

Processing Tevatron Data Around the World: *Lessons Learned*

Adam Lyon (lyon@fnal.gov)
Fermilab Scientific Computing Division

Argonne National Laboratory APS
September 19, 2012

What's this about?

The Tevatron experiments have been processing enormous amounts of data world wide for over 12 years

There are many lessons we've learned

The most important: The difficulties of data management are easy to underestimate; they still surprise us!

Who am I?

With Fermilab since 2002

**Running Experiments Department Data
Handling Group Leader**

Project Manager for “SAM” 2005-2011

Scientist on D0 and g-2 Experiments

Outline of this talk

An aerial night photograph of a large circular construction site. The site is illuminated by various lights, including green and blue spotlights. Light trails from traffic are visible around the perimeter of the site. The background shows a city skyline with lights under a dark sky.

Some context

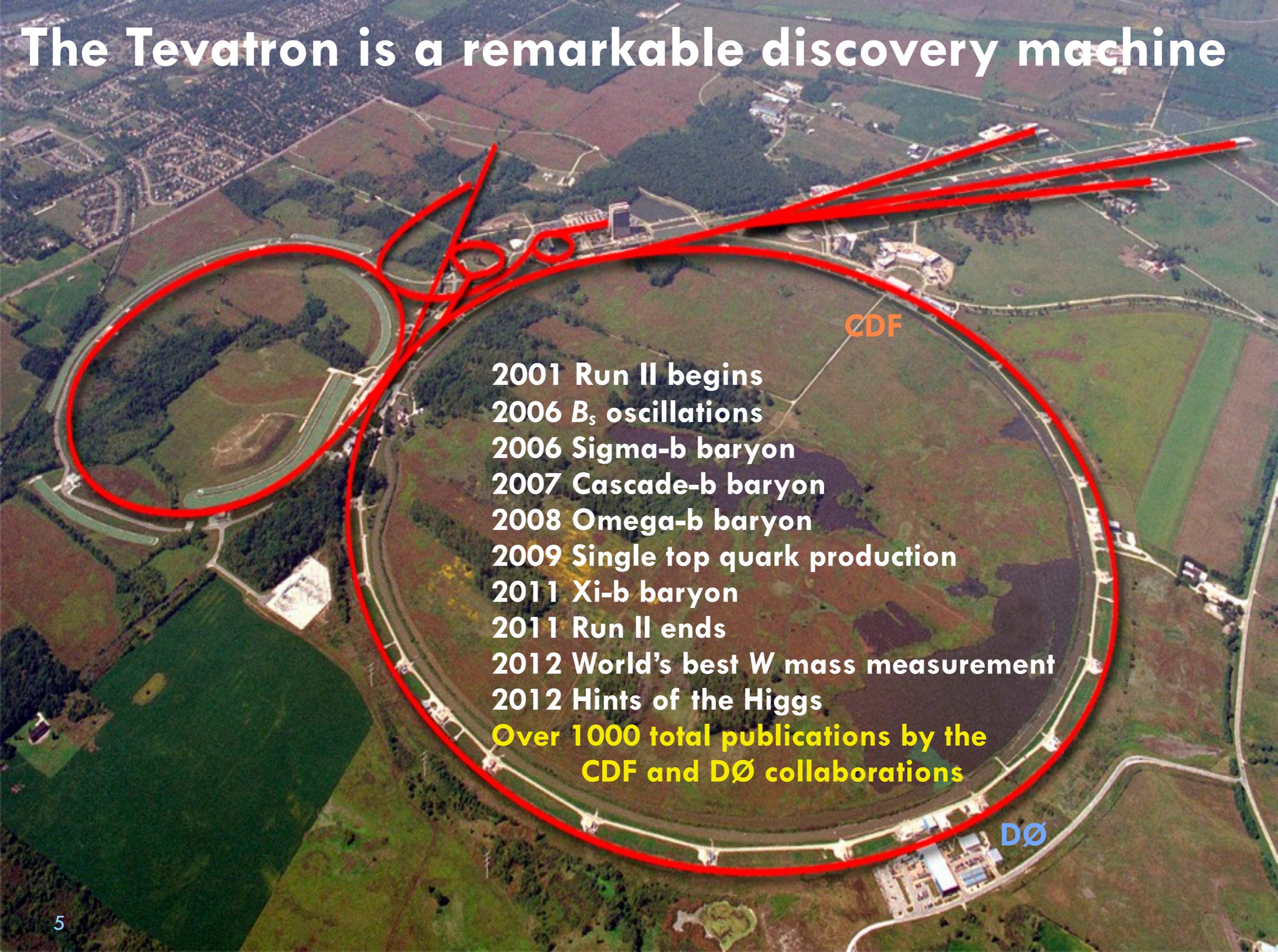
Requirements

Implementation

Evolution

Future

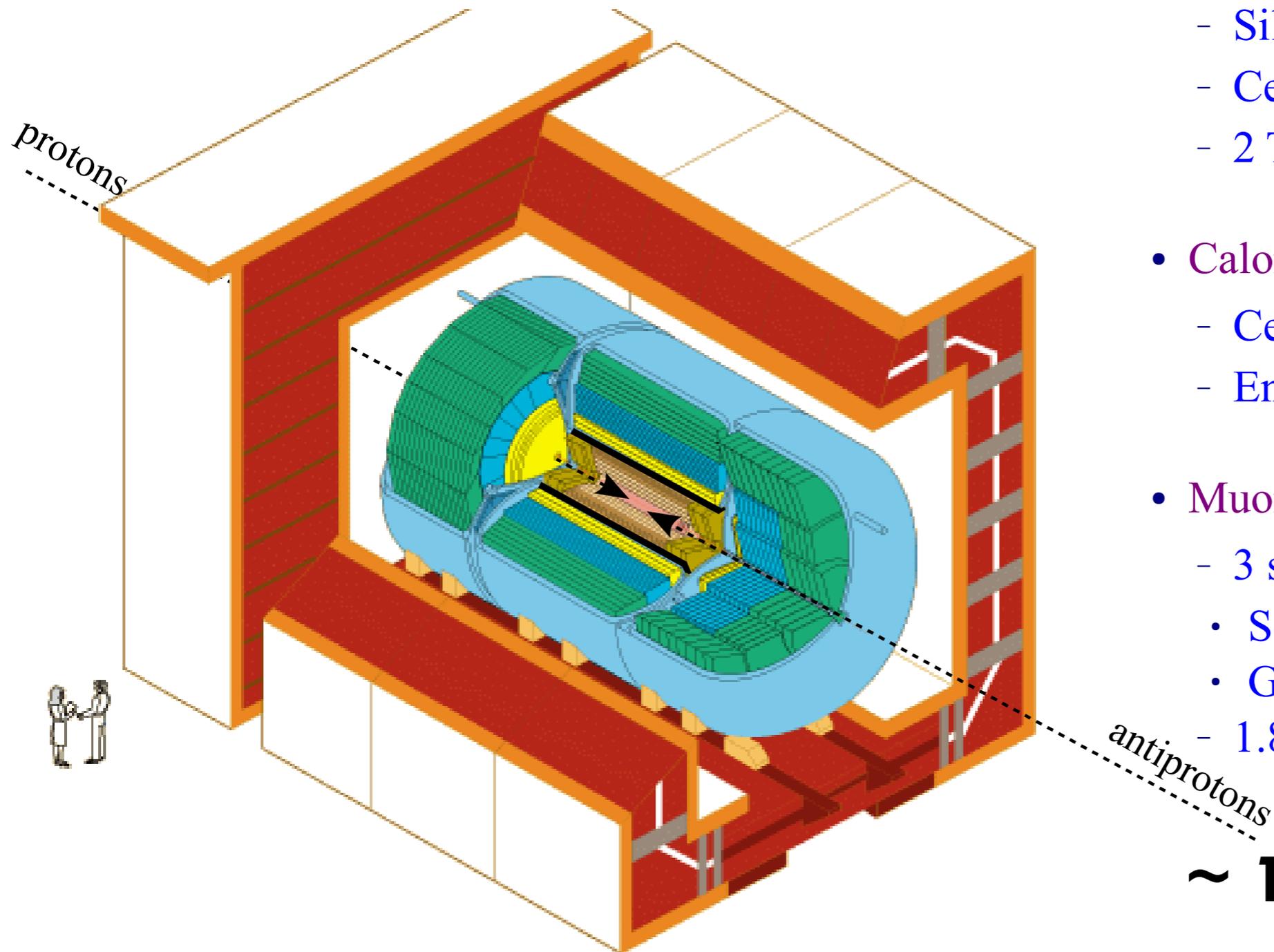
Reveal lessons learned along the way



The Tevatron is a remarkable discovery machine

- 2001 Run II begins
- 2006 B_s oscillations
- 2006 Sigma-b baryon
- 2007 Cascade-b baryon
- 2008 Omega-b baryon
- 2009 Single top quark production
- 2011 Xi-b baryon
- 2011 Run II ends
- 2012 World's best W mass measurement
- 2012 Hints of the Higgs
- Over 1000 total publications by the CDF and DØ collaborations

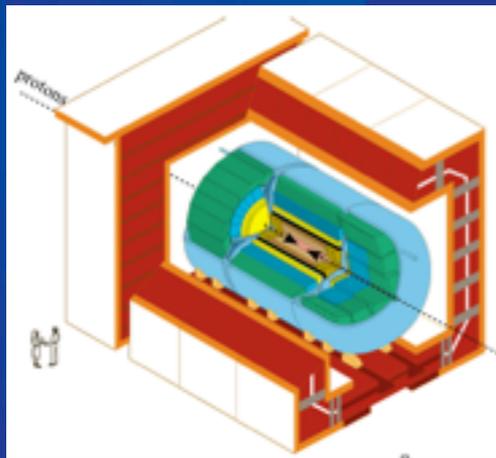
DØ is one of two large detectors at the Tevatron



- Central Tracking System
 - Silicon Micro-strip Tracker
 - Central Fiber Tracker
 - 2 T Solenoid Magnet
- Calorimeters
 - Central Calorimeter (CC)
 - End Calorimeters (EC)
- Muon System
 - 3 sets of detectors
 - Scintillating tiles
 - Gas Drift Tubes
 - 1.8 T Toroid Magnets

~ 1M channels

The processing steps from detector to analysis



Detector

raw



Tape storage

raw

thumbnail (tmb)

tmb

tmb

tmb

root-tree

root-tree

root-tree

root-tree

Reconstruction

Correct & Skim

Make CAF Trees

Select data

Analysis

Plots, Results, Thesis, Paper,
Trip to Sweden

Other work steps

Reprocessing – go back to raw data and reprocess again with new reconstruction software

Monte Carlo production – Produce simulated events and overlay real detector noise

DØ typically does reconstruction and analysis on Fermilab resources and Monte Carlo production offsite

Big Reprocessings done offsite

Behind the scenes of a processing step

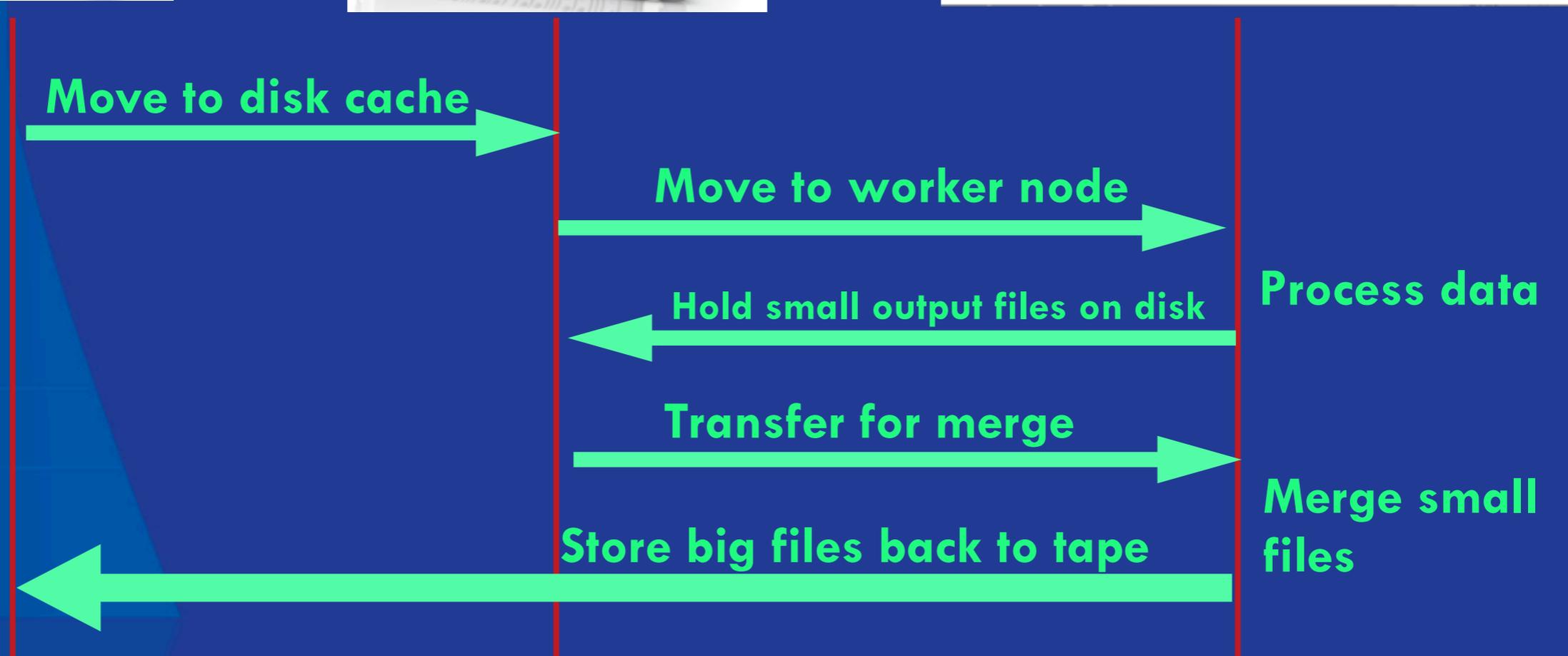
Tape storage



Disk



Compute farm



Requirements of a data management system

Catalog data files with metadata

Query and group files into datasets

Maintain file status

Track locations of files (tape & disk)

Manage disk cache

Reliably transfer files from disk to tape and disk to disk

Store files onto tape

Track file consumption

Implementation: SAM

Sequential Access to data via Metadata

A comprehensive data management system

Started as a CD project for DØ in 1998 and continues today

Lots of developers:

2 database, 4 original DØ developers, +3 when added grid, +3 when added CDF ... now down to ~1.5

Shifters from the experiment for operations

Code in C++ and Python

Close integration with experiment framework C++ code

One code base for everyone <-- **has worked out well**

Place focus on metadata

De-emphasize the file name

Store metadata: Data tier, skim name, application name/
version, monte carlo production details, data collection details,
dates, # of events

Lesson learned: It's hard to keep metadata out of the filename

Create dataset definition with a metadata query

Datasets are dynamic (grow with new files, shrink when files go “bad”)

A project is a set of jobs processing files in a dataset

The snapshot is the list of files when the project starts and is static

File status is extremely important

Availability status: Bad means that file is not available anywhere (not cached and tape is out of service). Automated

Content status: Bad means that the contents of the file is, for some reason, invalid (usually pilot error when configuring an application). Completely manual

Lesson learned: The two part status has been extremely successful

Never “delete” files from the system

Lesson learned: Being unable to recycle file names was a hinderance. Added file “retirement” to cope

Throttling is a crucial component

In the early days, systems were fragile. Too many users requesting files caused big problems

SAM introduced many throttles to file access:

of files a node can have requested

of files a project can have requested

Careful pre-staging of files (“excess satisfaction”)

Lesson learned: Integrating throttling with efficient tape operations is difficult; some inefficiency will remain

Lesson learned: Even as systems become robust, fragility can remain and throttling remains necessary [Bluearc]

Lesson learned: File delivery is not deterministic. Get over it!

Data integrity is crucial

Tape system generates a CRC when file is stored. CRC is recorded in SAM database

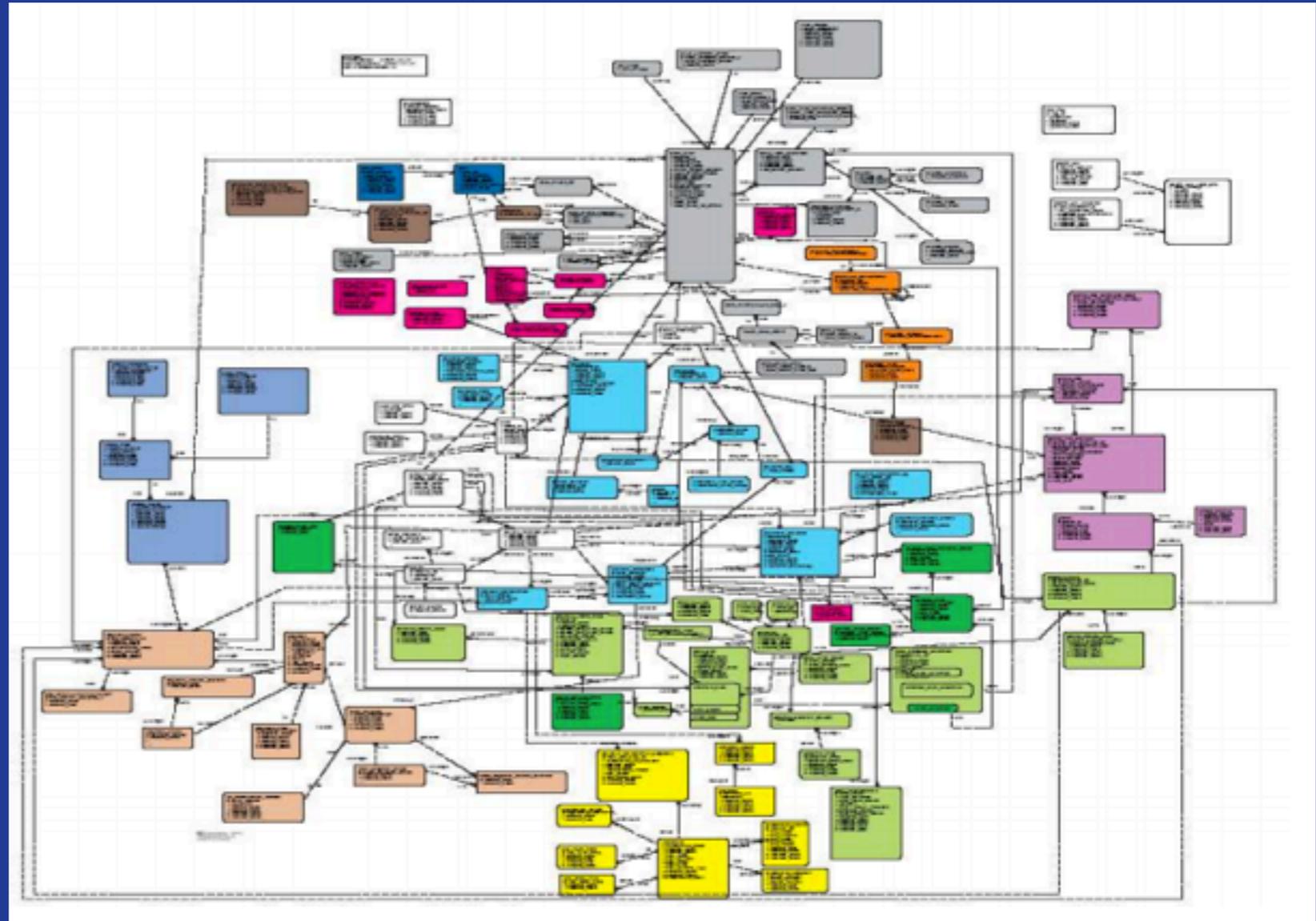
SAM checks the CRC after every file transfer

We catch a few corrupted transfers every year (rate seems to remain constant, though # of file transfers rapidly increase)

Lesson learned: CRC checks add time, but checking is absolutely critical for data integrity

The “SAM” Project implements the data management requirements

Oracle database for metadata catalog, replica catalog, cache status, “project” tracking, “process” tracking, consumption status, configuration



Lesson learned: Exclusive use of one database has plusses (query tuning, support) and minuses (hard to port)

A short glossary

Job - running an application on a batch system in a batch slot

SAM Project - A group of possibly parallel jobs requesting files from the same snapshot

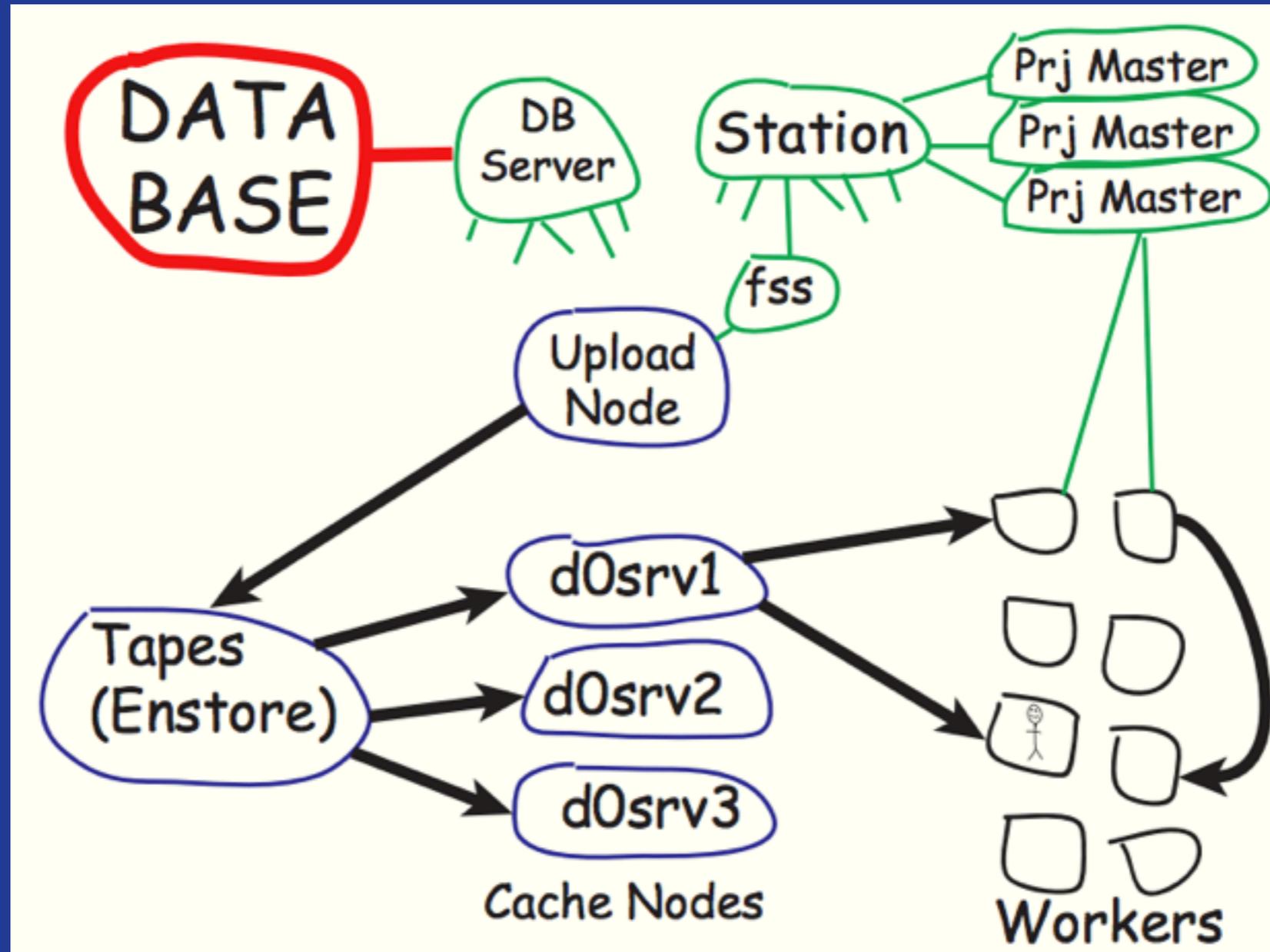
SAM Project Master - A service that manages file delivery for a project

SAM Process - corresponds to an individual job receiving and processing files

SAM Station - A service that manages all cache and projects (an experiment can have many stations)

Lesson learned: Grouping jobs into projects has been very successful

SAM System cartoon



**Interprocess communication all with CORBA
(was the best choice in 1999)**

Lesson learned: Protocol choices are hard to change later, but possible to “wrap”

Data Handling specs for DØ

Plan from 1999

Entire dataset: 0.5 - 1 PB

Tapes: 50 GB, 6 MB/s

Cache size: 20 TB

files in catl: 1 million

events in catl: few billion

Data Handling specs for DØ

	Plan from 1999	Actual in 2012
Entire dataset:	0.5 - 1 PB	8.2 PB
Tapes:	50 GB, 6 MB/s	800 GB, 120 MB/s
Cache size:	20 TB	760 TB
# files in catl:	1 million	140 million
# events in catl:	few billion	abandoned (too big)

**Lesson learned: Think big and prepare for more!
Scalability is crucial!**

Be ready for eco-system changes

Introduction of “Big Cache”

Introduction of “file spreading”

Lesson learned: Monitoring is crucial (and hard to get right). Be ready for conditions to change over time, necessitating action

Lesson learned: Manipulating the cache by hand (e.g. pinning files) leads to headaches; leave it automated

Lesson learned: Automate everything!

Monitoring is crucial

fnal-cabsrv1 station

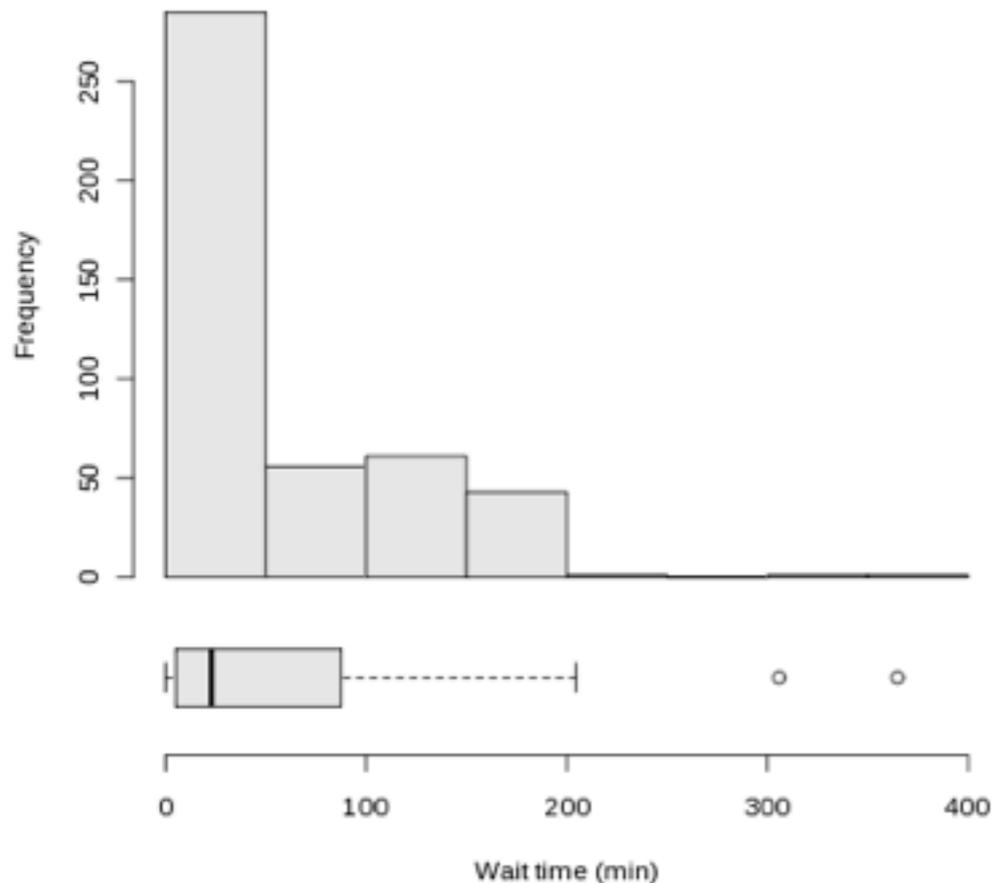
Generated at 2012-09-18 01:36:04

There are 1156 running projects

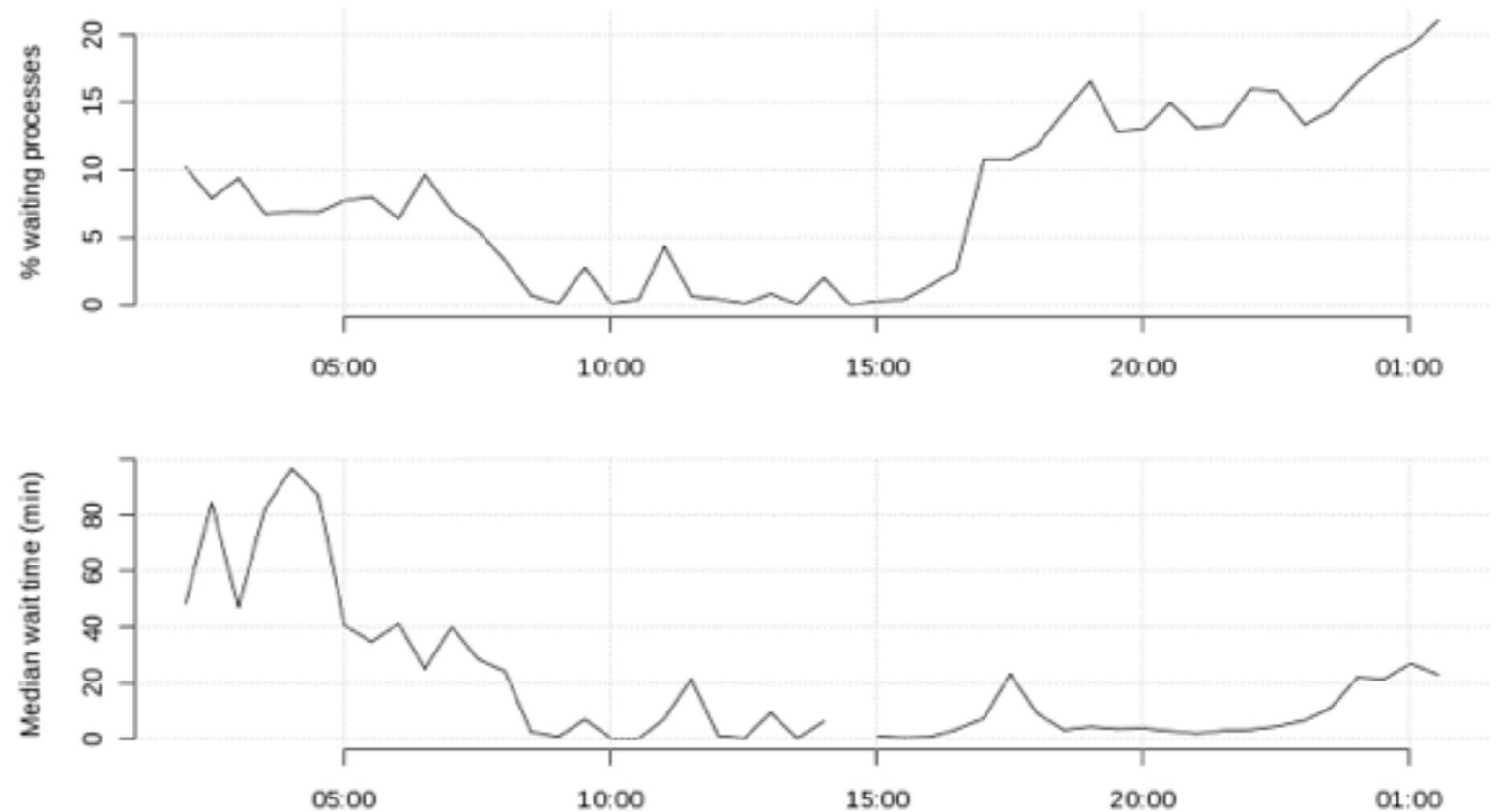
Active processes	2130
Waiting processes	448 (21.03%)

Important to understand waiting processes

Waiting process times



Waiting process history

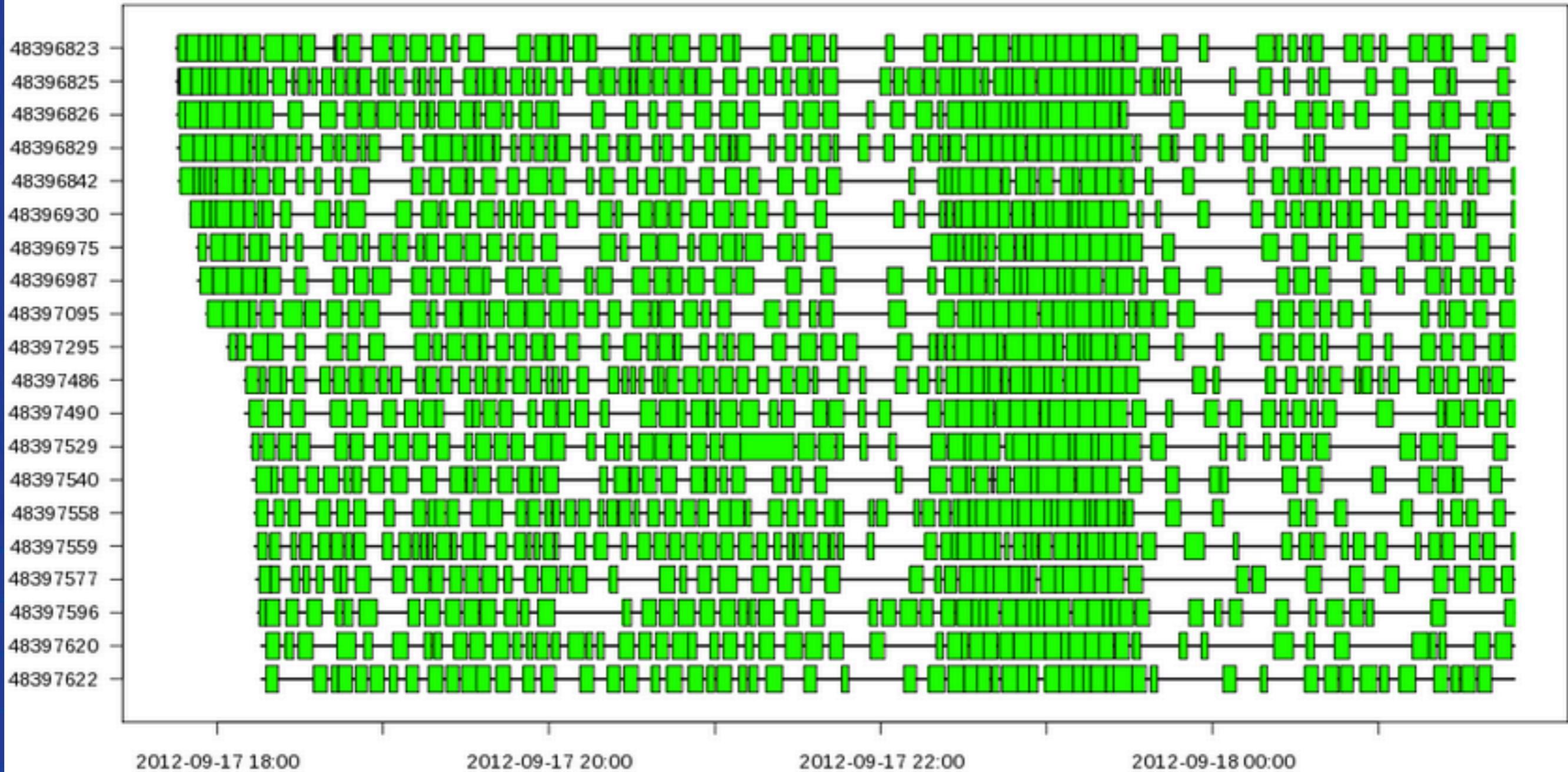


Monitoring individual jobs

Project name ▲▼	Project Id ▲▼	Owner ▲▼	Status ▲▼	Files in snapshot ▲▼	Files seen ▲▼	Processes ▲▼	Waiting processes ▲▼	Last activity ▲▼	Longest waiting process ▲▼	Mean wait time (per file) ▲▼	Mean busy time (per file) ▲▼
CSG Fix Upsilon_p20.18.05_2b3_p20.18.05_02-26-2010-12-14-09-17092012-234458	6480549	csgprod	running	145	0	0	-	project started at 2012-09-18 01:39:36	-	-	-
CSG Fix Upsilon_p20.18.05_2b2_20090221_makeup_20120917135708-17092012-234037	6480548	csgprod	running	1	1	1	-	file delivered at 2012-09-18 01:38:27	-	31s	-
CSG Fix Upsilon_p20.18.05_2b3_p20.18.05_02-27-2010-20-36-37-18092012-011912	6480547	csgprod	running	111	18	33	15	file delivered at 2012-09-18 01:32:04	7 minutes (process started)	2min 59s	-
CSG Fix Upsilon_p20.18.05_2b2_20080130_makeup_20120917135706-17092012-214219	6480543	csgprod	running	1	1	1	-	file delivered at 2012-09-18 00:55:23	-	43s	-
CSG Fix Upsilon_p20.18.05_2b3_p20.18.05_02-26-2010-01-26-02-17092012-212943	6480542	csgprod	running	32	16	16	-	file delivered at 2012-09-18 00:52:46	-	1min 30s	-
guochen_8167_20120918003650	6480541	guochen	running	11	6	1	-	file delivered at 2012-09-18 01:19:45	-	17s	7min 43s
subhendu_10033_20120917213916	6480540	subhendu	running	19317	21	20	-	file delivered at 2012-09-18 01:45:51	-	0s	8min 28s
subhendu_8781_20120917213846	6480539	subhendu	running	2827	96	80	-	file delivered at 2012-09-18 01:28:37	-	1s	16min 58s
ayesh_9085_20120917191015	6480538	ayesh	running	5	4	1	-	file delivered at 2012-09-18 01:29:12	-	10s	26min 0s
ayesh_11806_20120917191012	6480537	ayesh	running	5	2	1	-	file delivered at 2012-09-18 00:53:43	-	12s	43min 55s
ayesh_4746_20120917191008	6480536	ayesh	running	5	4	1	-	file delivered at 2012-09-18 01:26:34	-	6s	25min 51s
ayesh_30384_20120917190955	6480535	ayesh	running	5	4	1	-	file delivered at 2012-09-18 01:26:36	-	6s	26min 11s
ayesh_27804_20120917190959	6480534	ayesh	running	5	4	1	-	file delivered at 2012-09-18 01:29:37	-	5s	27min 15s
ayesh_13922_20120917191003	6480533	ayesh	running	5	4	1	-	file delivered at 2012-09-18 01:24:58	-	5s	25min 43s

Monitoring individual jobs

File busy time by process



The “SAM” Project Events

Started as a Fermilab Computing Division project for DØ in 1998

Introduced to CDF in 2003

DØ Reprocessing in 2005 (first large-scale use of the “Grid”) 1.5B events, 82K files, ~82 TB

DØ Reprocessing in 2007 (use of Open Science Grid and LHC Computing Grid) 1.2B events

Monte Carlo production on OSG and LCG

SAM makes a Grid for offsite processing (SAMGrid)

Run offsite with a consistent and central interface
-- sounds like the Grid, but before the Grid was real

We wrote our own middleware and “batch adapters”



Very painful to operate and maintain. Software very fragile. Needed students to help. We had to debug site issues.

Lesson learned: Be pioneers only when you have no other options. At that time, we had no other options.

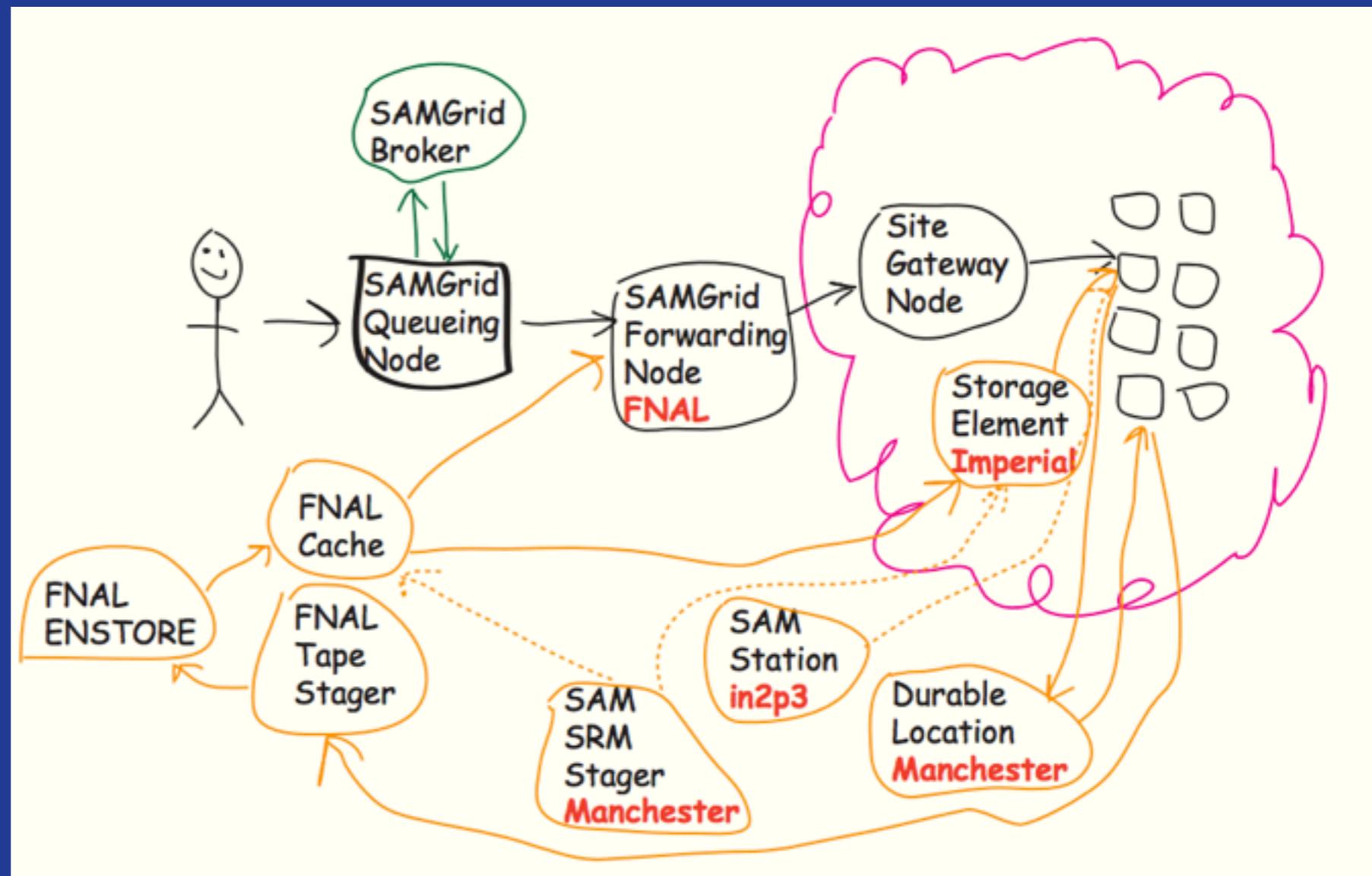
Adopted standard Grid tools

We still have a central interface, but now to OSG and LCG. Adapted SAM to use Grid Storage.

We use
GlideinWMS

OSG and LCG
oversee site
maintenance

Use polling to
overcome
worker node
access restrictions



Recovering when things go wrong

**Jobs accumulate output on the worker node
(usually an output file is the result of reading several input files)**

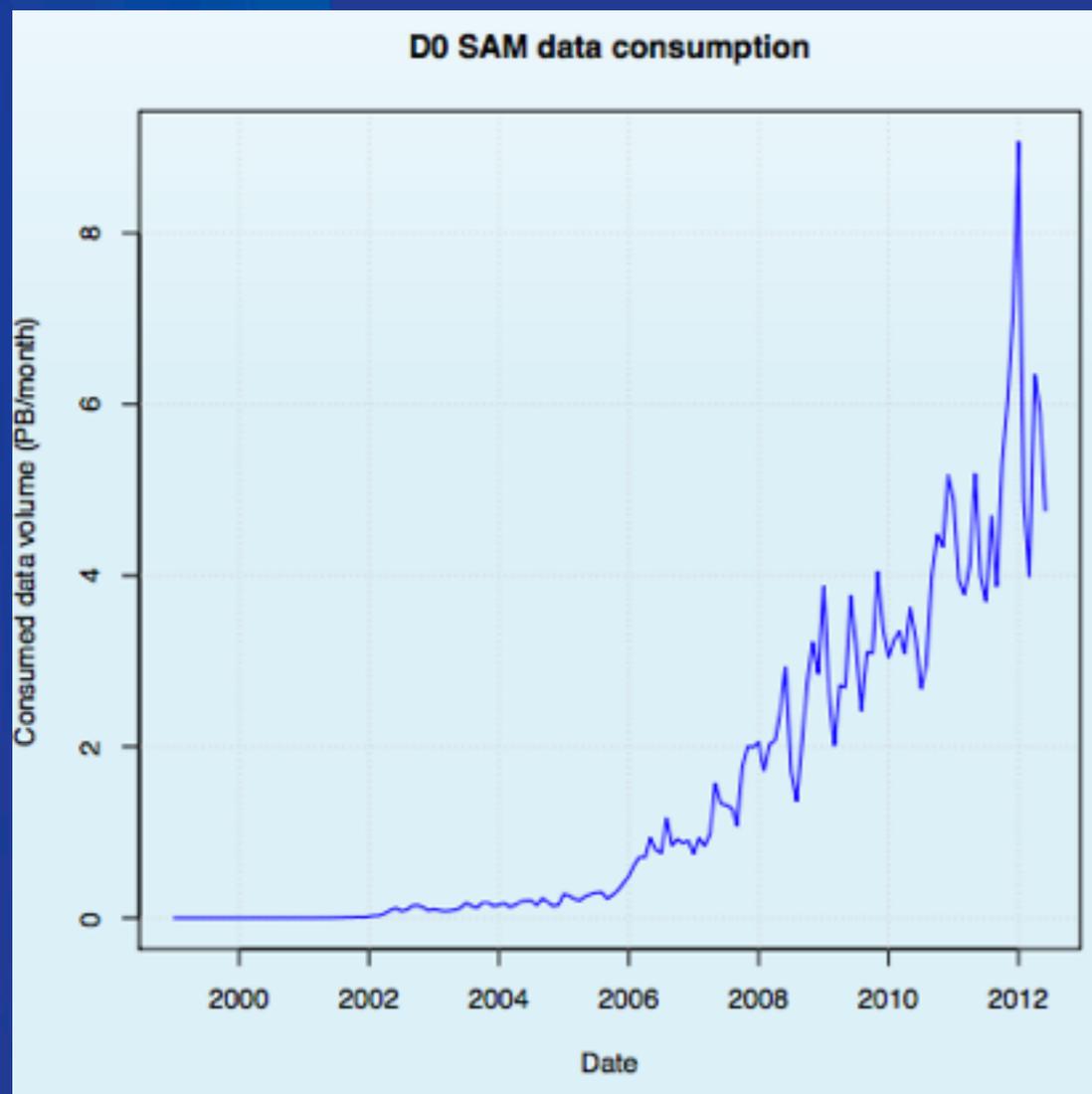
**What happens if some jobs are successful but others fail
(crashes, dies, lost output)? What happens if a file can't be
delivered?**

**SAM keeps track of all unfulfilled transfer requests (unconsumed
files). Can run a recovery project to try these files again**

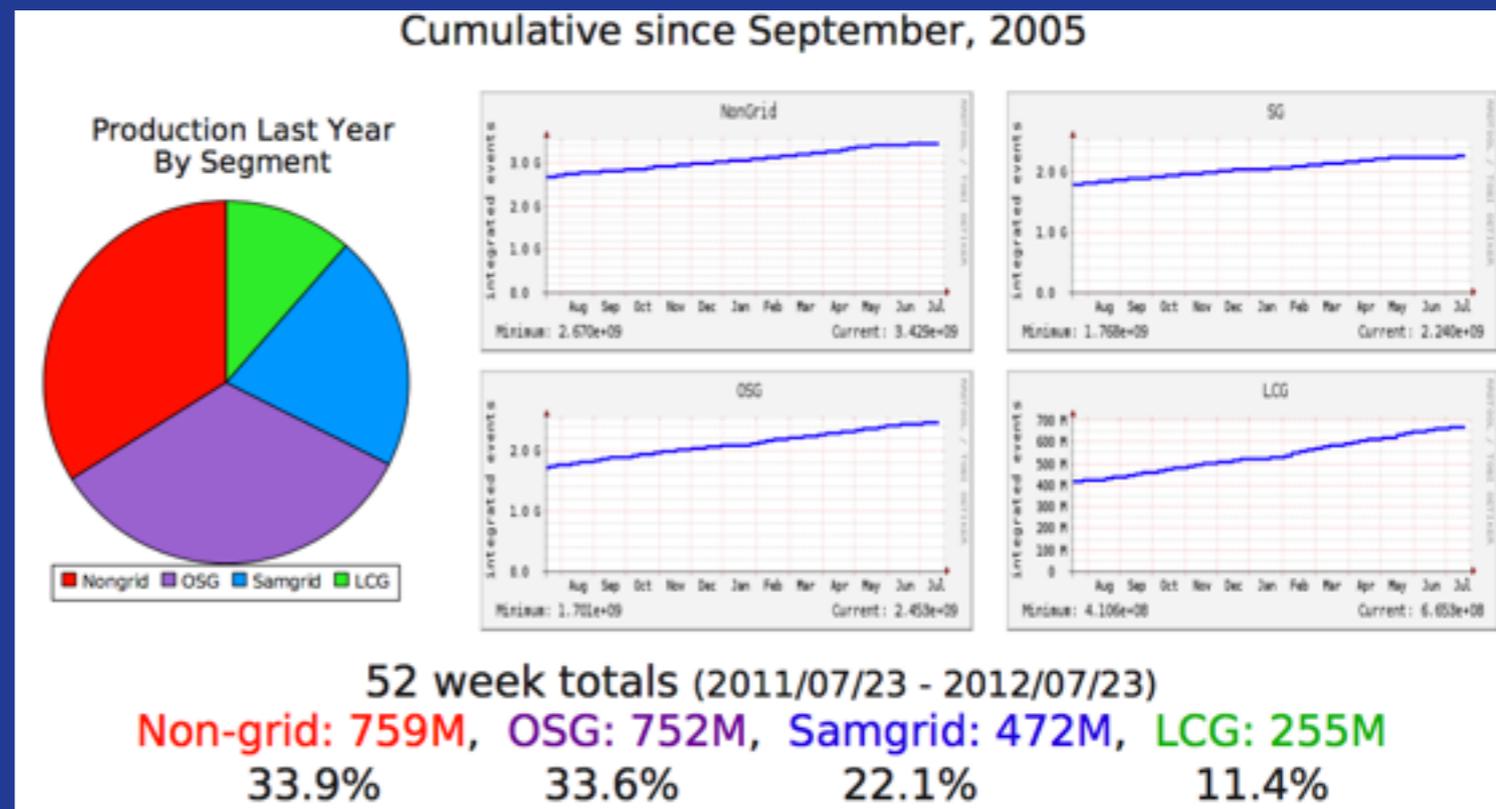
**SAM also tracks if a job reports successful completion. For those
that don't, users can run a recovery project to re-process the
input files corresponding to lost output**

**Lesson learned: Users appreciate these features. Important to
think about recovery when failures occur**

Some statistics



All time total of ~ 250 PB consumed



Monte Carlo Production
8.75B events produced total

~40% on OSG+LCG

We still have a powerful old-style SAMGrid site in the Czech Republic

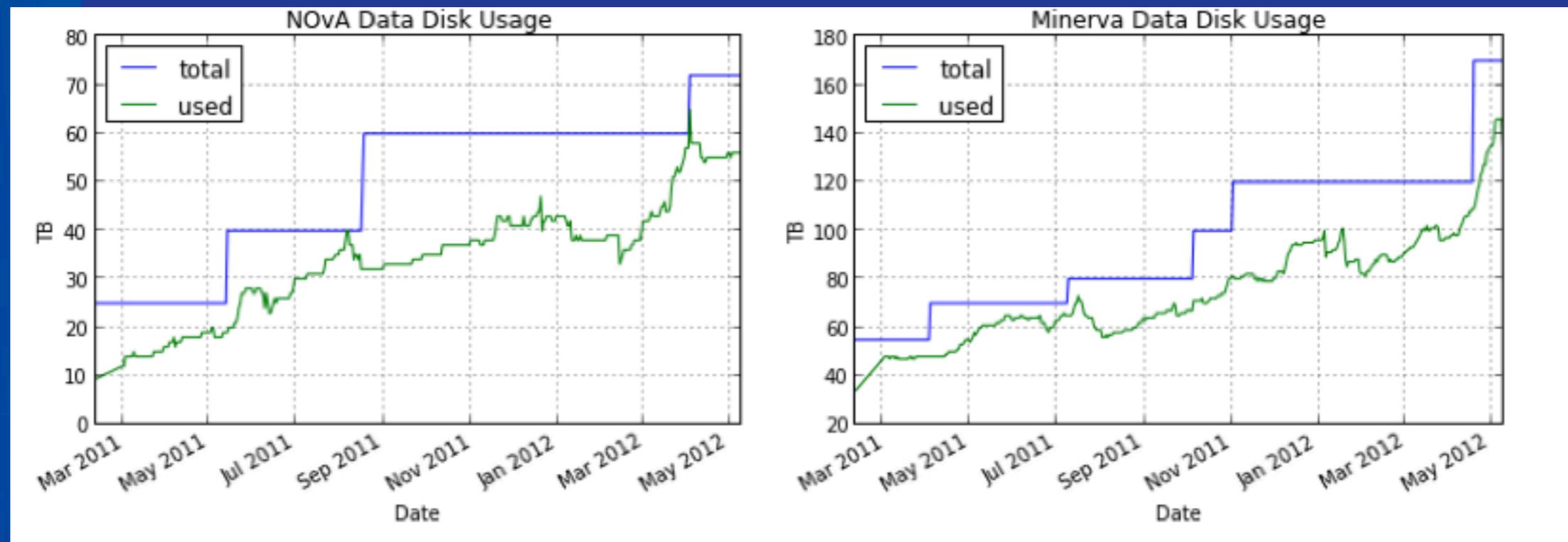
France hosts a big compute farm for special MC requests (non-grid)

SAM's future

Fermilab is transitioning to smaller specialized experiments at the *Intensity Frontier (IF)*

New experiments are 100 TB - PB scale data production

Remember underestimating data management? We see this from the new experiments (“We’ll just put everything on disk” - having a big SAN yields false security).



Introducing SAM, but with improvements to make it more palatable

Fermilab at the Intensity Frontier

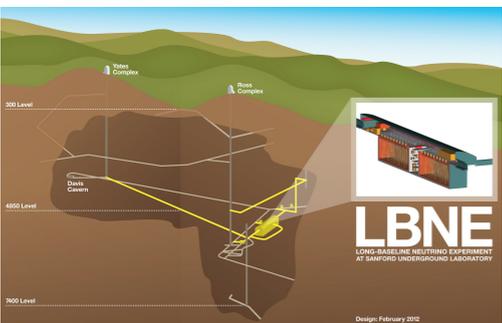
Operating: MINOS+, Minerva



Constructing: NOvA



Planning: Mu2e, g-2, LBNE



New challenges:

More experiments, more diverse

Though experiments are smaller than CDF and D0, data sizes can be large

Experiments have fewer computing experts

Experiments tolerate sharing resources

Try to provide common solutions (mantra of the SCD)

Also interest from D0 & CDF Data Preservation teams

Data Handling for IF experiments

We soul searched whether to continue with SAM, adopt something else, or start writing something new

Decided that there's no other product like SAM (LHC data handling systems are not generic enough and use a different data handling philosophy)

Decided it would be too much work to embark on something new, especially since SAM is working extremely well

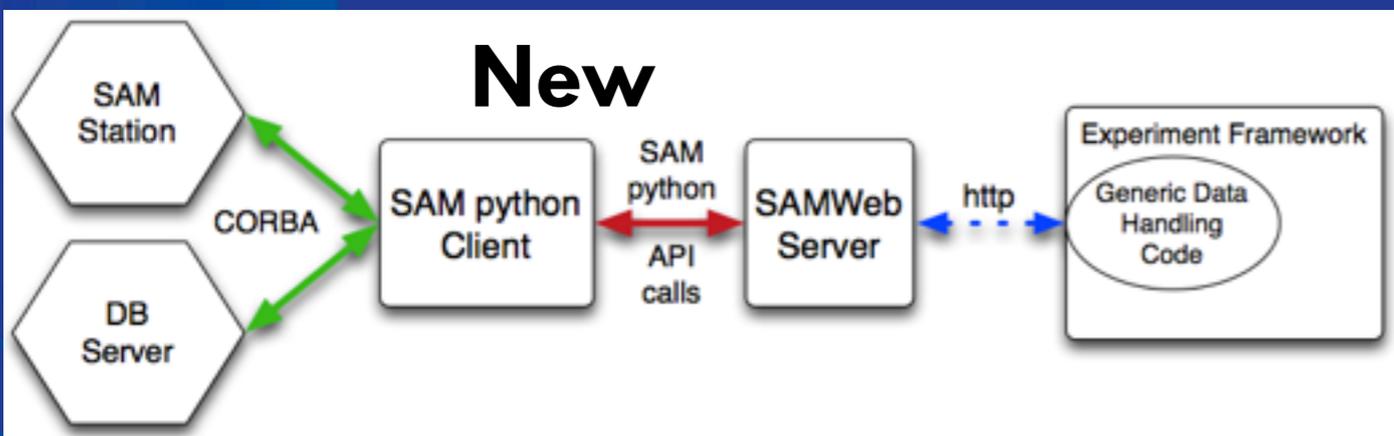
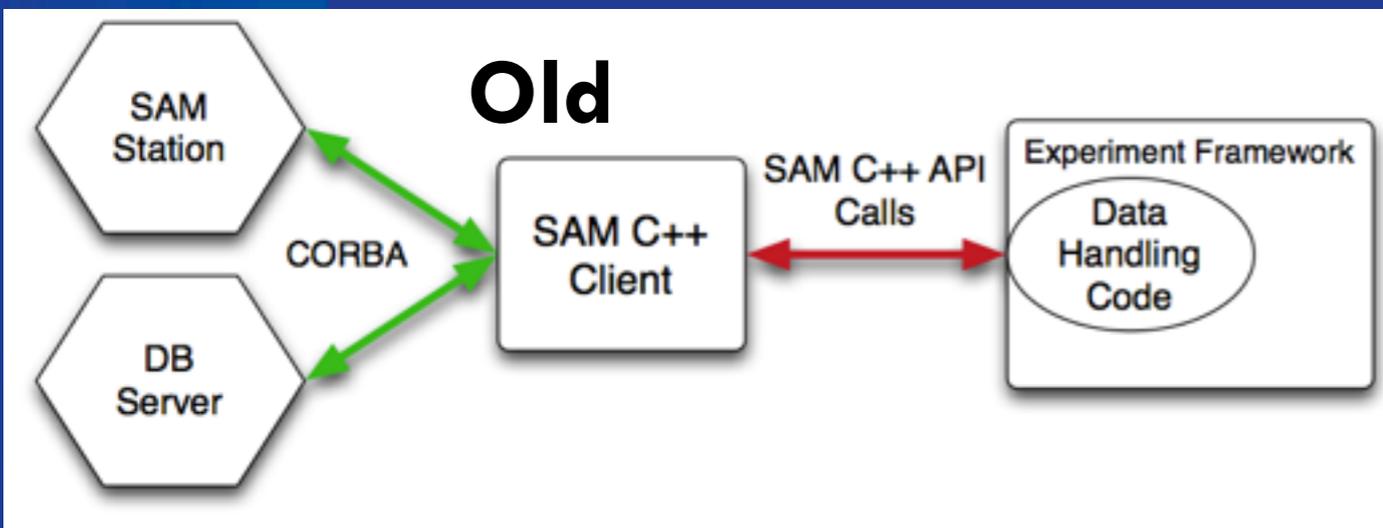
Decided to leave the SAM internals as they are, but wrap over the warts

Two big warts:

- 1) Integration with experiment frameworks**
- 2) User interface, especially for file transfers to your desktop**

SAMWeb & SAMfs

Dramatically improve SAM + Experiment Framework Integration and our deployment burden



Make SAM behave like a file system with FUSE

- Navigate metadata as directories

```
ls -F /samfs/run
ERROR-supply-a-run-number-like-200000@

> ls -F /samfs/run/268211 # will list sub-runs
111/ 112/ 113/

> ls -F /samfs/run/268211/111 # will list data-tiers
raw/ reconstructed/

> ls -F /samfs/run/268211/111/raw # will list files
raw_nu_268211_111_1.root* raw_nu_268211_111_2.root*
raw_nu_268211_111_3.root*
```

- Navigate pre-defined datasets

```
> ls -F /samfs/dataset
ERROR-supply-a-user-name@

> ls -F /samfs/dataset/lyon # list my datasets
my-special-run-15593/ oscillated-nu-bars-5/
bad-tracking-examples/

> ls -F /samfs/dataset/lyon/my-special-run-15593/
reconstructed_nubar_15593_112_4.root*
reconstructed_nubar_15593_112_5.root*
```

- Load file to cache with "get"

```
> ls -F /samfs/run/268211/111/raw
raw_nu_268211_111_1.root* raw_nu_268211_111_2.root*
raw_nu_268211_111_3.root*

> ls -F /samfs/get/run/268211/111/raw # prohibit ls with get
ERROR-trying-to-list-directory-with-a-GET-path@

> cat /samfs/get/run/268211/111/raw/raw_nu_268211_111_2.root
# binary output here

> root # Open file directly in root
TFile* f = new TFile("/samfs/get/run/268211/111/raw
268211_111_2.root")
# ... back to shell

> ls -F /samfs/run/268211/111/raw # @ == soft link == cached
raw_nu_268211_111_1.root* raw_nu_268211_111_2.root@
raw_nu_268211_111_3.root*
```

Lesson we hope to learn: Improve by wrapping instead of by scrapping (we hope this doesn't make maintenance difficult, we'll see)

Exporting SAM

Improving documentation

Writing up experiences for Run II

Migrate away from Oracle

Understand non-HEP metadata and use cases

Migrate away from Corba for internal communications

Provide a support and maintenance structure

Summary & Discussion

SAM is a very successful comprehensive data management system

But it hasn't been easy - many lessons learned

Many lessons applicable to others

How do your data handling needs fit in?

Does a facility level data handling system make sense?

Sales pitch: Fermilab SCD is interested exporting SAM to other labs & experiments

BACKUPS

SAMGrid Workflow Management

“DORunJob” is our home written workflow manager

Not very exciting – knows how to generate Monte Carlo data, usually 200 events per job

Scripts for generation, geant, simulation, reconstruction, and Root-tree making

Has hooks for declaring meta-data to SAM

Requests noise overlay files from SAM

Stores output files on “durable locations” for merging

Tape System (Enstore)

For entire systemx

7 tape libraries; 70K slots

~100 LTO4 drives, 53 T10000C drives

47K LTO4 tapes, 2.5K T10000T2 tapes

Over \$6M investment

Why our own tape system? In 1999, tried HPSS and had poor experience. Not clear if problems could be resolved for Run II