A Data Handling System for Modern and Future Fermilab Experiments

Robert Illingworth

Fermilab Scientific Computing Division

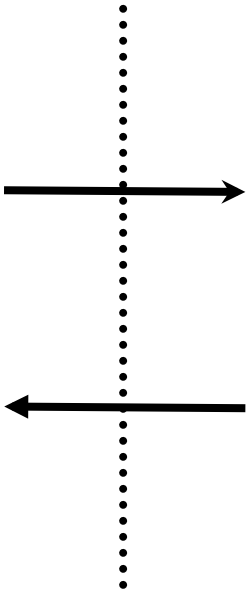
Why is data movement difficult?

Your files don't reside where your jobs run

Your files reside here

Tape Robot

Disk Storage



Not here, where you need them

Compute Farm of worker nodes

Files must be moved to where a worker node can access them with efficiency and scalability

Output files must be returned from worker node

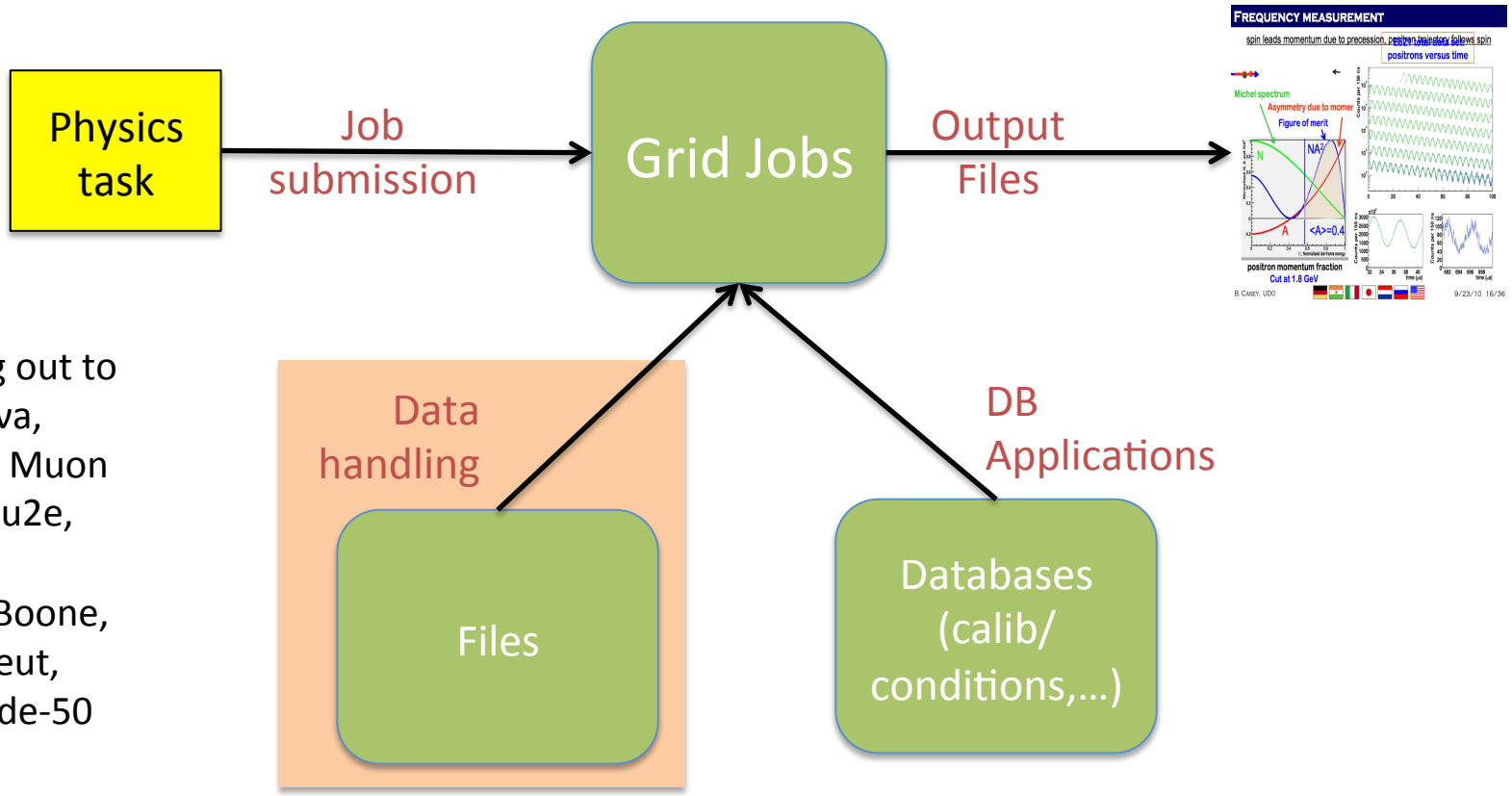
NOT a simple process, especially the efficiency and scalability part!

Supporting many experiments at Fermilab

- Fermilab supports multiple experiments which require data handling services
  - scale ranging from 10s TB/year up to multiple PB/year
- To minimize support requirements, need a common solution
  - It's not practical for every experiment to go off on its own way
- Evaluated various options, including adopting existing system or implementing entirely new one
  - Some existing systems are tightly coupled to experiment computing model. Others are too general.
- Decided to adopt and modernize the existing Fermilab SAM system from Run II of the Tevatron

The FIFE project

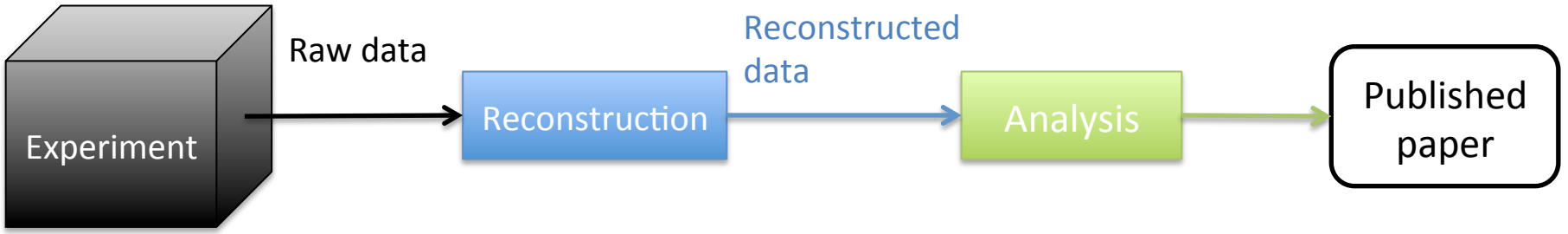
FIFE is Fermilab's overarching program of common tools and systems for scientific data processing (see yesterday's presentation by Mike Kirby)



Rolling out to Minerva, NOvA, Muon g-2, Mu2e, LBNE, MicroBoone, Argoneut, Darkside-50

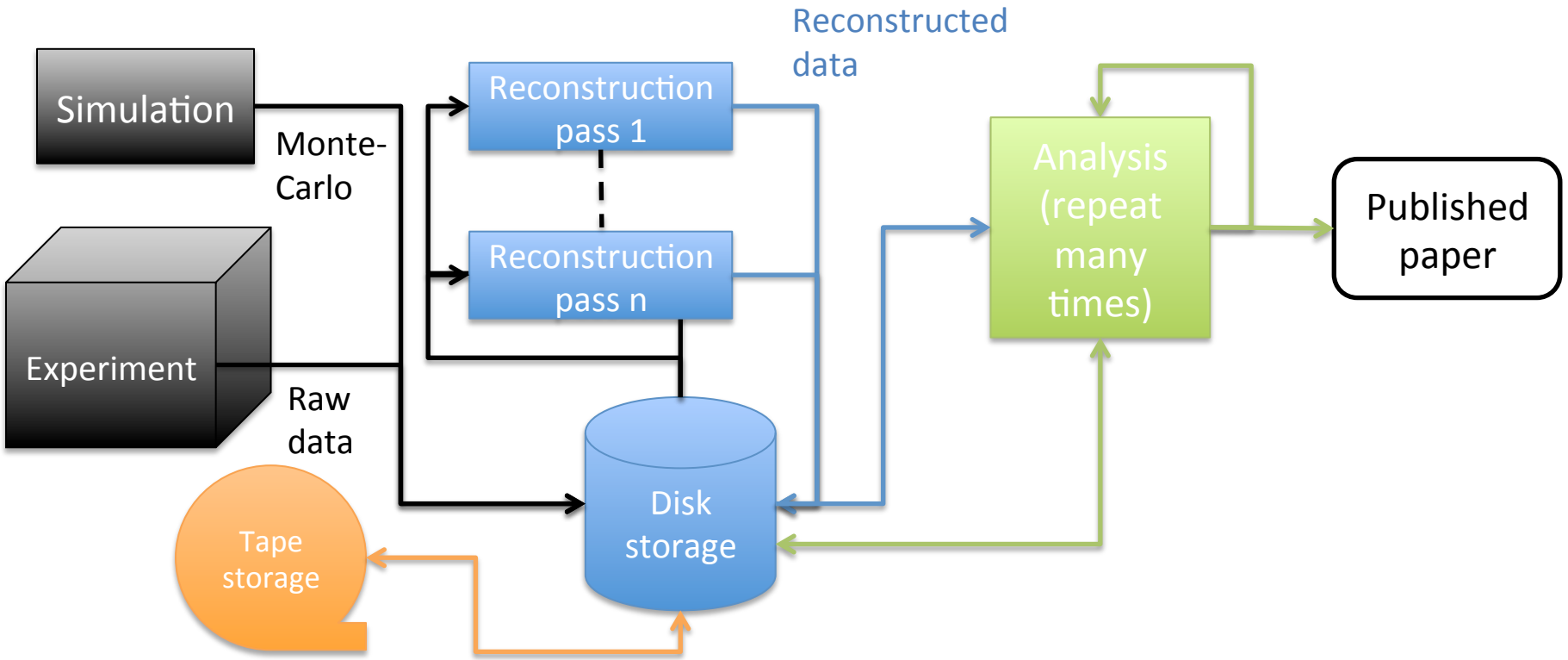
Analysis dataflow (user view)

- Take raw data
- Reconstruct it
- Analyze reconstructed data
- Publish



Analysis dataflow (more realistic view)

- Not usually a nice linear progression
- Even a small HEP experiment has millions of data files to keep track of and feed the correct ones into the next stage of processing



User requirements of a data management system

- What is in my data?
- Give me my data
- What did I do with my data?
- Store my output data safely

User requirements of a data management system

- What is in my data?
  - Metadata catalogue
- Give me my data
  - File access service that turns a dataset into concrete locations
- What did I do with my data?
  - Extensive tracking of job activity; what files were provided to a job; what did the job report was successfully processed
- Store my output data safely
  - Reliably add output files to catalogue and archive them on disk or tape

SAM data management

- SAM (Sequential Access via Metadata) was originally begun as a data handling system for Run II of the Tevatron
- Used very successfully by CDF and D0 (metadata for over 207 million files; over 600 physics papers published)
- Now modernizing the architecture for the next decade of operations
  - build on extensive past experience
  - make use of modern technologies

Metadata catalogue

- Metadata catalogue stores information about the files
  - Physical data (size) and physics data
  - Experiments can define their own fields; not restricted to a predefined set
- Example raw file from NOvA on right

File Name:	fardet_r00011060_s16.raw
File Id:	3929242
File Type:	importedDetector
File Format:	raw
File Size:	539664496
Crc:	2841833047 (adler 32 crc type)
Content Status:	good
Group:	nova
Data Tier:	raw
Application:	online datalogger 33
Event Count:	17465
First Event:	276659
Last Event:	294123
Start Time:	2013-09-01T04:48:38
End Time:	2013-09-01T04:56:00
Data Stream:	all
Online.ConfigIDX:	0
Online.DataLoggerID:	1
Online.DataLoggerVersion:	33
Online.Detector:	fardet

Making datasets

- Individual files are combined into datasets
- Datasets are defined by a simple query language, for example “run_number 11060 and file_format raw”
- Users can create their own datasets; they aren’t confined to predefined ones
- Datasets can be dynamic; new files that match the criteria are included each time they are evaluated

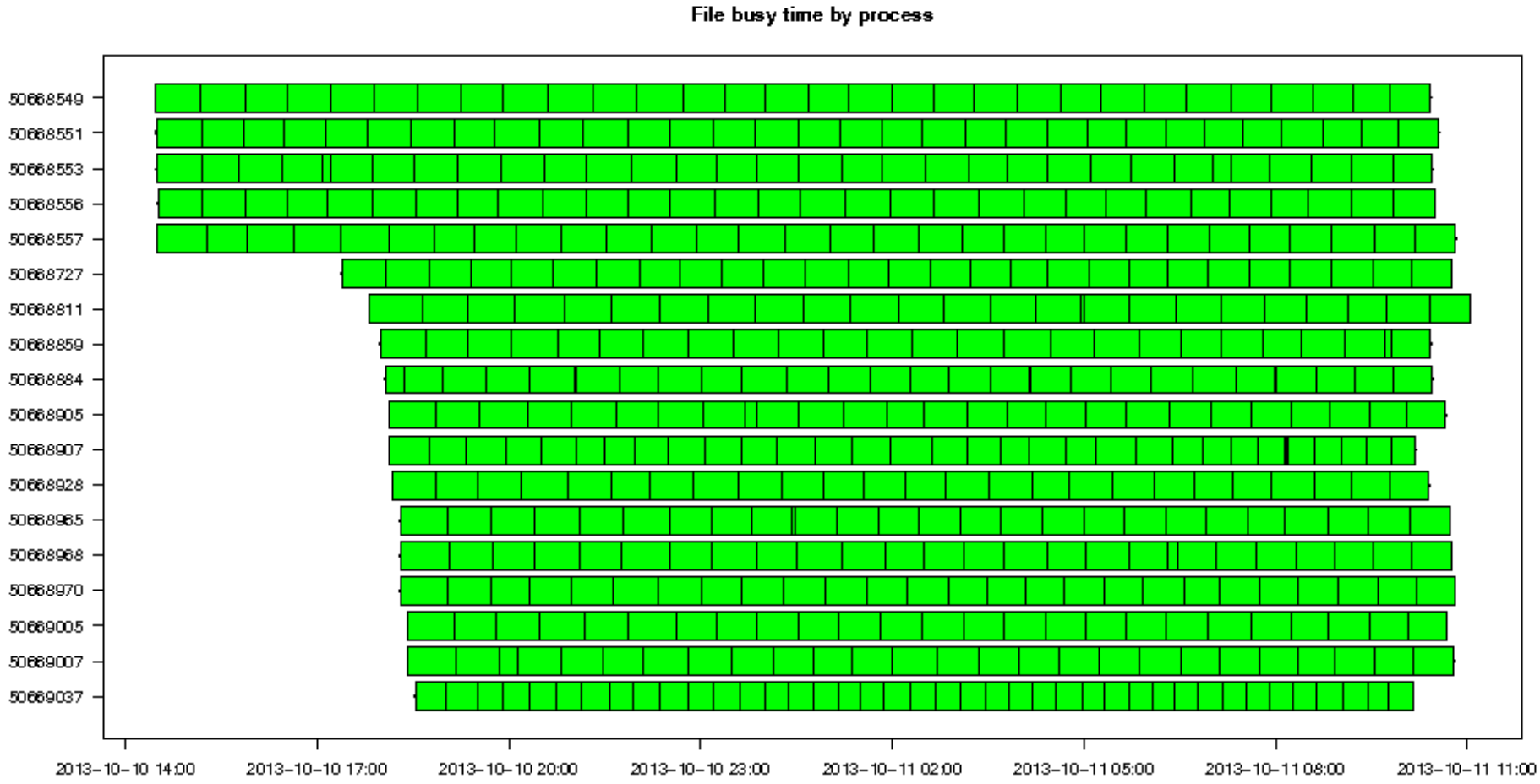
Running jobs

- Jobs only need the dataset name as input; no need for users to know where the files are physically located
- The files will be staged from tape if needed
- Multiple jobs can be run on a single dataset. The files will be dynamically distributed across all the jobs

Tracking and recovery

- Each job updates the database as it opens and closes each file
  - Provides a history of the entire job
- This allows simple recovery of failures; make a new data set that contains:
  - Files not provided due to storage system problems
  - Files which were processed but the output was lost (job crashed, was evicted, transferring the output failed, etc.)

Monitoring of processing activity



Operational requirements

- Designed as a highly-automated, "hands-off" system requiring minimal routine intervention
  - Ideal for the "smaller" (by LHC standards) collaborations, which can't provide dedicated expertise for operating data management systems
- Data staging from tape and from storage element to storage element is automatic
  - Philosophy is "bring the data to the jobs", not "bring the jobs to the data"

SAM as part of FIFE

- Part of an integrated system – FIFE – with other components
  - Jobsub glideinWMS based job submission
  - *art* framework
  - dCache disk cache
- But loosely coupled (via http REST interfaces), so can easily substitute other services

Job submission integration

- The FIFE job submission system (jobsub) takes the SAM dataset name when the job is submitted
- It notifies SAM that the specified dataset is needed, and passes contact information to the individual jobs

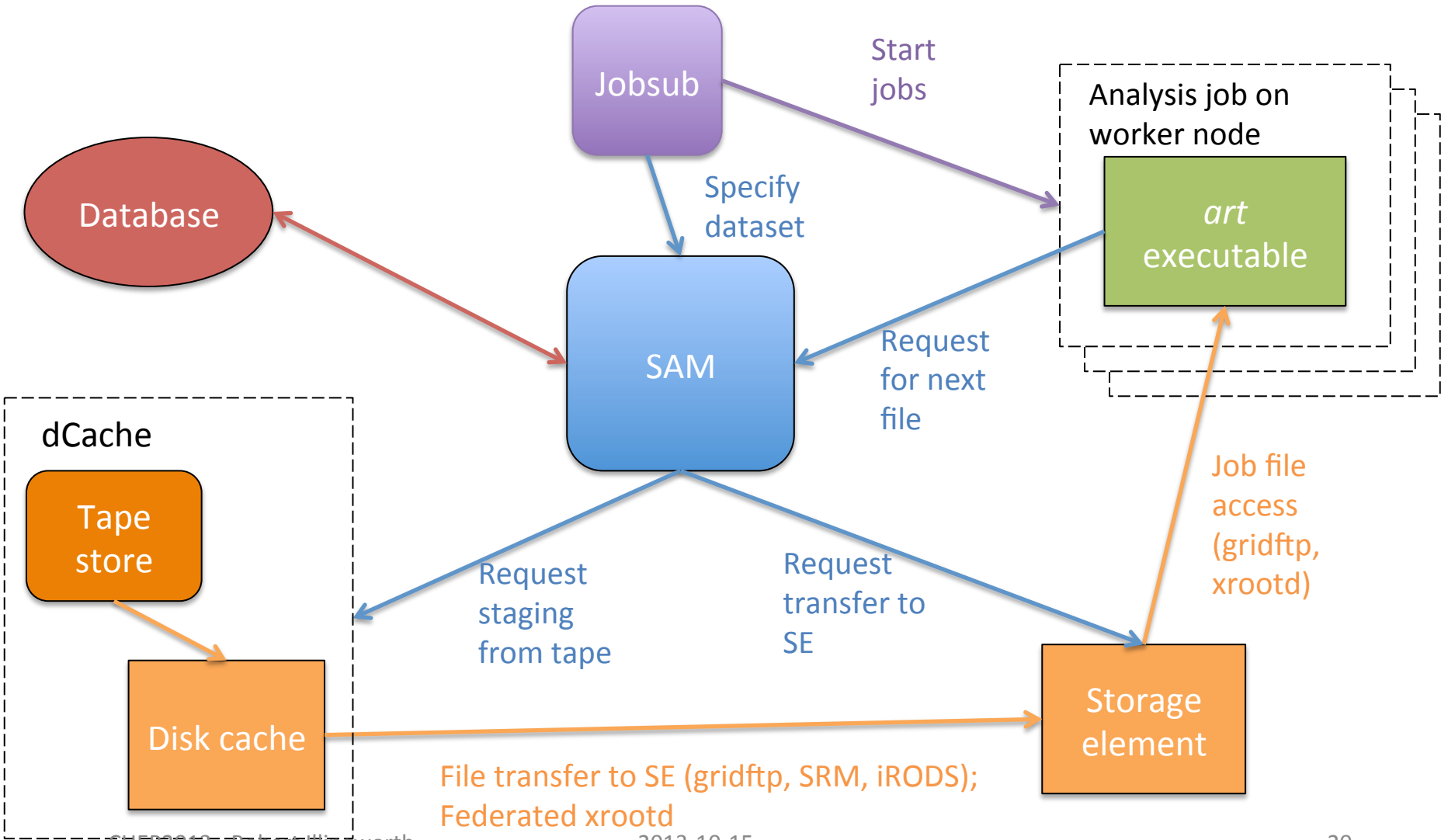
Storage integration

- SAM acts as a “high level” manager; leaves the low level details of handling storage to other systems (dCache, SRM, etc)
  - can command prestaging from tape and transfers from system to system
  - because SAM works with entire datasets it knows which files will be used before they are accessed; allows efficient automated prestaging
- Can work with multiple file access methods:
  - gridftp
  - xrootd
  - direct local access
  - …anything that can be expressed as a URL

Framework integration

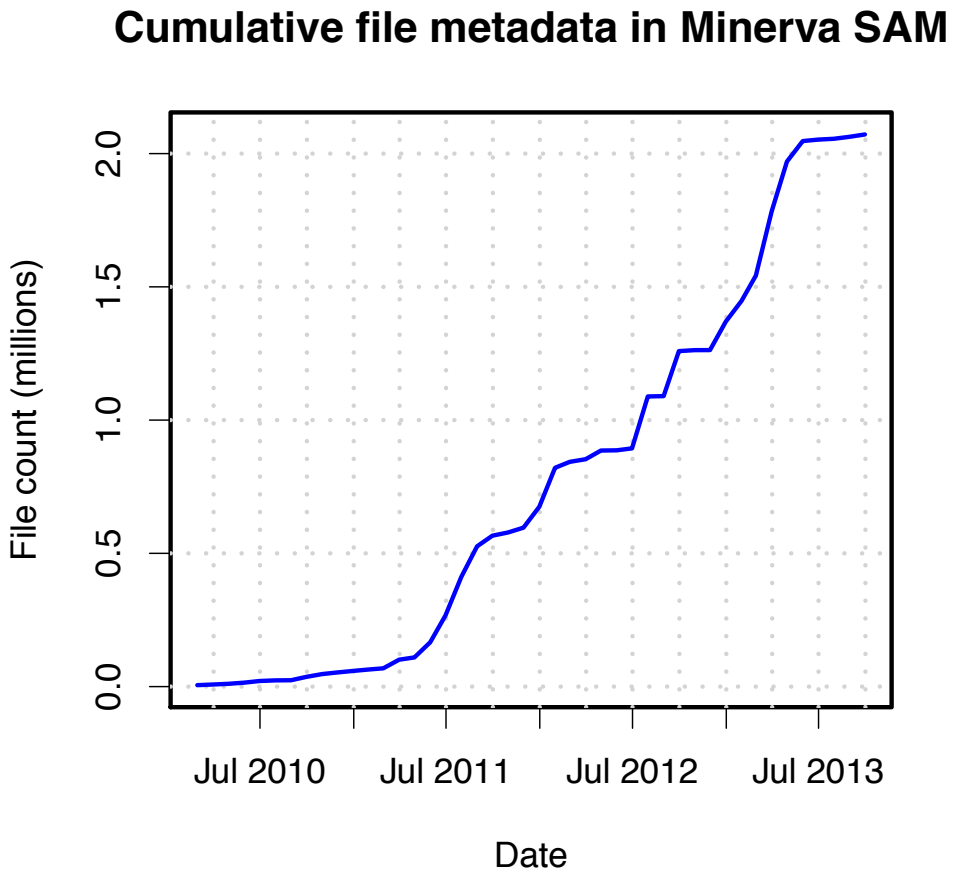
- *art* framework modules provide integration with SAM
- Input module provides next file from SAM
  - SAM provided files are just another source of events to process
- Metadata handling stores metadata within the output file
  - Makes cataloguing the file easy; just read the metadata out and add it to the database

Putting it all together



Current and future deployment

- In full use
  - Minos (~100 TB/yr)
  - Minerva (~10 TB/yr)
  - NOvA (~PB/yr)
- Deploying
  - MicroBoone (~100 TB/yr)
  - Darkside (~100 TB/yr)
  - LBNE (many PB/yr)
- Planned
  - g-2 (~1 PB total)



Conclusion

- SAM is the data handling system used by multiple current and future Fermilab experiments
- Builds on long experience from the Tevatron; updated to use modern technologies
- Experiment agnostic design provides flexibility
- High level of automation requires low level of administrative effort from experiments