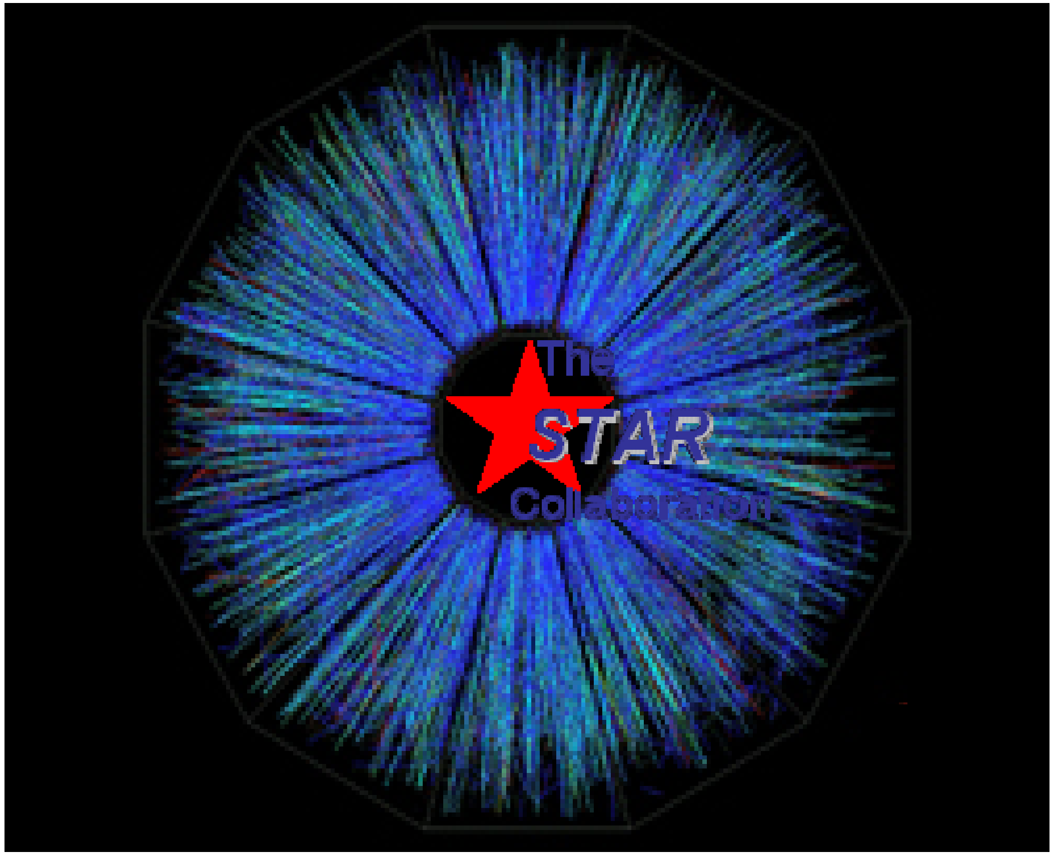
ROOT4STAR: a ROOT based framework for user analysis and data mining

Jérôme Lauret - **jlauret@bnl.gov**



Introduction

- This should really have been a talk with a title such as "*How ROOT helps an experiment with its data taking and analysis – Real life experience from PBytes experiments*"

- ROOT is HEAVILY used in RHIC/STAR
  - In fact, all RHIC experiments use it
  - It is used to build frameworks for standard or distributed computing
  - It is versatile enough to allow developments (QtRoot, …)
  - It is stable enough that we use it from online to offline …

But before more ROOT commercials …

RHIC facility …

- The **S**olenoidal **T**racker **A**t **R**HIC **http://www.star.bnl.gov/ is an experiment located at the Brookhaven National Laboratory (BNL), USA**

New Haven
Bridgeport
Stamford
Image © 2005 EarthSat
© 2005 Sanborn
©2005 Google
Pointer 40°54'46.76" N 72°48'50.07" W
Streaming |||||||||| 100%
Eye alt 72.41 mi

STAR experiment ...

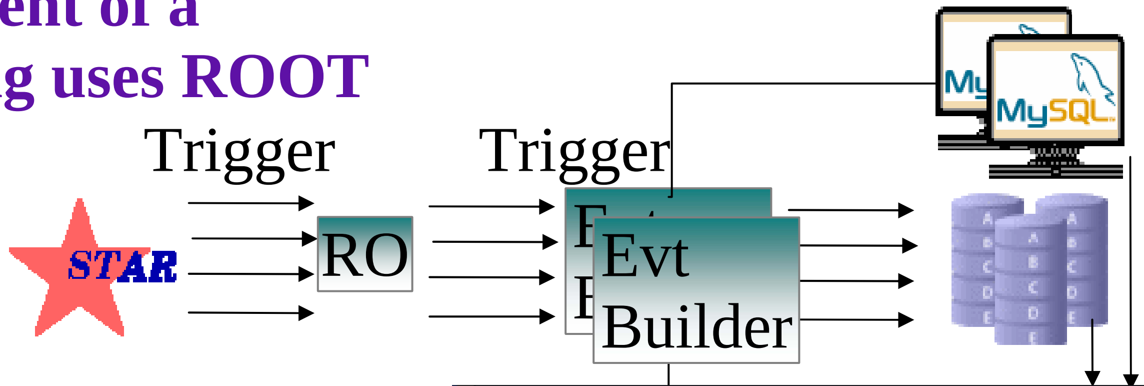
- The Solenoidal Tracker At RHIC
  - A collaboration of **616 people** wide, spanning over **12 countries** for a total of **52 institutions**
  - A Pbytes scale experiment overall (raw+reconstructed) with several Million of files

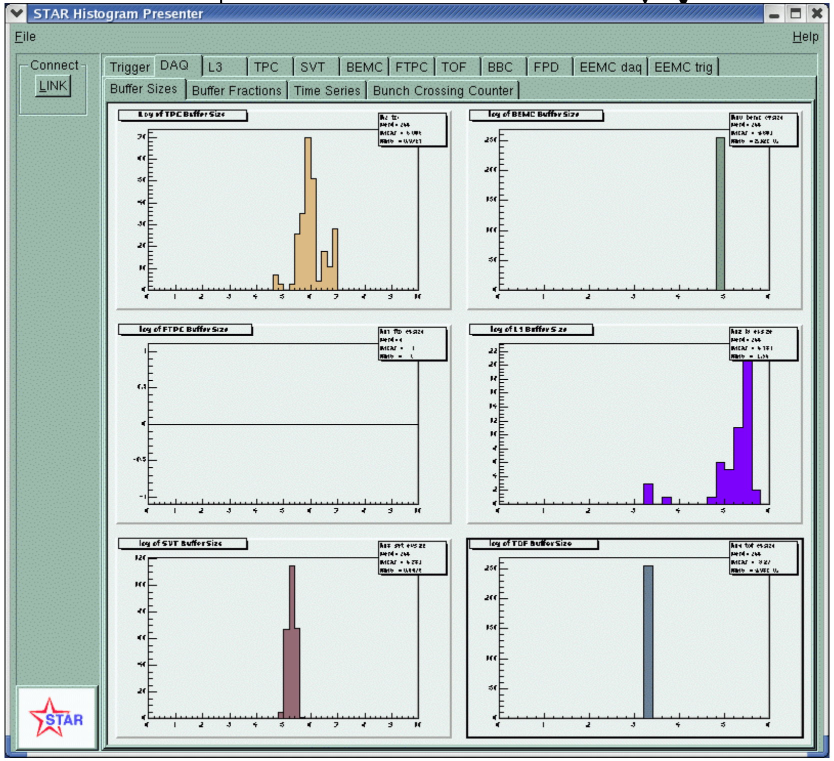
- The Physics
  - A multi-purpose detector system for Heavy Ion and p+p program

First use of ROOT – Online

Right from the very moment of a collision, STAR computing uses ROOT to

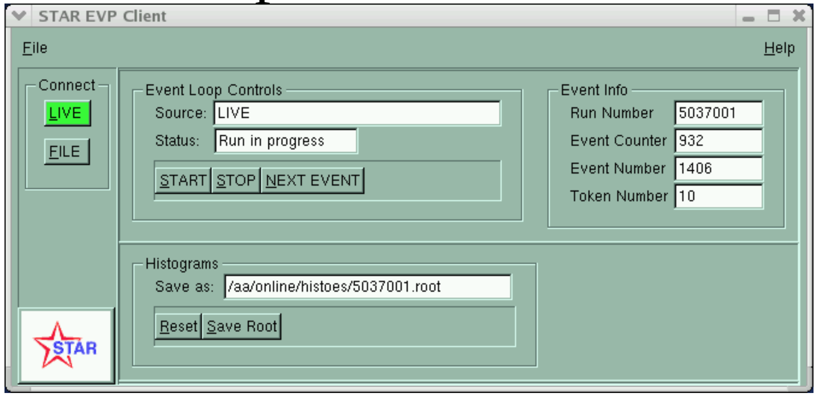
- Provide Online QA
- Monitoring
- Event display





Standard approach here again …

- GUI are easy to build …
- Interfacing event-Pools and ROOT histogram is not complicated …



Event Loop Controls

Source: LIVE

Status: Run in progress

Event Info

Run Number 5037001

Event Counter 932

Event Number 1406

Token Number 10

Histograms

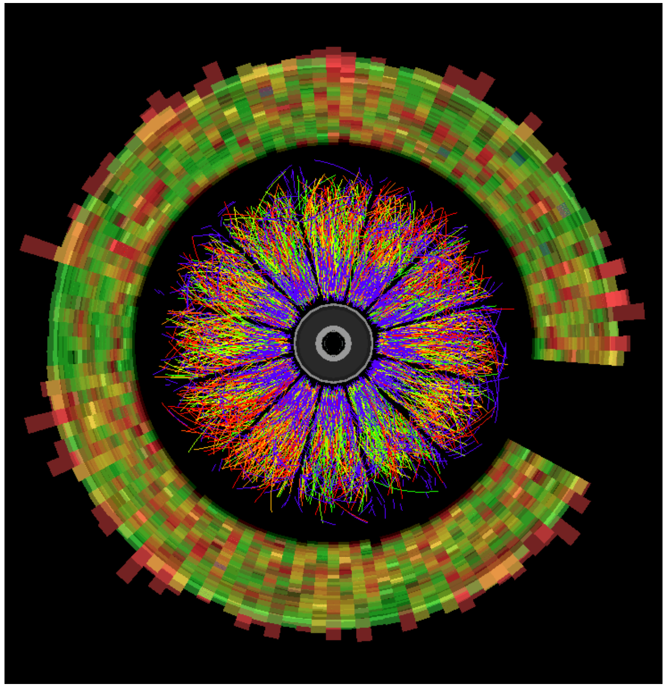
Save as: /aa/online/histoes/5037001.root

Event display

Such images are produced minutes after collision. During the run, collision “movies” are available from the Web for public display …

http://www.star.bnl.gov/STAR/comp/vis/StarEvent.html

http://www.star.bnl.gov/STAR/comp/vis/StarEvent_S.html



More exotic

We had the events displayed on the façade in Munich “live”

Art & Physics

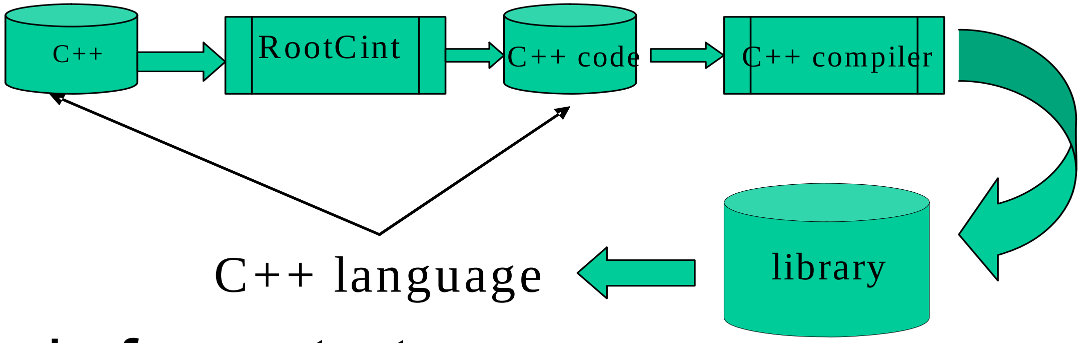
An unusual art project taking place in Munich, Germany uses live data from RHIC's STAR detector to create a public light show. March 14 - 20.

Live webcam from 11 a.m. to 2 p.m.

The Offline STAR framework

Framework in STAR - root4star

- STAR had a few phases of framework
  - **Early adoption of ROOT ~ 2000+** – V. Fine / V. Perevotchikov
  - ROOT4STAR used for ALL STAR published analysis to date
- ROOT Provides a lot …
  - Histogram and fitting, Graphics (2D, 3D), IO, NTuples/Trees
  - Collection of classes, schema evolution, stability
  - UI
    - Browsers, Panels, Tree Viewers
    - Buildt-in OO model, C++



As many features as one needs for a start …

So the early adoption …

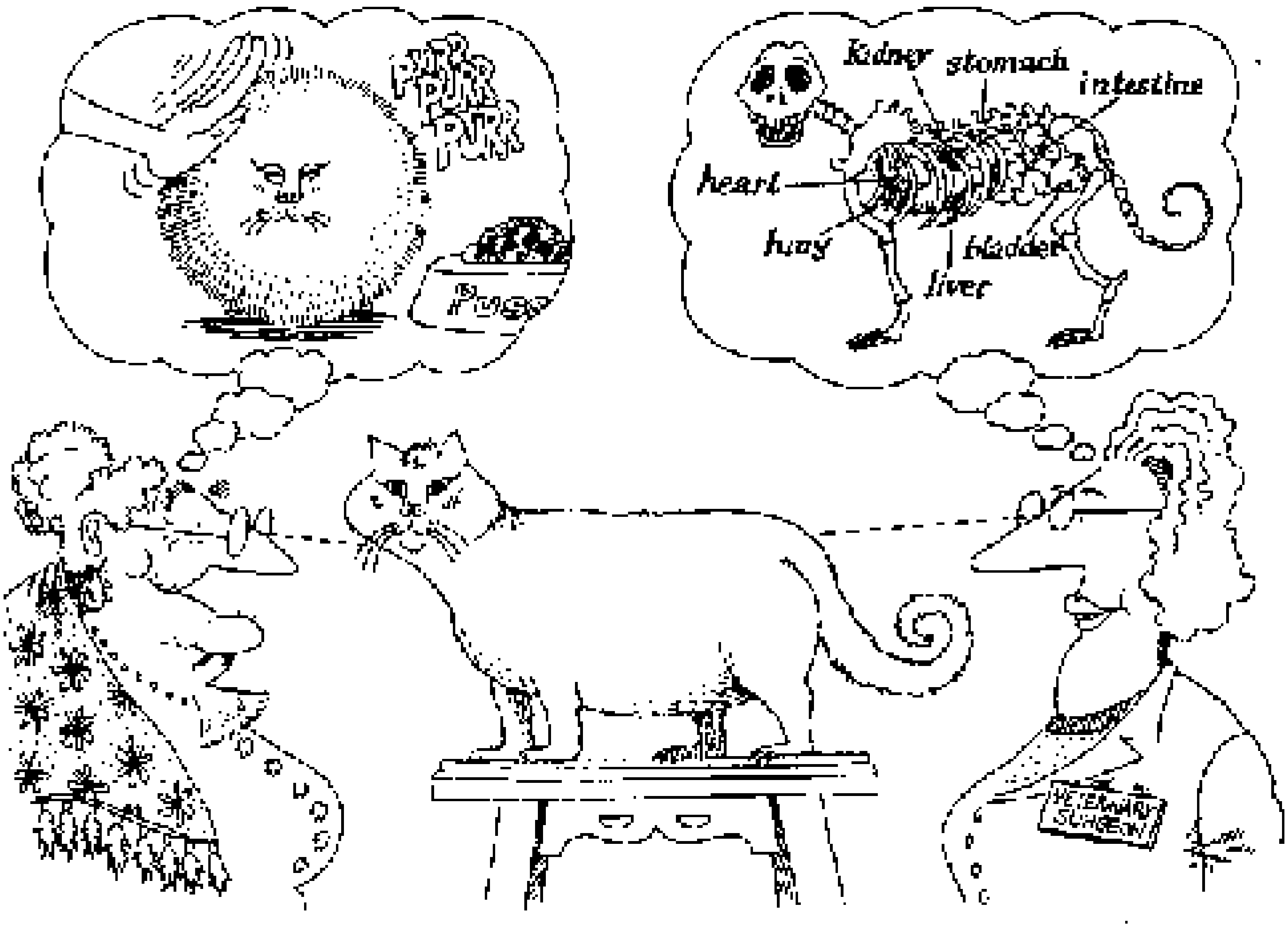
root4star or ROOT?

- ROOT4STAR = ROOT+STAR specific
  - All ROOT features
  - Interfaces with G3 (strong linking)
  - STAR additional base Classes (TTable, ...)
  - Qt based GUIs and Event display
  - ...

- A single framework for
  - Simulation
  - Data mining
  - User analysis

Physics ready micro-DST DO NOT need the framework (root+few libs suffices)
Even works on a laptop running Windows ;-)

The two ways to see a framework …

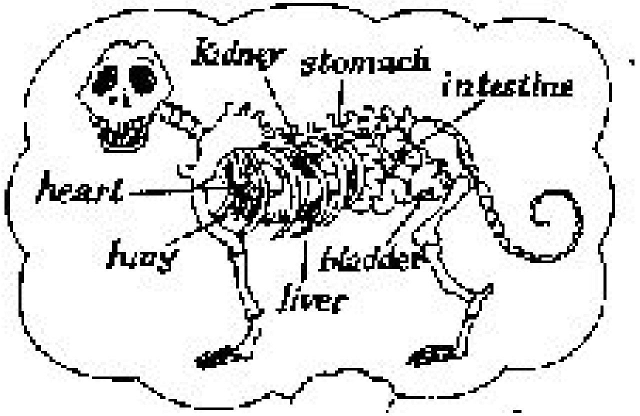


In real life (hopefully the user end)

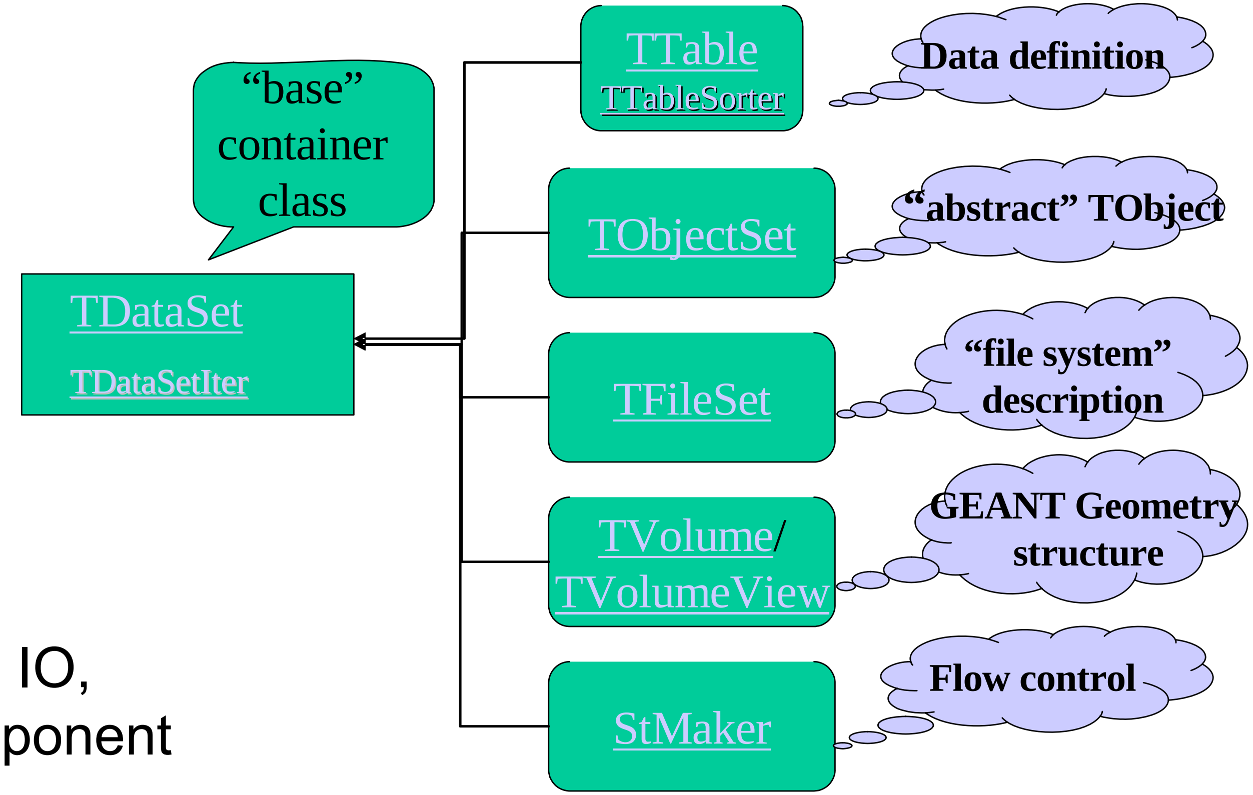
- The STAR code is
  - Codes deriving from a base class StMaker
    - Arranged into libraries, blocks are components
    - Makers are loaded via dynamic libraries
    - User can start from **template example**
  - A hierarchical collection of STAR Makers handling data-sets
  - A single instance of a “chain”, a “steering” component
    - All dependencies in one another sorted
    - **NO NEED TO KNOW** from users

- A few special makers
  - IOMaker, handles all IO - Persistent event model StEvent
  - Messenger manages all messages (do not prevent cout / printf)
  - DBMaker manages transparently all DB related access (event timestamp based)

root4star



- A **TDataSet** class
  - from which data sets and makers inherit
  - allows the construction of **hierarchical organizations of components and data**
  - **centralizes almost all common tasks**
    - Data set navigation, IO, database, inter-component communication



"base" container class

TDataSet

TDataSetIter

TTable

TTableSorter

Data definition

TObjectSet

"abstract" TObject

TFileSet

"file system" description

TVolume/ TVolumeView

GEANT Geometry structure

StMaker

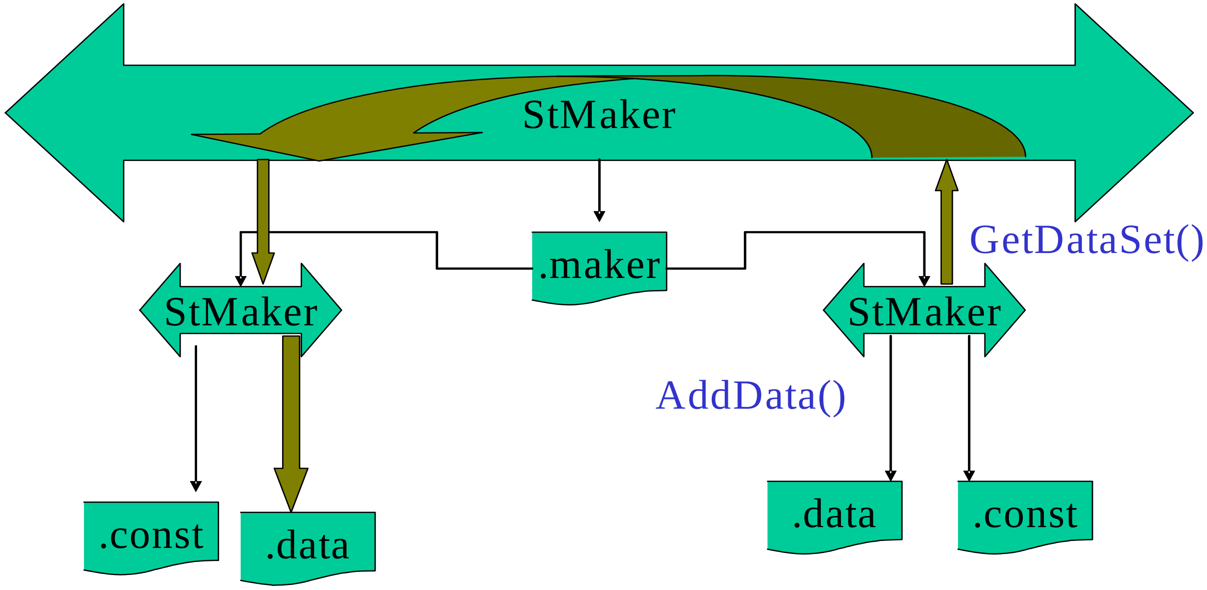
Flow control

root4star

Presented in past ROOT Workshops and CHEP conference …

- **STAR framework designed to support**
  - **chained components**
  - can themselves be composite sub-chains
  - components (“**makers**”) **managing “datasets**” they have created and are responsible for
  - “makers” can communicate and Get/Add global datasets

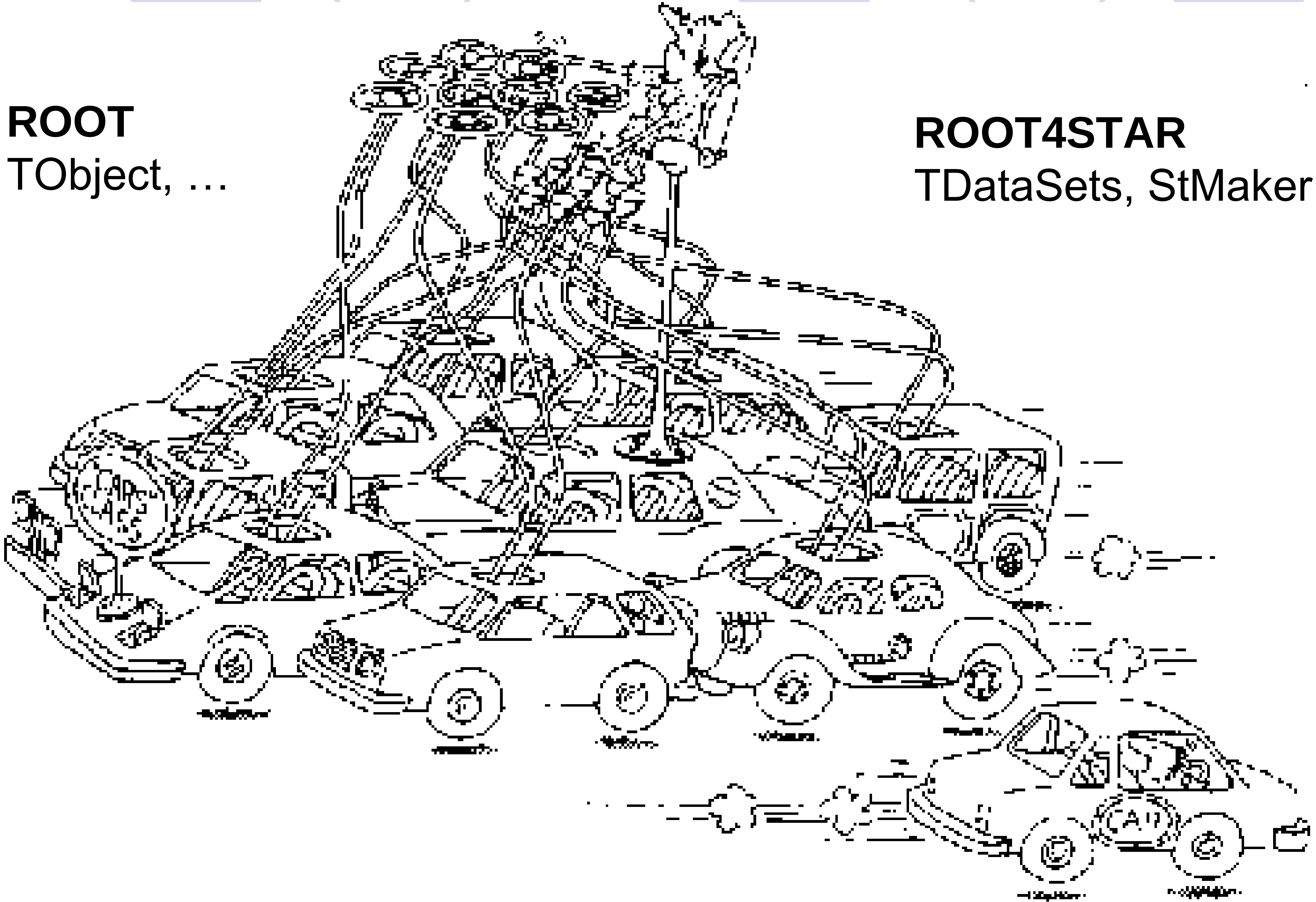
Usual OO approach
Base class have common methods
Init() InitRun()
Finish() FinishRun()
Make()
…



OO analogy …

ROOT
TObject, …

ROOT4STAR
TDataSets, StMaker…



Under the hood …

```
class StMaker : public TDataSet{
public:
                         StMaker(const char *name="",const char *dummy=0);
  virtual               ~StMaker();
  virtual Int_t     IsChain() const {return 0;}

  /// User defined functions
  virtual void        Clear(Option_t *option="");
  virtual Int_t        InitRun(int runumber);
  virtual Int_t        Init();
  virtual void        StartMaker();
  virtual Int_t        Make();
  virtual Int_t        Finish();
  virtual Int_t        FinishRun(int oldrunumber);

  // Get methods
  virtual TDataSet  *GetData(const char *name, const char* dir=".data") const;
  virtual TDataSet  *GetDataSet (const char* logInput) const
                                         {return FindDataSet(logInput);}
  virtual Int_t          GetEventNumber() const ;
  virtual Int_t          GetRunNumber() const ;
  virtual TDatime     GetDateTime() const;
  virtual Int_t          GetDate()  const ;
  virtual Int_t          GetTime()  const ;
```

For example, methods to handle timestamps (simulation or reco)

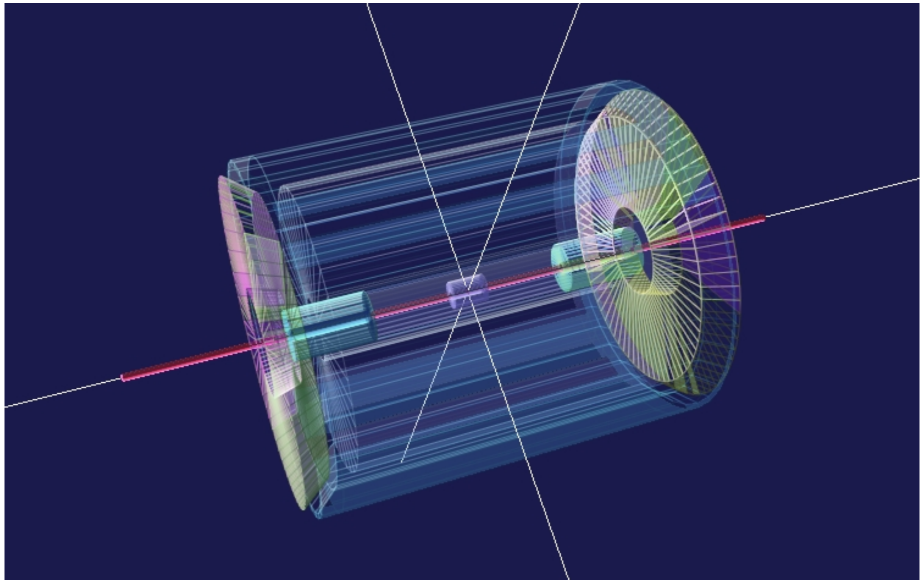
Why ROOT4STAR instead of ROOT?

Good question …

- In principle, we are (were) only tight to G3
- Evaluated VMC approach
  - Got rid of legacy codes using (yes ☹) c-blocks
  - Introduced TGeo based geometry
    - Currently made using g2root from G3
    - Geo comparison evaluated – seems to do
  - Shaping a common IO model for data / simulation
    - VMC means we retire our FZ format
    - New model was already there …
- We will have news by CHEP 06

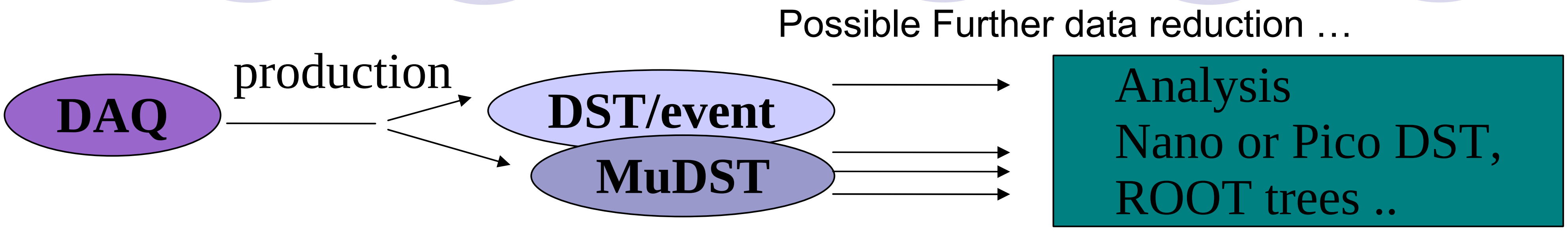
Geometry user front end …

- We like XML ;-)
  - Worked with AGDD and GraXML
  - Note that our current geometry do not make use of db parameters to “re-scale” (perfect geometry)
  - Basic [initial] idea: [?]GDD to TGeo
- So far
  - Have basic description in place
  - Missing “many” (still), polyhedral, negative dimension volume (expanding)
- Next
  - Putting a regression test in place
  - Porting and testing the full geometry



The Offline STAR data & model

Analysis …



Possible Further data reduction …

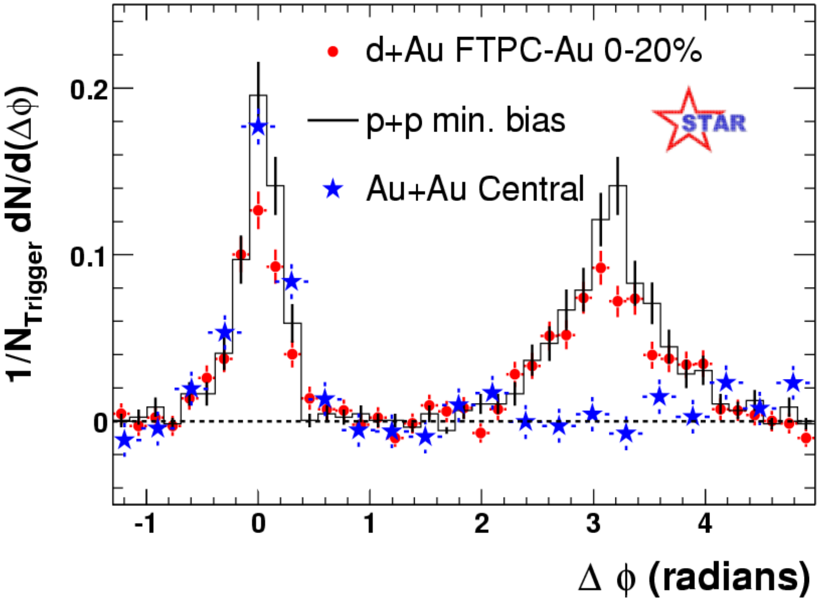
DAQ

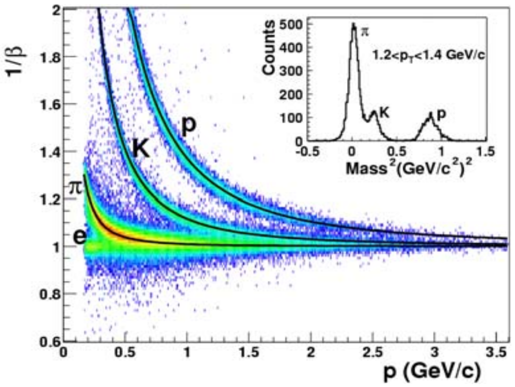
production

DST/event

MuDST

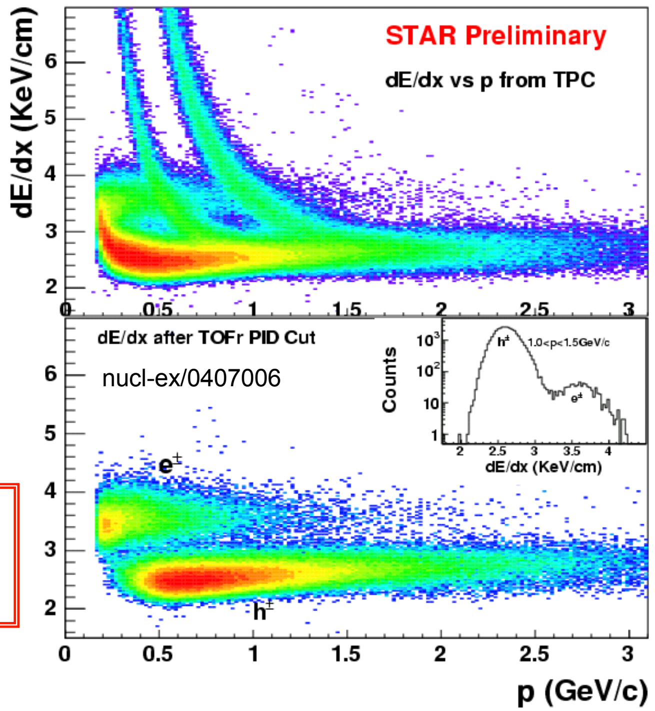
Analysis
Nano or Pico DST,
ROOT trees ..





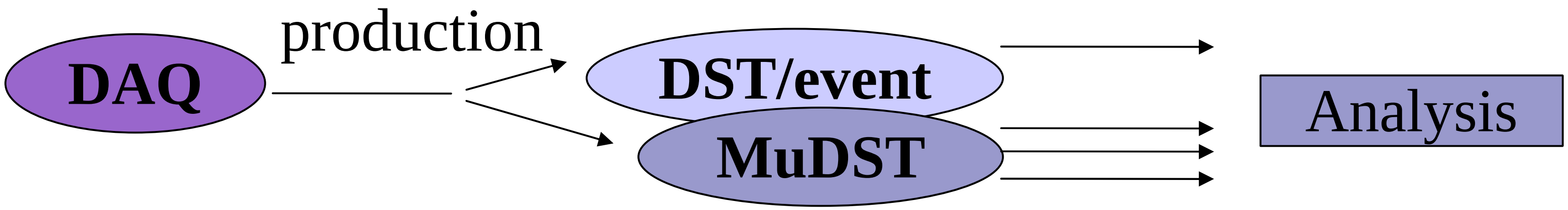
Hadron identification:
STAR Collaboration, *nucl-ex/0309012*

Electron identification:
TOFr |1/ß-1| < 0.03
TPC dE/dx electrons!!!



ALL is classic-root based …

Data Sets sizes



DAQ

production

DST/event

MuDST

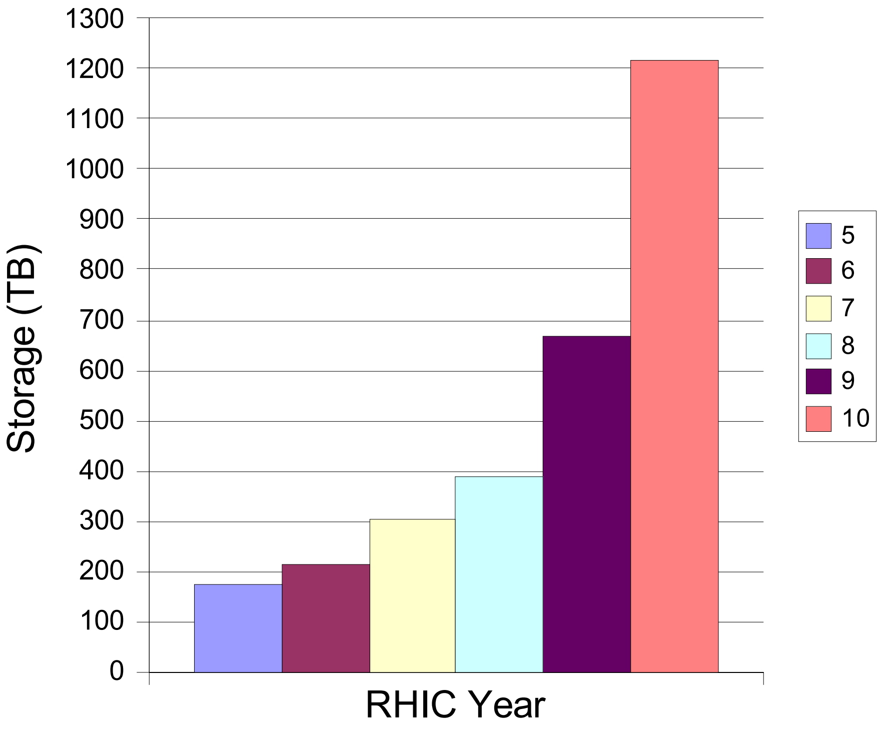
Analysis

- Raw Data Size
  - <> ~ 2-3 MB/event - All on Mass Storage (HPSS as MSS)
  - Needed only for calibration, production – Not centrally or otherwise stored
- Real Data size
  - Data Summary Tape+QA histos+Tags+run information and summary: <> ~ 2-3 MB/event
  - Micro-DST: 200-300 KB/event
- Total Year4 (example)

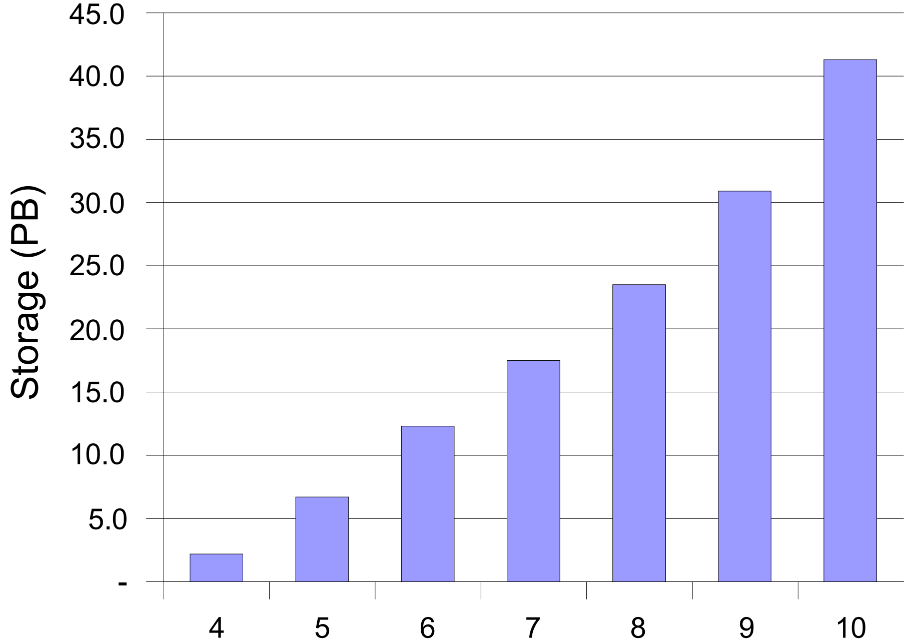
Total num events	138260234
GB total	357369,72
TB total	348,99
MuDst	34,9

Data Sets sizes - Tier0 Projections

Raw data projection

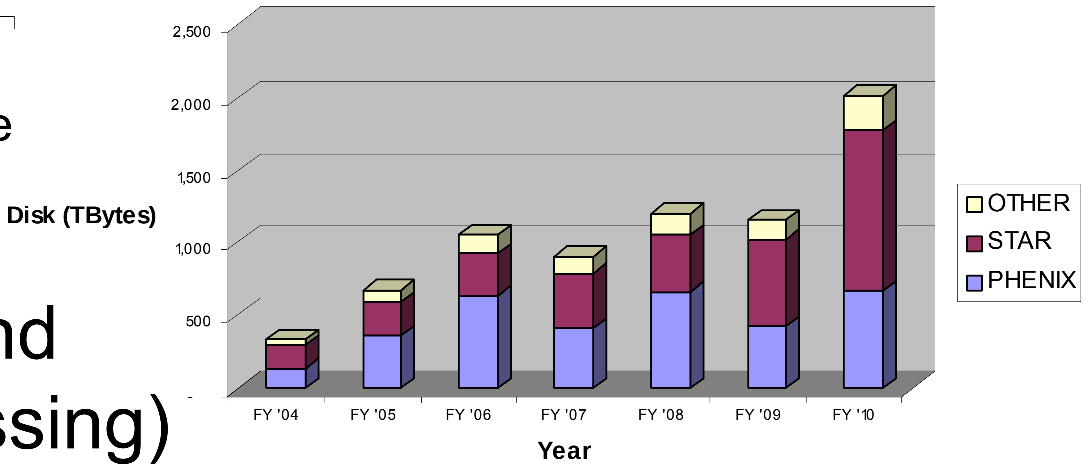


RHIC Total Tape Required



Projected Disk Requirement

RHIC Year



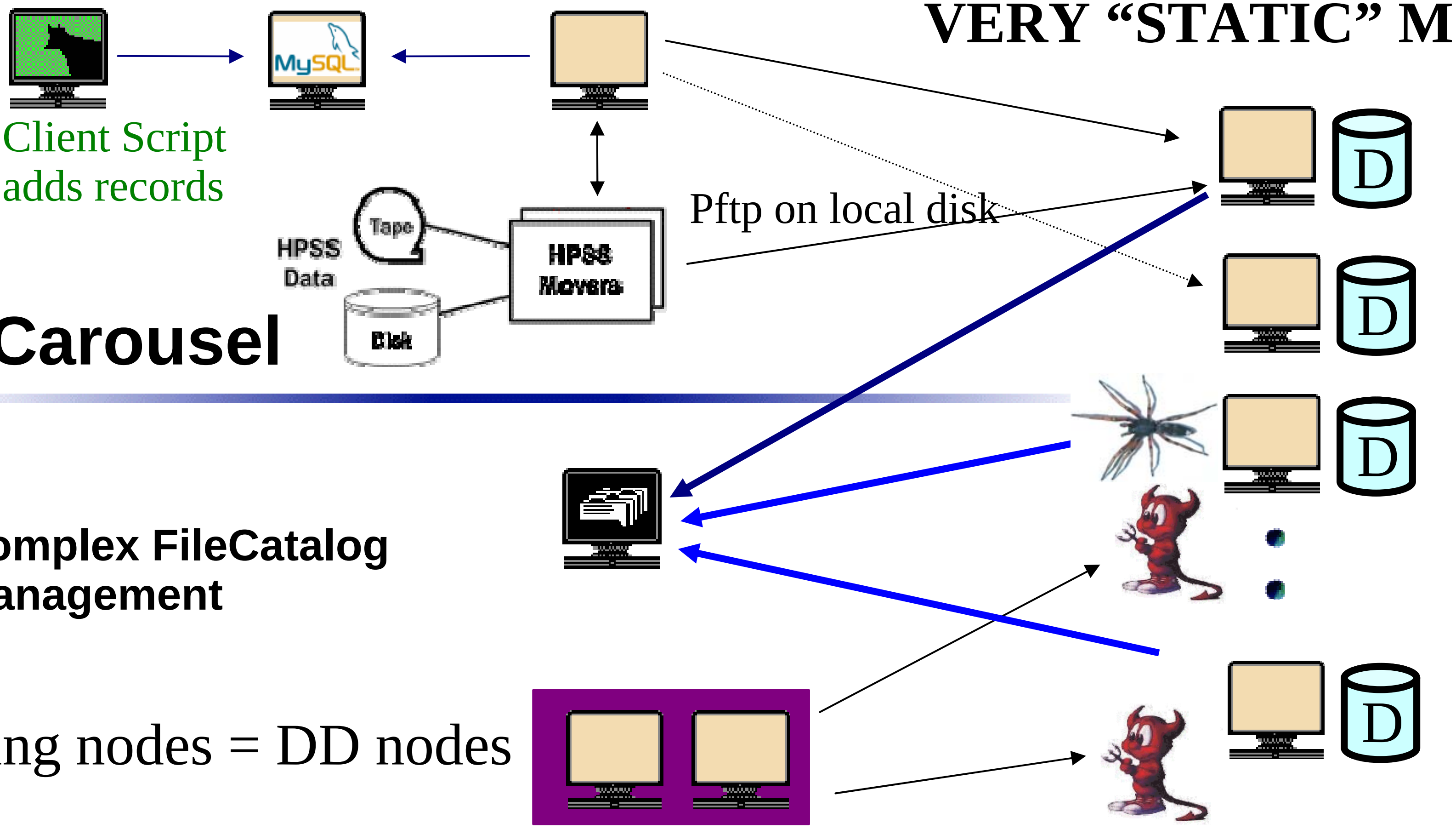
Note – Reconstructed ~ raw size
2.2 reconstructed passes …

Huge storage demand
(even greater processing)

Distributed Disks = SE attached to specific CE – **ROOTD** allows accessing those …

Where does this data go ??

STARTED WITH A VERY HOMEMADE VERY "STATIC" MODEL



Client Script adds records

Pftp on local disk

DataCarousel

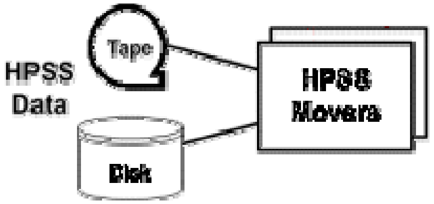
Complex FileCatalog Management

Processing nodes = DD nodes

ROOTD

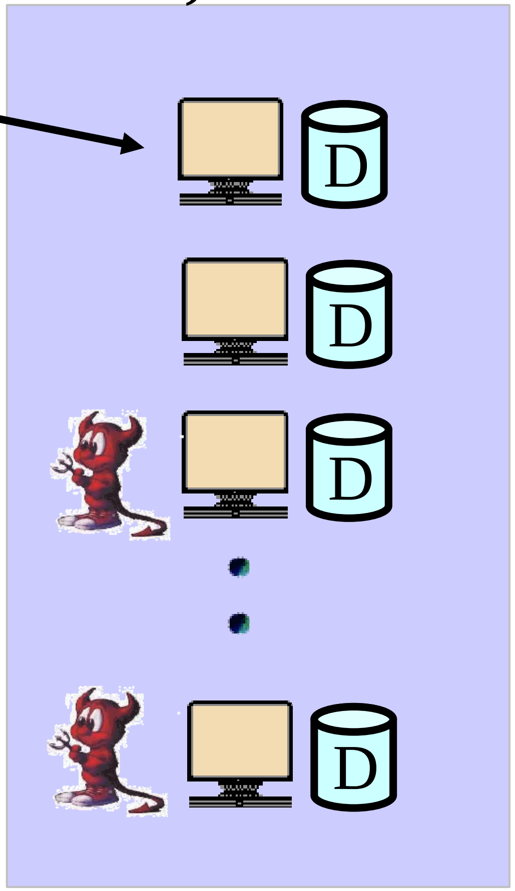
A purely economic model … DD cost 1/5 to 1/10th of CD

Distributed Disks – Xrootd era



Pftp on local disk from Xrootd

In the process of replacing "it" with XROOTD, XROOT+SRM



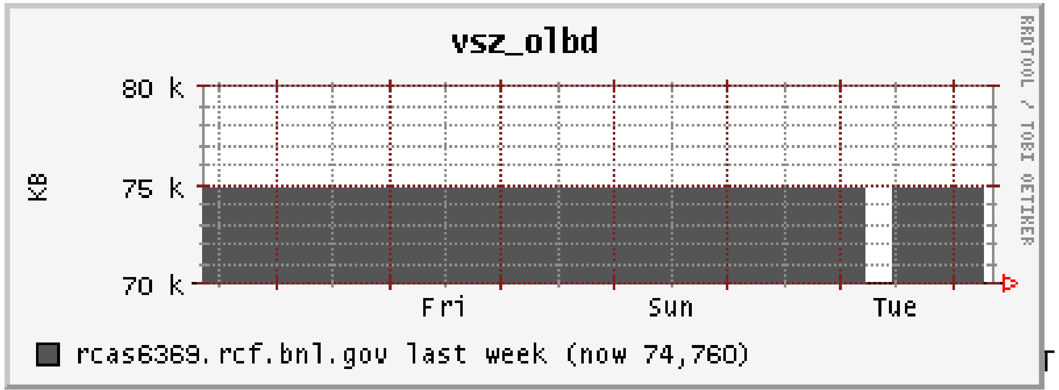
XROOTD provides load balancing, scalability, a way to avoid LFN/PFN translation

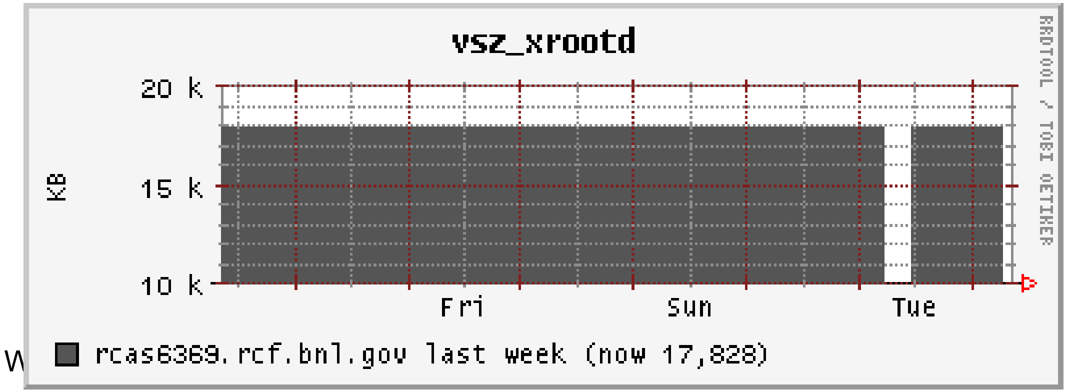
- Deployed on 380 nodes (biggest Xrootd usage?)
  - Needed to wait for the 64 node limitations removal (reported in February, available ~ in April/May)
- Warning: moving from ROOTD to Xrootd is not as trivial as it seems in ROOT 4.xx.xx …
  - Different security model, shaky initial implementation [*Should be fixed - Gerri*]
  - ROOTD does only PFN, Xrootd cannot do both PFN and LFN [*in progress??*]
  - Several patches sent to the Xrootd team

Xrootd stability

- Xrootd provides load balancing, scalability. A way to avoid LFN/PFN translation ..
  - ○ Initial version was not stable enough for us (end of 2004) …
  - ○ Needed to wait for the 64 nodes limitation removal (reported in February, available ~ in April/May)
  - ○ Warning: moving from ROOTD to Xrootd is not as trivial as it seems in ROOT 4.xx.xx …
    - Different security model, erroneous initial implementation [*Should be fixed - Gerri*]
    - ROOTD does only PFN, Xrootd cannot do both PFN and LFN [*in progress??*]
  - ○ Several patches sent to the Xrootd team

- Deployed on 380 nodes (biggest Xrootd usage?)
  - ○ Stable enough now for sure …
  - ○ **We are ready to go along ROOTD -> XROOTD ALL THE WAY !!!!**

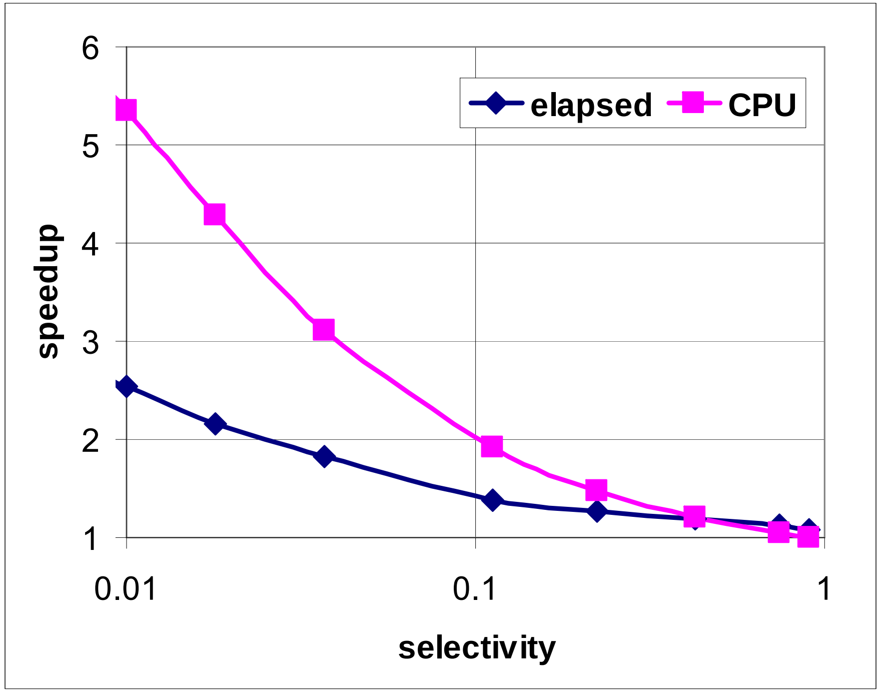




GridCollector

```
root4star -b -q doEvents.C'(25,"select MuDst where Production=P04ie \
and trgSetupName=production62GeV and magScale=ReversedFullField \
and chargedMultiplicity>3300 and NV0>200", "gc,dbon")'
```

- **Usage in STAR – c.f. Kesheng/John Wu**
  - Based on TAGS produced at reco time
  - Rest on now well tested and robust SRM (DRM+HRM) deployed in STAR anyhow
    - Immediate Access and managed SE
    - Files moved transparently by delegation to SRM service
  - Easier to maintain, prospects are enormous
    - “Smart” IO-related improvements and home-made formats no faster than using GridCollector (a priori)
      - **Physicists could get back to physics**
      - **And STAR technical personnel better off supporting GC**



- **It is a WORKING prototype of**
  - Grid interactive analysis framework
  - **VERY POWERFULL Event “server” based (no longer files)**

Summary

- **Is ROOT great ??**
  - I think it fair to say **YES**
  - Fair to mention that I had at maximum 4 “major” issues in 4 years (resolved within 24 hours, special thanks to Philippe, Rene, …)
    - Outstanding support and outstanding team
  - Fait to say that it provided all features needed for data processing, framework, analysis, …

- Could it be better ? Always …
  - We need more display capabilities
    - Qt like work important (to us and other efforts like QScan, …)
    - Geometry work (Andrei, …) should be a priority …
  - We need MORE distributed computing aware capabilities / Grid
    - Managing SE (Xrootd, perhaps GridCollector …)
    - Managing CE, …
    - Ideally, event based a-la-GridCollector is necessary (in the plans?)