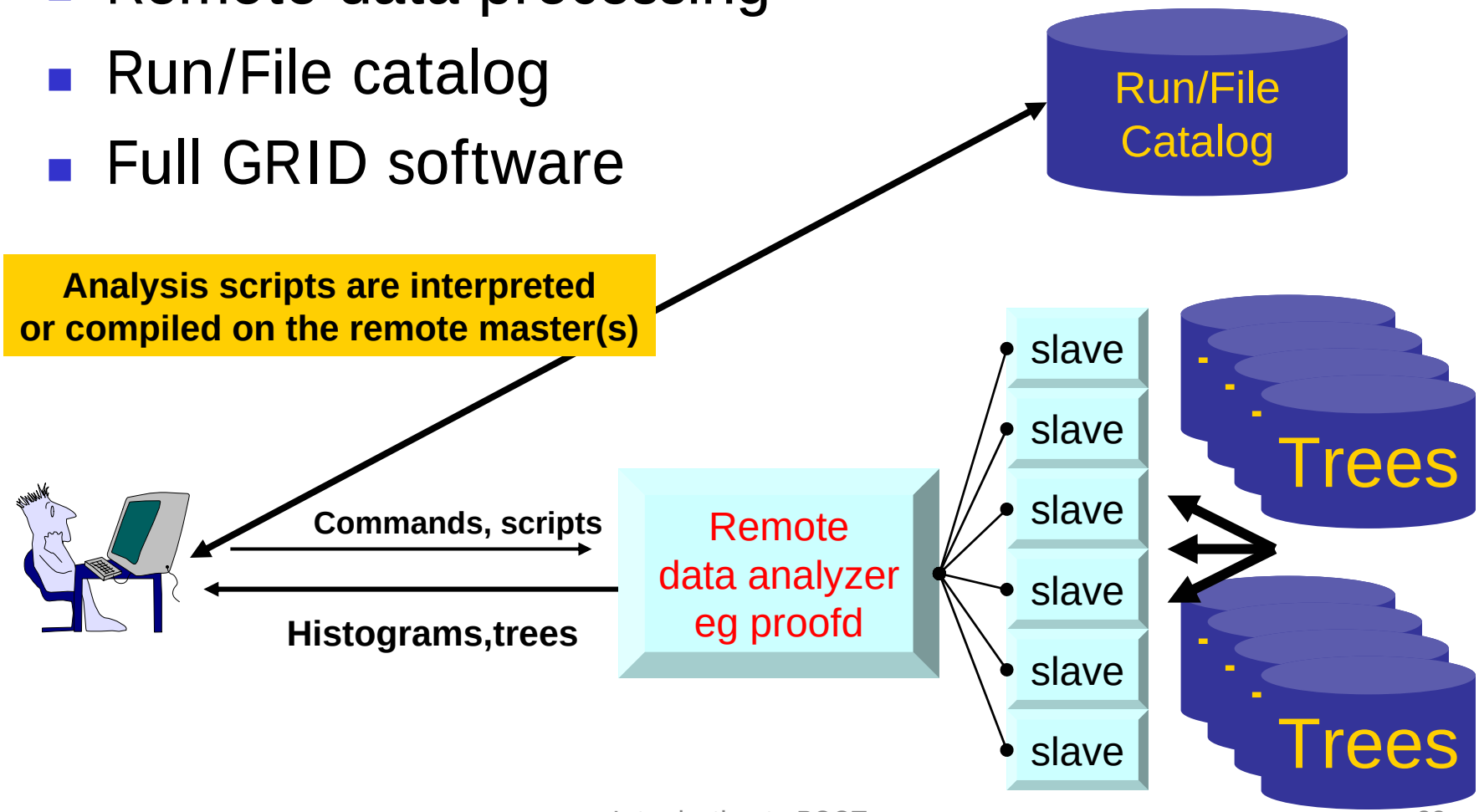
GRID: Interactive Analysis Case 2

- Remote data processing
- Optional Run/File catalog
- Optional GRID software



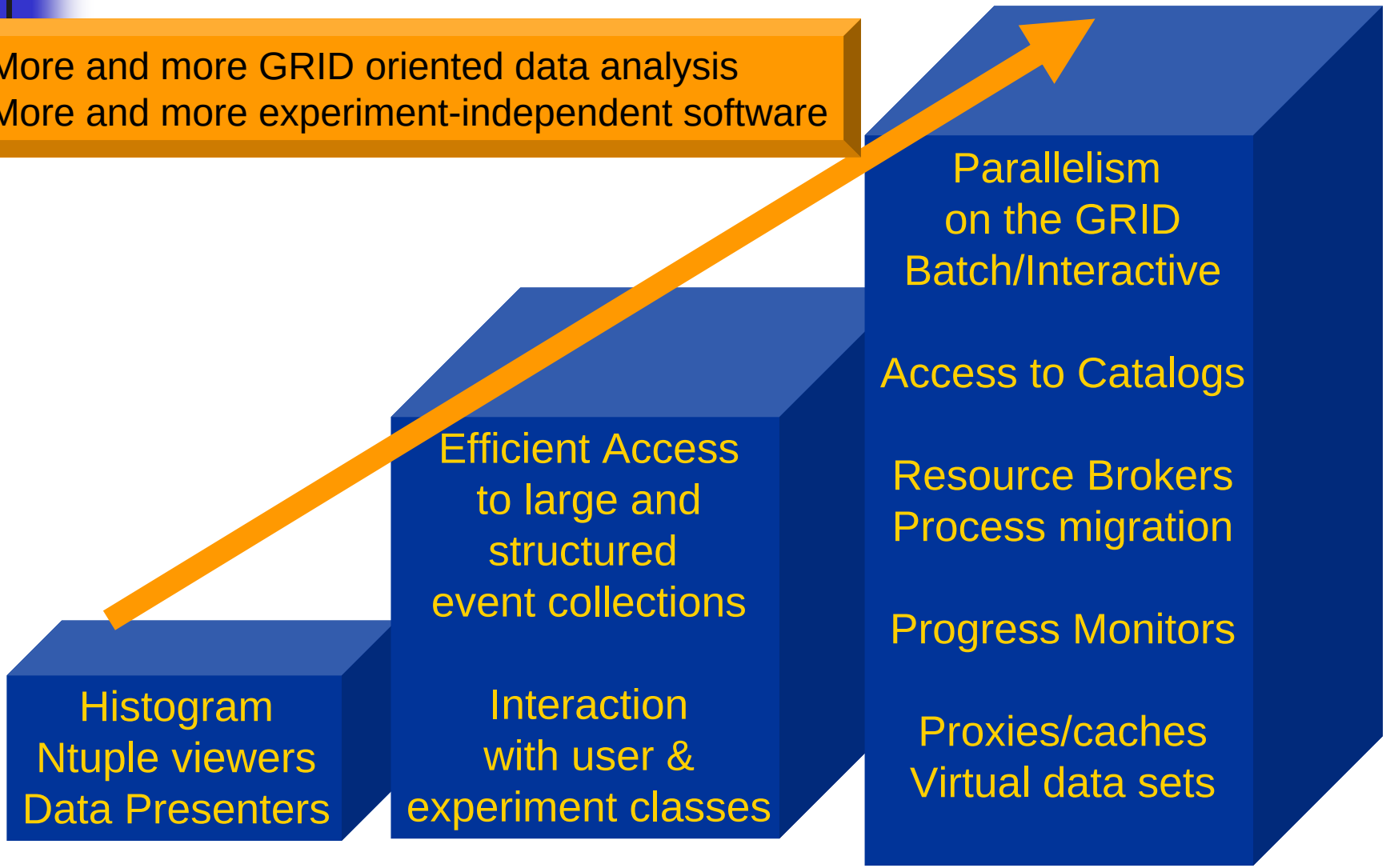
GRID: Interactive Analysis Case 3

- Remote data processing
- Run/File catalog
- Full GRID software



Trends Summary

More and more GRID oriented data analysis
More and more experiment-independent software



Google: a good model
Make it simple

Web Images Groupes Annuaire Actualités

Google ARDA interactive analysis Rechercher Recherche avancée Préférences

Rechercher dans : Web Pages francophones Pages : France

Web Résultats **1** - **10** sur un total d'environ **1,610** pour **ARDA interactive analysis**. (**0.17** secondes)

[PDF] The ALICE **Analysis** Approach **ARDA**
Format de fichier: PDF/Adobe Acrobat - Version HTML
... Andreas J. Peters CERN/Geneva @ **ARDA** Workshop 21./22.1.04 **Interactive Analysis** model with PROOF PROOF allows **interactive analysis** on local clusters with a **...**
agenda.cern.ch/askArchive.php?base=agenda& categ=a036745&id=a036745s1t1/transparencies - Pages similaires

[PDF] Microsoft PowerPoint - ArdaAppsJan04_gro1074762582.ppt
Format de fichier: PDF/Adobe Acrobat - Version HTML
... ARDA Workshop Jan 2004 Slide 7 Torre Wenaus, BNL/CERN **Interactive Analysis** Tools Interfacing to tools supporting **interactive** (low-latency, rapid- response **...**
agenda.cern.ch/askArchive.php?base=agenda& categ=a036745&id=a036745s6t1%2Ftransparencies%2FArdaApp... - Pages similaires
[Autres résultats, domaine agenda.cern.ch]

RTAG11 **ARDA** Documents - [Traduire cette page]
... presentation to GriPhyN meeting [ppt, pdf]; October 3, **ARDA** SC2 report **...** Components and Services" [link] CS11 use cases for **interactive analysis** -> Grid services **...**
www.uscms.org/s&c/lcg/ARDA/docs.html - 7k - En cache - Pages similaires

[PPT] www.uscms.org/s&c/lcg/ARDA/presentations/2003-09-18-ARDA-rwg.ppt
Format de fichier: Microsoft Powerpoint 97 - Version HTML
... ARDA services present an API, called by applications like the experiments frameworks, **interactive analysis** packages, Grid portals, Grid shells, etc. **...**
Pages similaires
[Autres résultats, domaine www.uscms.org]

[PPT] ADA: ATLAS Distributed **Analysis**
Format de fichier: Microsoft Powerpoint 97 - Version HTML
... Incorporate ideas from PPDG, **ARDA**, ... If available in time. **...** 14. David Adams. ATLAS. Deliverables for first release (cont). **Interactive analysis** service. **...**
www.usatlas.bnl.gov/ADA/talks/031215_ada.ppt - Pages similaires

Simple interface
Available everywhere
Hidden Parallelism
Distributed DB
Don't know a priori the data location
Fast

Playing with ROOT on lxplus

- Set the environment variables
  - setenv ROOTSYS /afs/cern.ch/sw/root/v4.00.08/rh73_gcc32/root
  - setenv LD_LIBRARY_PATH $ROOTSYS/lib:$LD_LIBRARY_PATH
  - setenv PATH $ROOTSYS/bin:$PATH
- Copy the ROOT tutorials
  - cd $HOME
  - cp $ROOTSYS/tutorials .
  - cd tutorials
- Run ROOT
  - root

Playing with ROOT on a PC

- Import a binary tar file from:
  - http://root.cern.ch/root/Version400.html
  - Eg: Intel x86 Linux for Redhat 7.3 and gcc 3.2, version 4.00/08
- Untar the file in your home directory
  - cd $HOME
  - tar zxvf root_v4.00.08.Linux.RH7.3.gcc32.tar.gz
- Set the environment variables
  - setenv ROOTSYS $HOME/root
  - setenv LD_LIBRARY_PATH $ROOTSYS/lib:$LD_LIBRARY_PATH
  - setenv PATH $ROOTSYS/bin:$PATH
- Go to the ROOT tutorials
  - cd $HOME
  - cp $ROOTSYS/tutorials .
  - cd tutorials
- Run ROOT
  - root

My first session

root

```
root [0] 344+76.8
(const double)4.20800000000000010e+002
root [1] float x=89.7;
root [2] float y=567.8;
root [3] x+sqrt(y)
(double)1.13528550991510710e+002
root [4] float z = x+2*sqrt(y/6);
root [5] z
(float)1.09155929565429690e+002
root [6] .q
```

root

See file $HOME/.root_hist

root [0] try up and down arrows

My second session

root

```
root [0] .x session2.C
for N=100000, sum= 45908.6
root [1] sum
(double)4.59085828512453370e+004
Root [2] r.Rndm()
(Double_t)8.29029321670533560e-001
root [3] .q
```

unnamed macro
executes in global scope

session2.C

```
{
   int N = 100000;
   TRandom r;
   double sum = 0;
   for (int i=0;i<N;i++) {
      sum += sin(r.Rndm());
   }
   printf("for N=%d, sum= %g\n",N,sum);
}
```

My third session

root

```
root [0] .x session3.C
for N=100000, sum= 45908.6
root [1] sum
Error: Symbol sum is not defined in current scope
*** Interpreter error recovered ***
Root [2] .x session3.C(1000)
for N=1000, sum= 460.311
root [3] .q
```

session3.C

```
void session3 (int N=100000) {
   TRandom r;
   double sum = 0;
   for (int i=0;i<N;i++) {
      sum += sin(r.Rndm());
   }
   printf("for N=%d, sum= %g\n",N,sum);
}
```

Named macro
Normal C++ scope rules

My third session with ACLIC

```
root [0] gROOT->Time();
root [1] .x session4.C(10000000)
for N=10000000, sum= 4.59765e+006
Real time 0:00:06, CP time 6.890
root [2] .x session4.C+(10000000)

for N=10000000, sum= 4.59765e+006
Real time 0:00:09, CP time 1.062
root [3] session4(10000000)
for N=10000000, sum= 4.59765e+006
Real time 0:00:01, CP time 1.052
root [4] .q
```

File session4.C
Automatically compiled
and linked by the
native compiler.
Must be C++ compliant

session4.C

```
#include "TRandom.h"
void session4 (int N) {
   TRandom r;
   double sum = 0;
   for (int i=0;i<N;i++) {
      sum += sin(r.Rndm());
   }
   printf("for N=%d, sum= %g\n",N,sum);
}
```

Macros with more than one function

```
root [0] .x session5.C >session5.log
root [1] .q
```

```
root [0] .L session5.C
root [1] session5(100); >session5.log
root [2] session5b(3)
sum(0) = 0
sum(1) = 1
sum(2) = 3
root [3] .q
```

.x session5.C
executes the function session5 in session5.C

use gROOT->ProcessLine to execute a macro from a macro or from compiled code

session5.C

```
void session5(int N=100) {
   session5a(N);
   session5b(N);
   gROOT->ProcessLine(".x session4.C+(1000)");
}
void session5a(int N) {
   for (int i=0;i<N;i++) {
      printf("sqrt(%d) = %g\n",i,sqrt(i));
   }
}
void session5b(int N) {
   double sum = 0;
   for (int i=0;i<N;i++) {
      sum += i;
      printf("sum(%d) = %g\n",i,sum);
   }
}
```

Interactive Demo

You can import the tar file with all demos from
ftp://root.cern.ch/root/SummerStudents2004.tar.gz

```
root summershow.C
```

You can find more demos, examples, tests at

$ROOTSYS/tutorials

$ROOTSYS/test

$ROOTSYS/test/RootShower