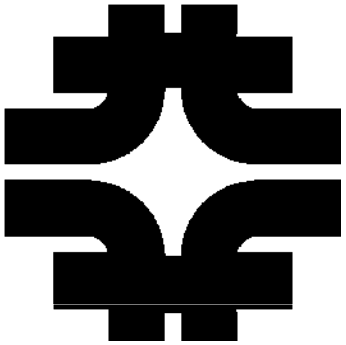
CMS ProdAgent and Production Infrastructure

An overview of the CMS experiment's production software and design.



Dave Evans
FNAL CD/CMS
On Behalf of the CMS Offline Team

Outline

- CMS Production
- Production System Overview
- ProdAgent
- ProdMgr
- ProdRequest
- Current Use
- Future Development

CMS Production Scale

- Large amount of simulated data required.
- The 2008 goal for CMS computing is to run 1.5×10^9 Simulated Events per year.
- Production to run at 30+ CMS Tier 2 Centers.
- 2007 Goals are currently 50M events per month, plus challenges.
- All produced using globally distributed computing on a variety of Grids & Farms.

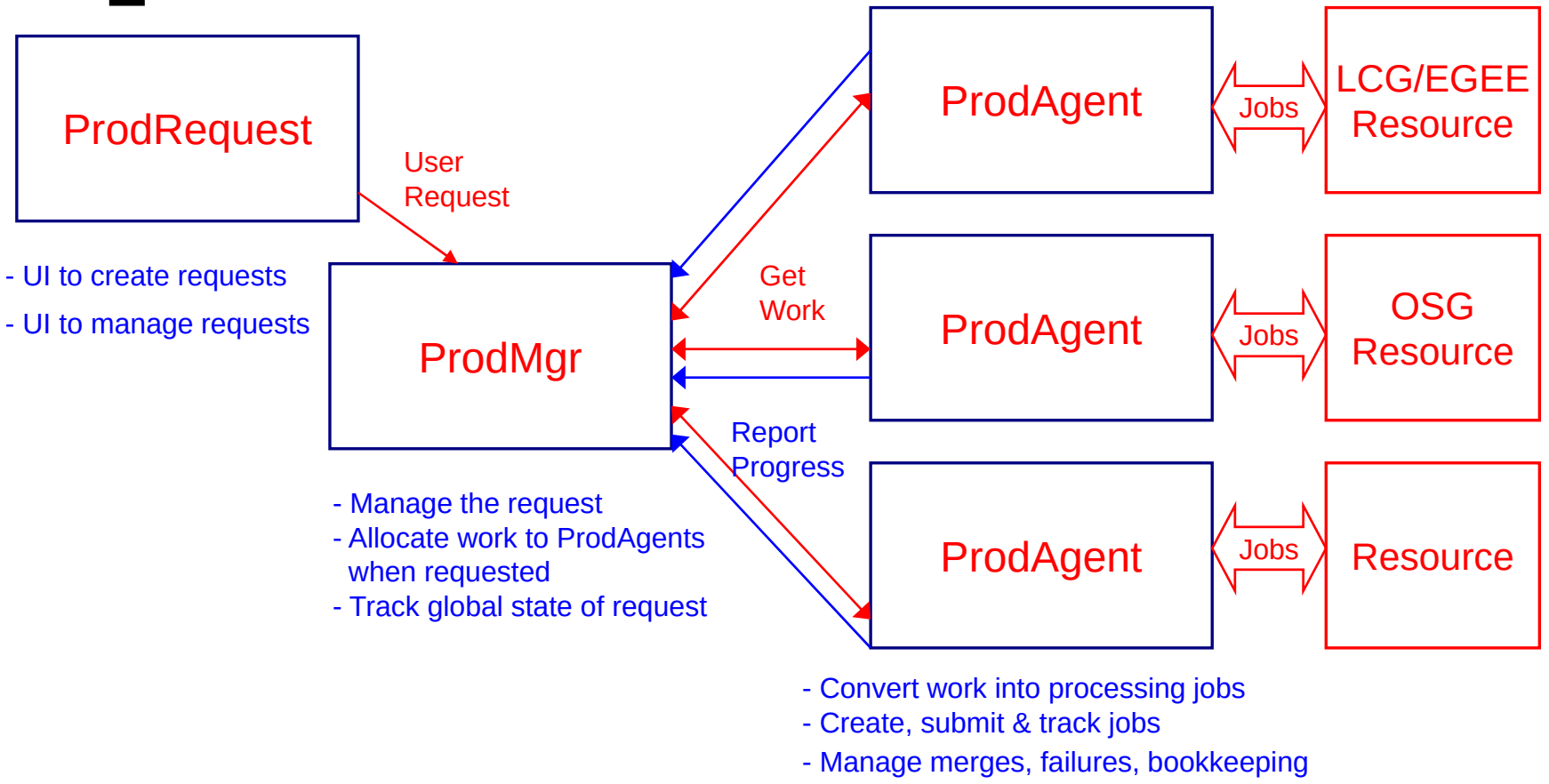
The Production System

- CMS changed it's event data model and completely redid the software implementation.
- The old production system was having serious maintenance problems
  - New data model provided a chance to completely redo the old production system.
  - Started in Fall 2005

The New Production System

- Automate as much as possible
- Low threshold for developers to contribute
- Ease of maintenance
- Scalability
- Avoid Single points of failure
- Support multiple Grid systems

Production System Data Flow



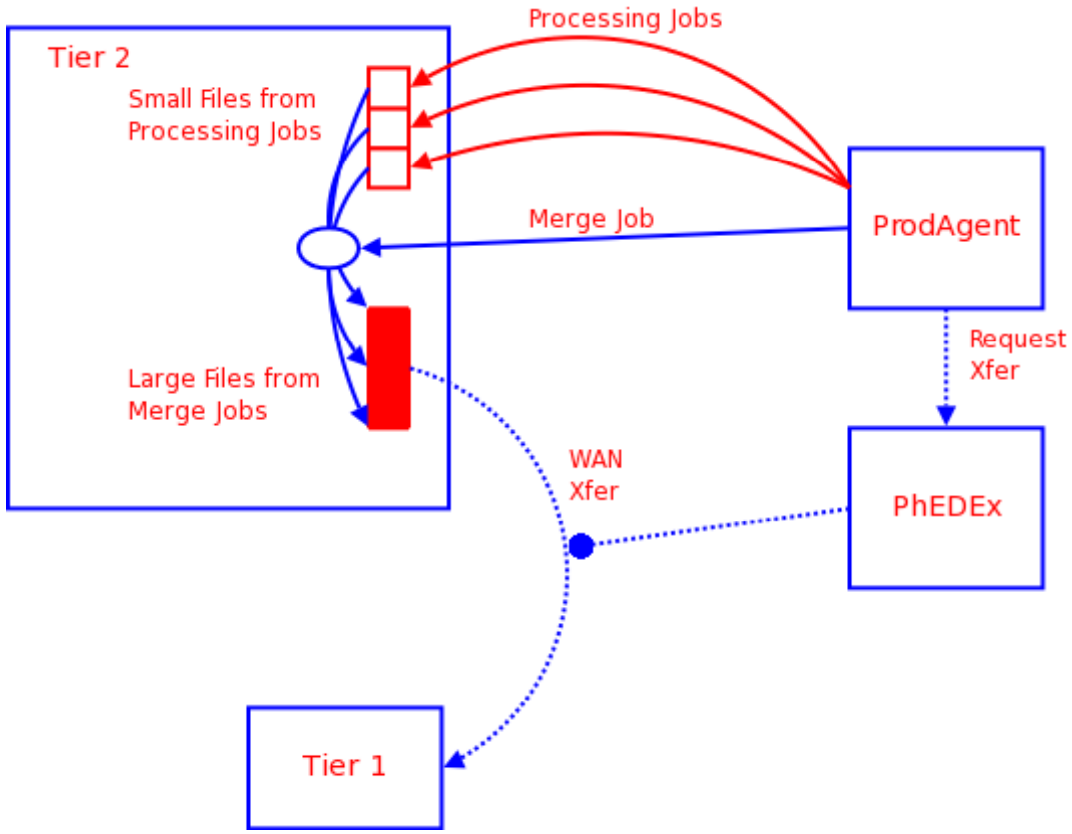
ProdAgent

- Act as a common front end to diverse resources, ranging from Grids, to Farms
- Split work into atomic components
- Simple API to communicate between components
- Can work independently
- Can cooperate on large tasks
- Scale by adding more ProdAgents

ProdAgent Workflow

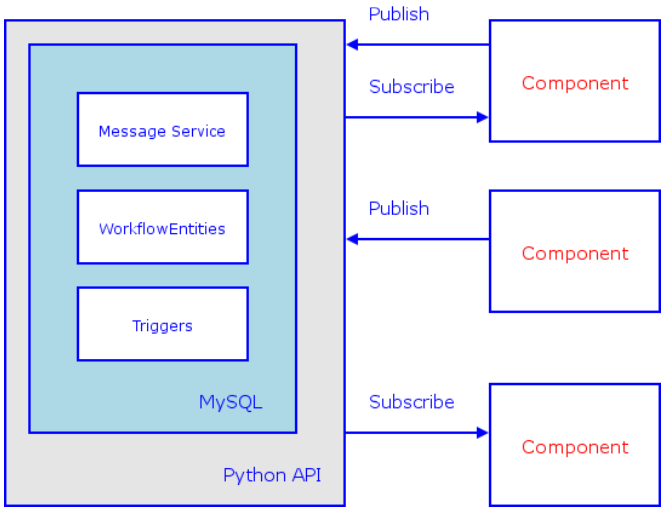
- Send Processing jobs to sites
- Drop off produced data in the sites local SE
- Report back to ProdAgent
- Merge data in situ at that site
- Catalog in Data Management catalog
- Automate retries on errors
- Delegate wide area transfers of merged data to the CMS PhEDEx system
  - Unmerged Data is cleaned up and never seen outside the agent's local scope

ProdAgent MC Workflow



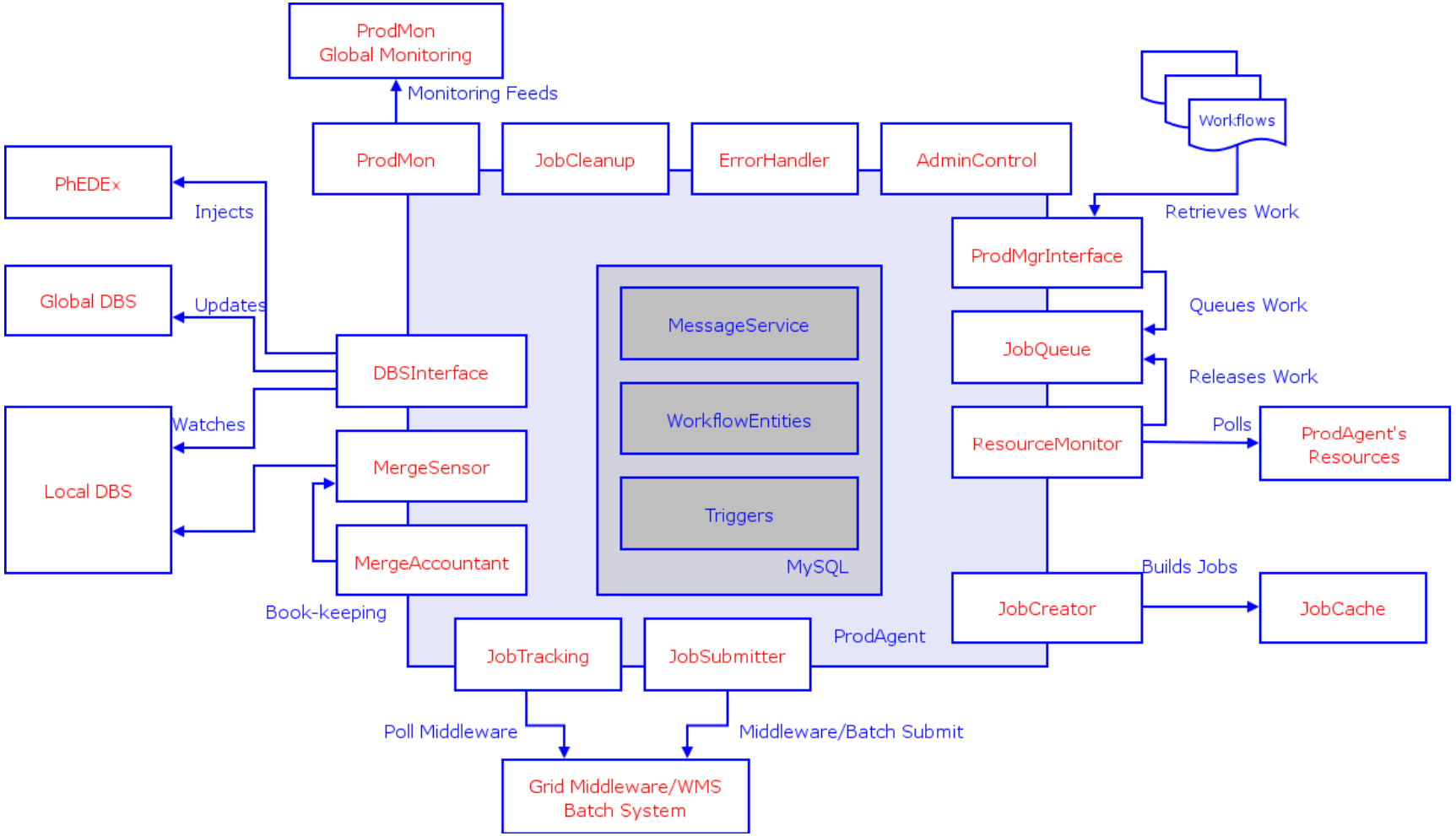
ProdAgent Architecture

- Core MySQL DB
- Python API Core Services
- Asynchronous Publish/Subscribe messaging between components
- Distinct Python Components run as Daemons
  - Encapsulate Functionality
  - Easy to swap in/out
- Local dedicated CMS Data Management Service (DBS) to allow independent processing



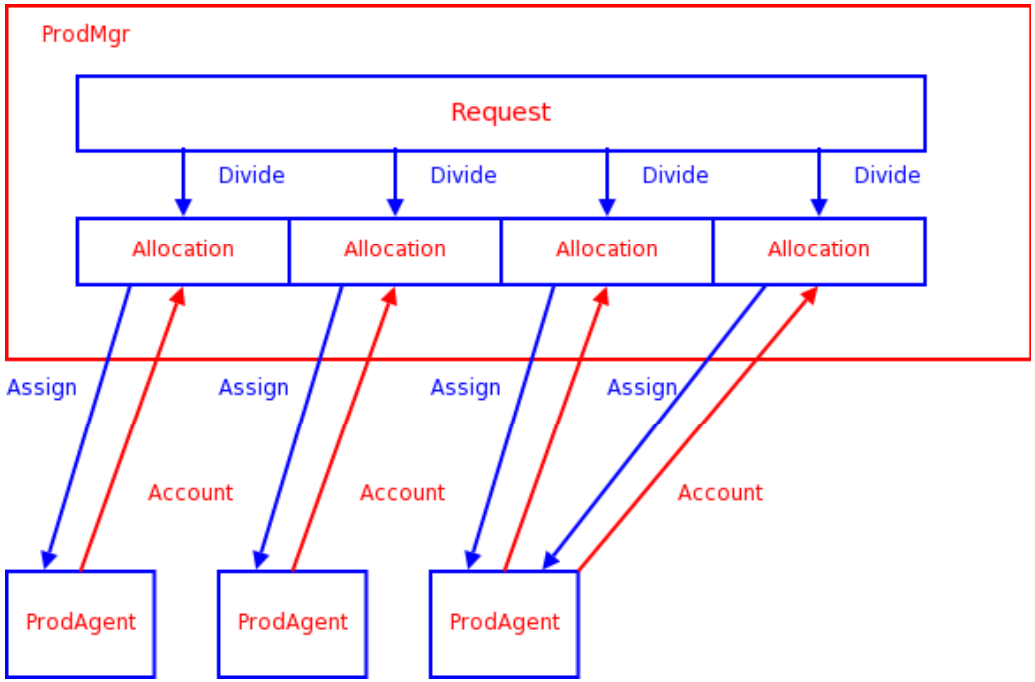
- Messages are Message + Payload
- WorkflowEntities provide state tracking for all requests & jobs
- Triggers allow synchronous actions where needed

ProdAgent Components



ProdMgr

- Allocations are large chunks of a request
- A ProdAgent cuts the allocations into jobs
- Those jobs are run, and the state of the allocation is then passed back
- Policy based logic at the ProdMgr then decides if further action needs to be taken:
  - Re-allocate
  - Expand request (new allocation)
  - Suspend request
  - Request is complete



ProdMgr Architecture

- MySQL & Python Based Accountant
- Modular Policy based management of requests
- Partitioning of requests into Allocations
- Distributes and coordinate Allocations between an array of ProdAgents
- Separation between the application layer (which talks to db) and the interface layer (web pages, command line utils)
- DB layer is abstracted from the actual backend
- Secure and authenticated access to service methods.

ProdRequest

- Global Point of entry to MC System
- Users can make/track requests
- Managers can approve/deny requests
- Production Managers can assign requests to ProdAgents
- ProdMgr retrieves workflow specs and feed work to the ProdAgents

ProdRequest Architecture

- Separation between the application layer (which talks to db) and the interface layer (web pages, command line utils)
- Modular design in interfacing with Production libraries and DBS (supports different versions)
- Role based authentication and workflows
- Uses CMS WEBTOOLS components for web interface
  - G.Eulisse "CMS Offline Web Tools" [266]

Current Operational Status

- ProdRequest Used to manage requests
- ProdAgent has been in use for just over 12 months by 4-8 operations teams
- ProdMgr Rollout happening now
- Operations Teams have successfully run millions of events (~50M per month)
  - J.Hernandez "CMS Monte Carlo production in the WLCG Computing Grid" [285]
- Automation is improving
- Global ProdMon Monitoring System rolling out
  - Coherent picture of PR/PM/PA System

Current Development

- Basic Workflow is implemented and deployed operationally
- Focus is now on automation and performance
- New features in the pipeline to improve operations, and expand the scope of the ProdAgent beyond basic production roles

Wider Adoption

- In early 2007 CMS decided to use a single system for MC and data processing
- ProdAgent is now being deployed for Tier 0 and Tier 1 Data Processing
- Prompt Repacking and Reconstruction at CERN
  - D. Hufnagel "CMS Tier0 - design, implementation and first experiences" [290]
- Re-reconstruction and skimming of datasets to at Tier 1 centres

Conclusion

- ProdAgent is in use and producing large amounts of data
- Component based design allows many developers to contribute
- Higher level tools ProdRequest and ProdMgr allow distributed ProdAgents to cooperate.
- Successful collaborative development has produced a product that is capable of much more than simple production
- Evolving into a much wider complete Workload Management System for CMS

Acknowledgements

- Alessandra Fanfani
- Carlos Kavka
- Frank Van Lingen
- Giulio Eulisse
- Peter Elmer
- All the MC and Data Ops teams

More Info

- ProdAgent TWiki
  - https://twiki.cern.ch/twiki/bin/view/CMS/ProdAgent
  - Dave Evans & Alessandra Fanfani
- ProdMgr TWiki
  - https://twiki.cern.ch/twiki/bin/view/CMS/ProdMgr
  - Frank Van Lingen
- ProdRequest TWiki
  - https://twiki.cern.ch/twiki/bin/view/CMS/ProdRequest
  - Giulio Eulisse