



Evolving Database Access Methods Towards the Grid

Alan Sill

Texas Tech University

CDF Institutional Computing
Representatives Board Meeting

Sept. 4, 2002



What Is The CDF Database?

- Consists of 7 basic applications (sets of tables with defined schema)
 - Hardware: Contains configuration parameters for data taking hardware
 - Run Configurations: What the conditions were for a given data run
 - Trigger: The decision criteria and paths by which events are chosen
 - Calibrations: Measured responses of detectors & hardware under known or reproducible conditions to map out variations and instrumental drifts
 - Slow Controls: Long-term monitoring of voltages, temperatures, etc.
 - Data File Catalog: An offline index of the files containing data taken
 - SAM: A “super-DFC” with enhanced functionality for labeling and access
- Each of the above applications has or will soon have a physicist in charge of its operation (both in tables and in code), care & feeding
 - These people are called “Application Coordinators”
- In addition, there needs to be someone in charge of matching calibrations to experimental conditions. (Rob Snihur right now.)



Why do we need it?

- Contains information critical to correct analysis of events:
 - Variation of detector response and calibration vs. conditions.
 - Changes of known settings and commands to hardware.
 - Information needed to be able to gather similar data together.
 - Database contains only <math><0.1\%</math> of information from experiment, but it is crucial to proper analysis.
- Need for access to information depends on the analysis stage:
 - Some constants and derived parameters (e.g. beam lines and alignment) are only known through extensive analysis
 - Interdependencies between tables exist and need to be kept consistent.
- Information can change through later analysis (alignments, calibrations, etc.), and a need exists to be able to apply retroactively.
- Traceability and reproducibility of analyses are required and essential.



What is the current status?

- We operate one online server for data acquisition support, and one offline server (copy of the online + file catalog info.) for all other uses.
 - Offline server subject to overloads.
 - Spikes can be caused in usage due to bugs in code and increasing number of deployed cpus.
 - By tuning things carefully, we presently are able to keep up with the load.
- Large amounts of future cpus will soon be deployed:
 - CAF plans to grow
 - Off-site institutions plan to (and in some cases already have) implement farms of tens to several hundreds of additional computers.
- Present usage patterns do not scale to fit within existing resources for serving the database contents to the world.
 - We will exceed our existing resources within the near-term future!

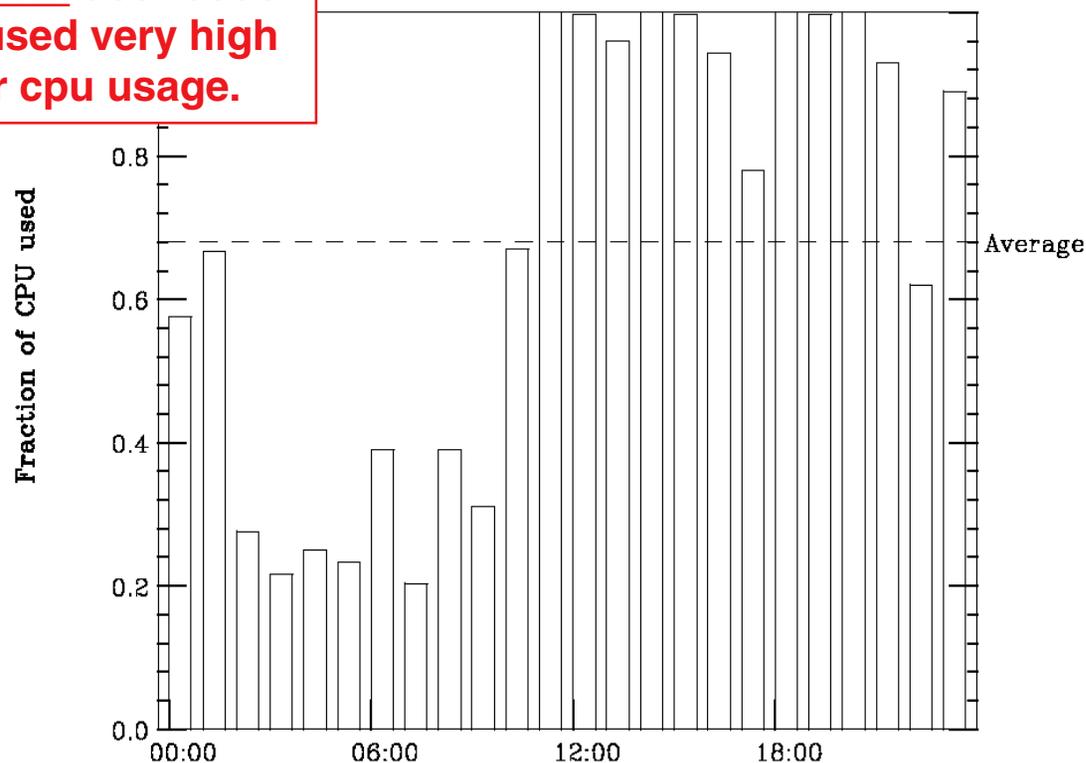
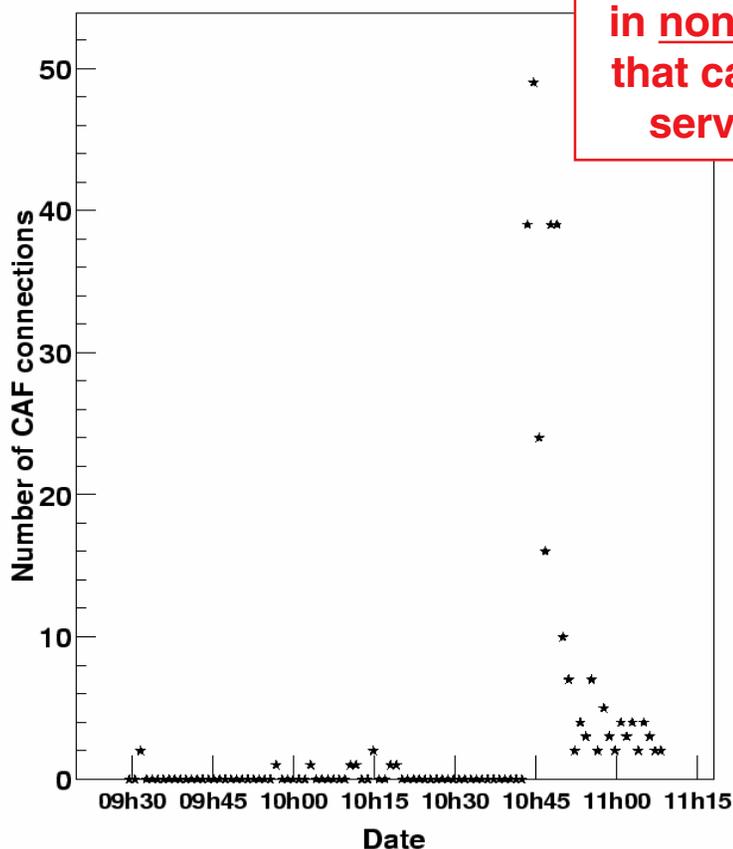


Example of DB overload. Initiated by CAF? Not exclusively...

CAF connections to database

fedfora1 Fri Jul 12 2002

In this case, this was caused by a problem in non-CAF user code that caused very high server cpu usage.

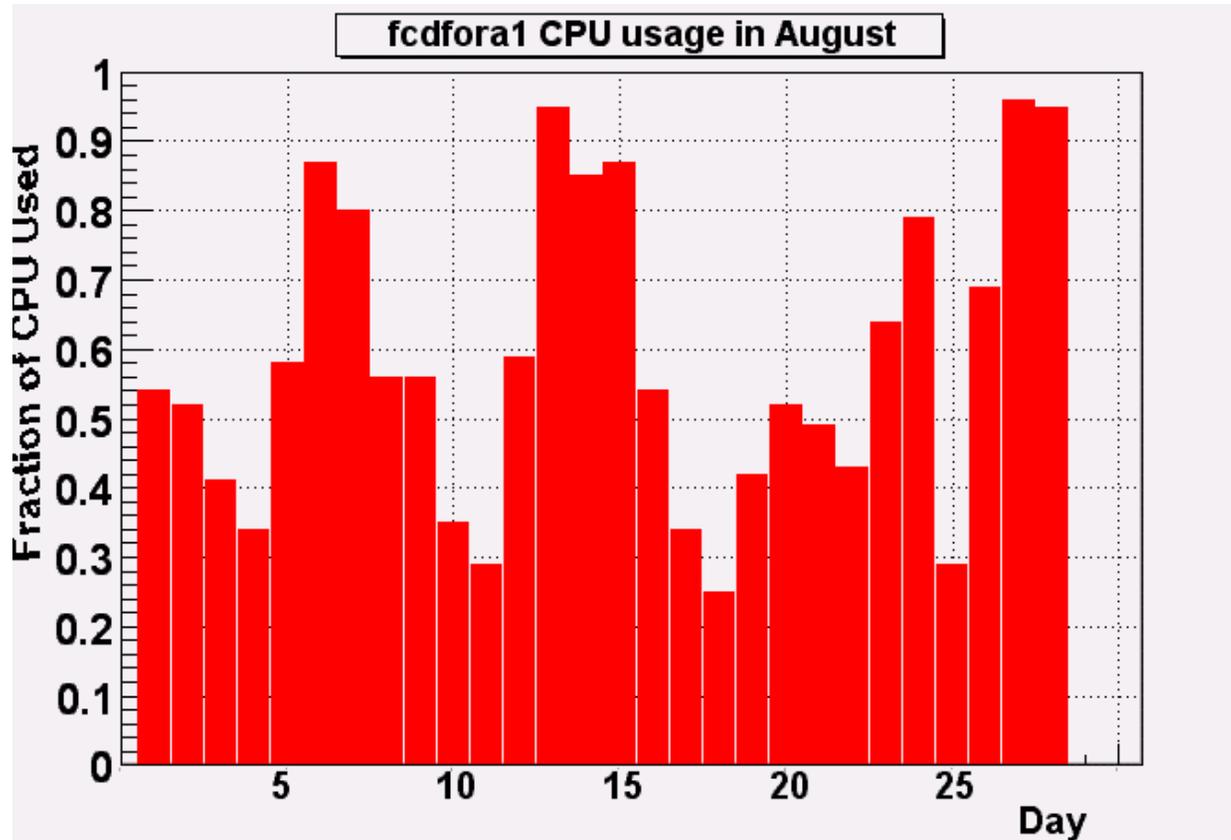


Spikes of 100% usage lasted 7 hours until user problem was found.



Continuing high cpu load on fcdfora1

CDF ICRB meeting
A. Sill
9/4/2002

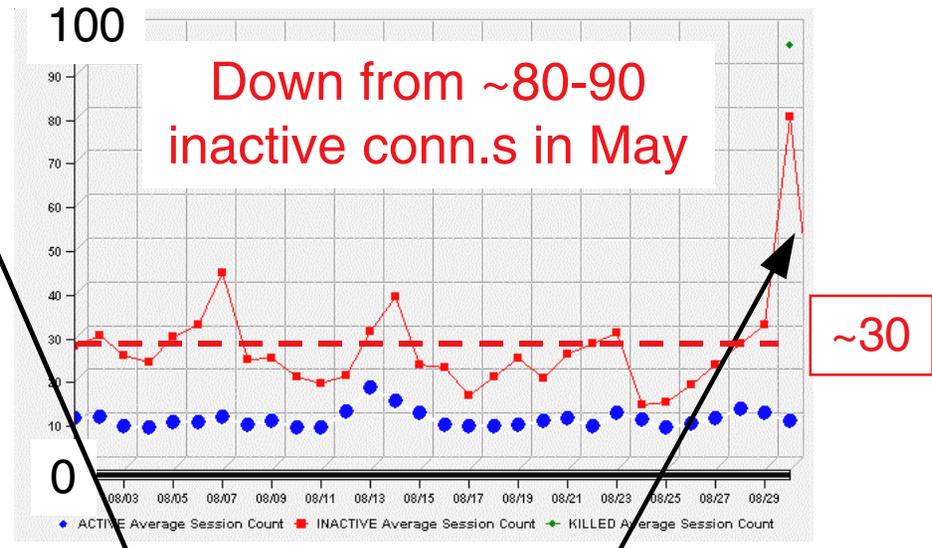
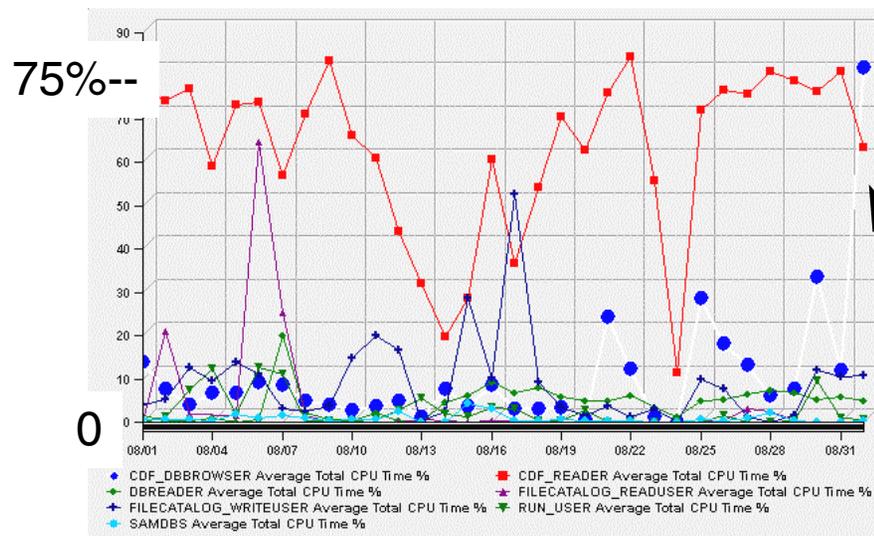


- Unacceptable loads of up to 95% CPU use over day, 100% for hours.
 - Causes long delays, connection timeouts, and interferes with farms operations.



More illustrative plots:

Server cpu and connection average totals for August:



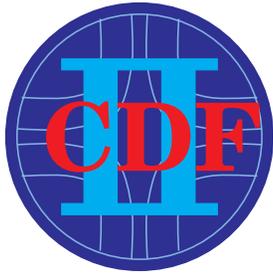
Buggy version of code

- Need more control of inactive connections!
(already better however than it was in Apr/May)
- Spikes in connections usually caused by bugs not usage



What can we do about this?

- Near term plans:
 - Learn how to survive current user usage patterns.
 - Proceed with timing, performance, replication, and other operational tests.
 - Implement at least one load-sharing offline replica.
- Longer term plans through rest of Run IIa:
 - Develop capability to field and deploy multiple replicas (Oracle 9i v2).
 - Establish optimum running conditions for each database copy.
 - Tune up performance and usage patterns.
 - Begin to prepare for grid-like deployment.
- If time allows:
 - Implement local small-scale (freeware?) copies of required calibrations.
 - Procedure to recover seamlessly from both major and minor failures.
 - Improve documentation, consistency of use, and user guidance & advice.



What can we do IMMEDIATELY to improve performance?

CDF ICRB meeting
A. Sill
9/4/2002

- 1) More CPU power is a GOOD THING ==> More and more powerful replica servers
 - Lessens clock time to serve a given query.
 - Therefore lessens your overall total license use.
 - Makes users happy.
 - (Applies to I/O power too.)

- 2) Study user connections and patterns of usage
 - Our most important immediate topic.
 - Use offline shift people resources also.
 - Could be big gains possible in optimizing tables, code and views.

... Leads to our current list of projects.



Current Projects

- Info / statistics package: (Jim K., Yuyi G., Rodolfo P.)
 - Very high priority project
 - Based on ErrorLogger; reports to separate logging server
 - User control of detail level on a per-job basis
 - Intended to be our primary tool for finding out connection usage patterns
- Calibration API & DBObjects/DBManager support: (Jim, Dennis, Yuyi)
 - Not usually listed by the CD as a distinct project, but:
 - Incredibly important to recognize that we spend a lot of our time chasing bugs and features uncovered by, or updates requested by users
 - Examples:
 - “Get by Process Name,” “Get All Instances Over Run Range” (new)
 - Connection management support, “metering patch,” etc.
 - Can occupy a large amount of programmer time to do all these tasks!

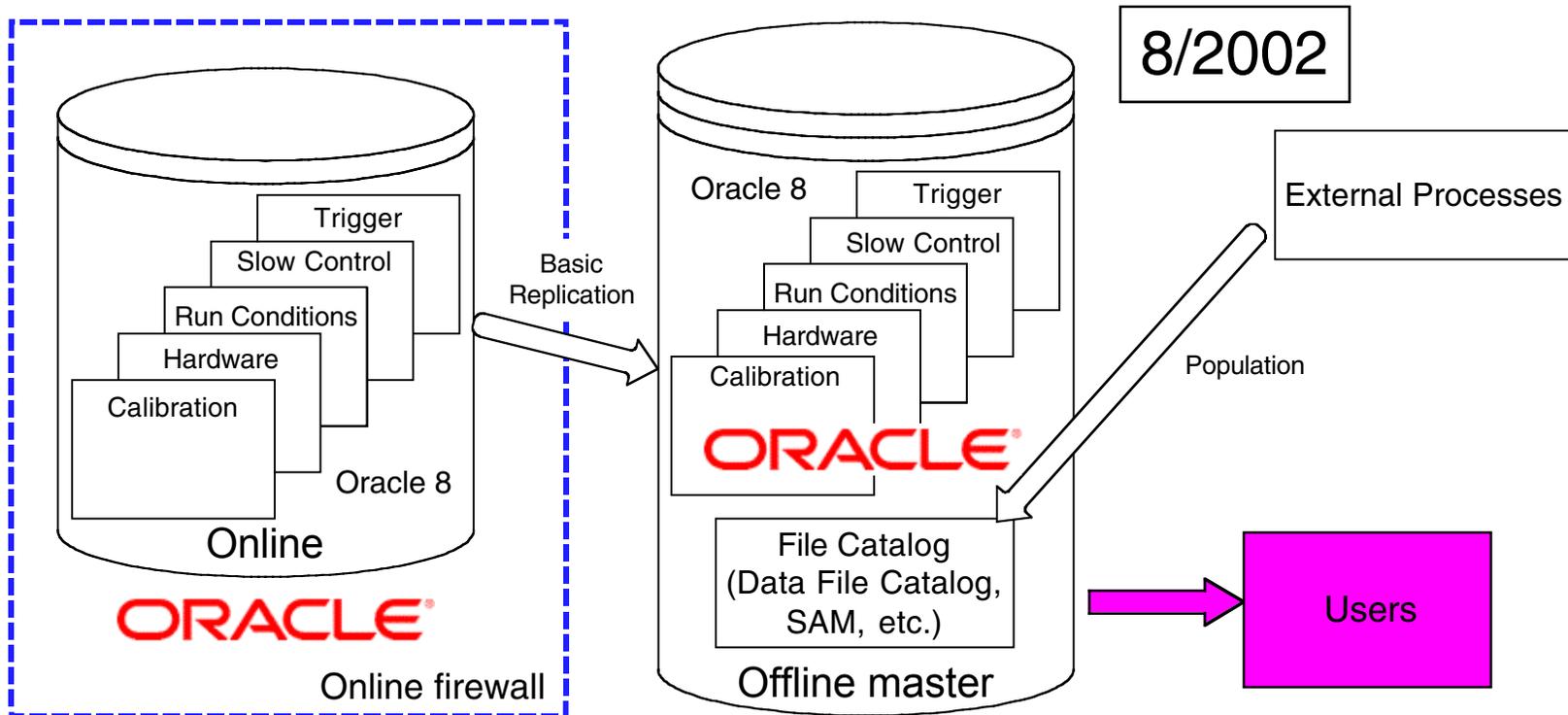


Current Projects, cont'd

- New API development: (No one defined to do this yet)
 - Need true APIs for all other DB applications (Hardware, Run Configurations, Trigger, Slow Controls)
 - Lack of API leads to lots of Oracle calls in user code
 - This is a necessary predecessor to deploying DB apps in freeware
- Freeware investigation: (Svetlana L., Richard H, David W.)
 - New joint CDF/CD project
 - Initial goals: reproduce MySQL calibration-only database done by two CDF collaborators, test, deploy, & support.
 - Move on from this to investigate alternatives (PostgreSQL, for example) and study classes of support needed for various user job types.
 - Road map exists, but people are in short supply.
 - Could be very important project in the future.



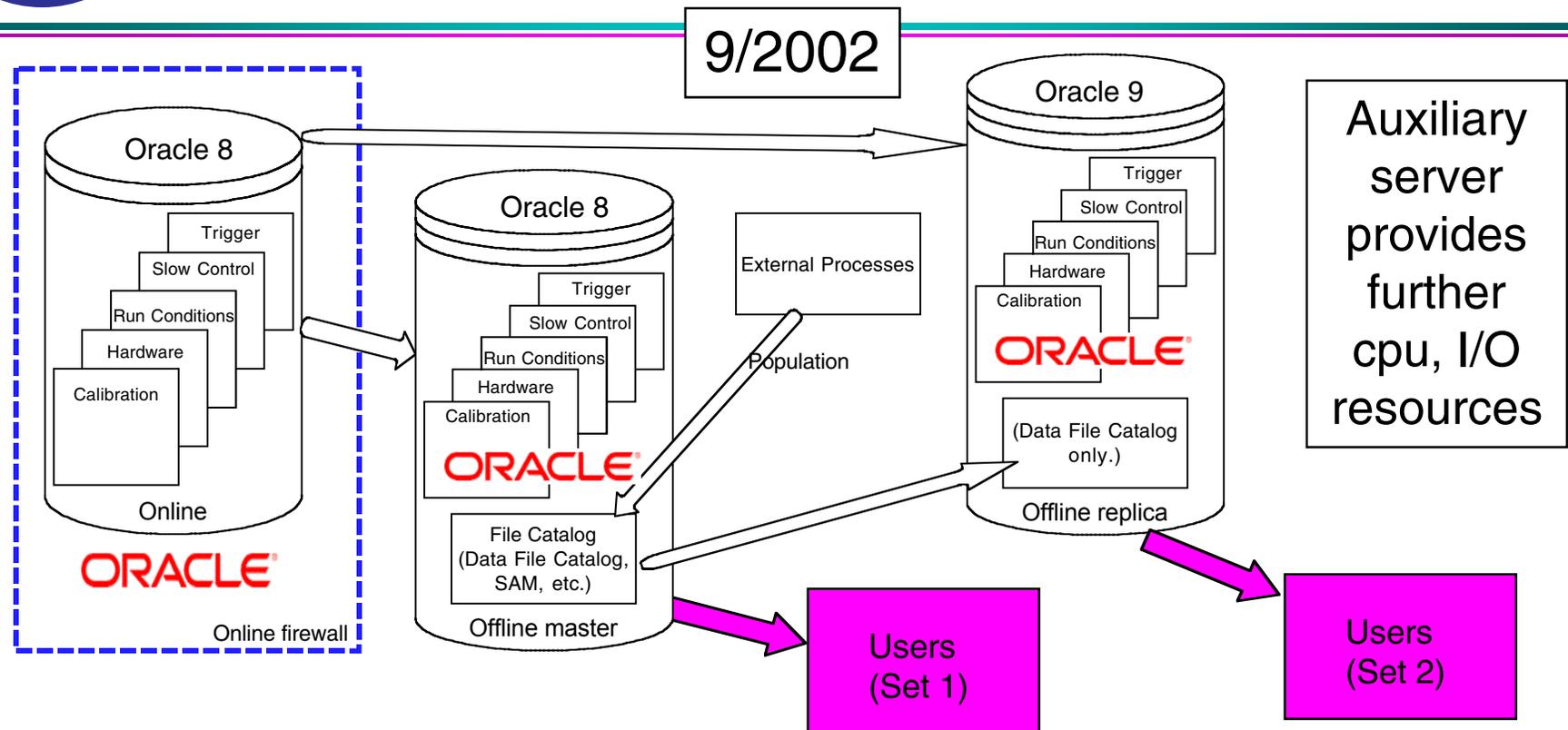
Present CDFDB distribution scheme



- Most (5) database applications originate online
- Data File Catalog, SAM added to offline server
- Replication only occurs online --> offline (to get out of firewall)



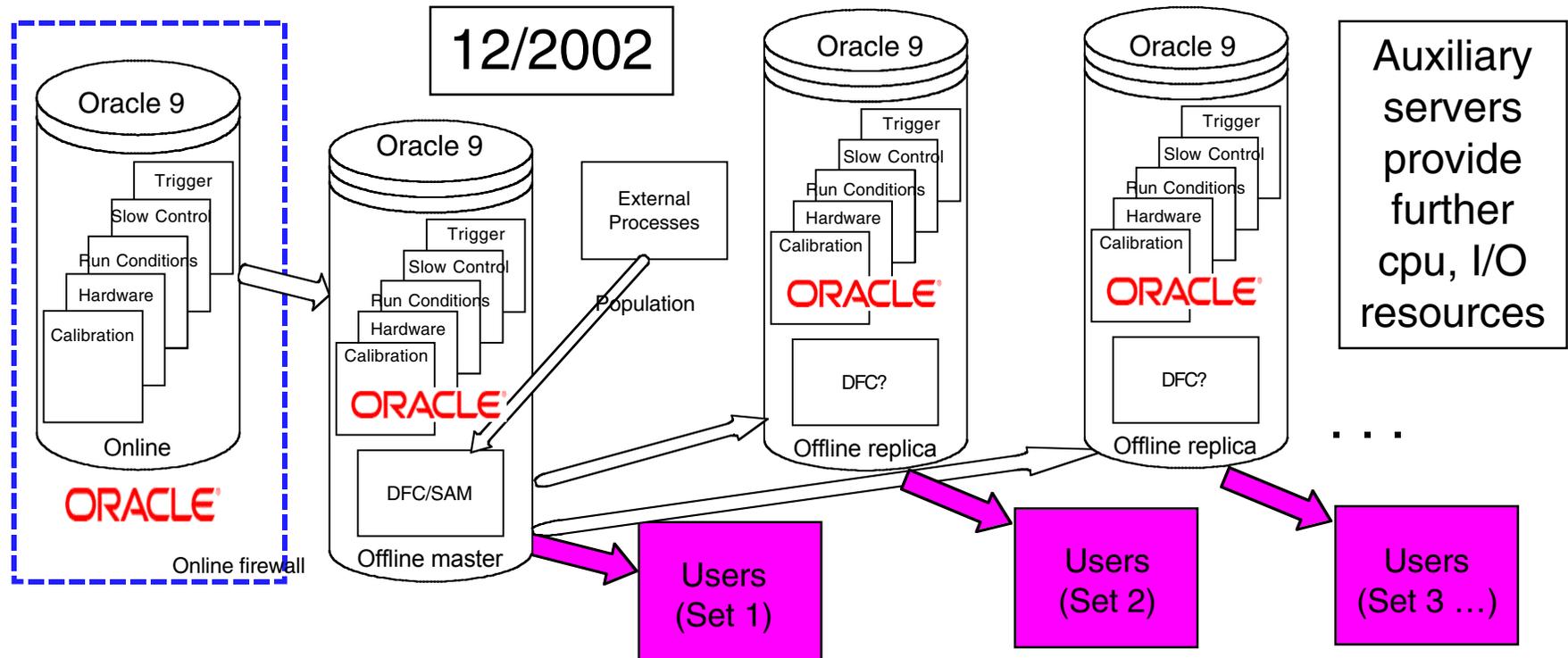
Near term distribution scheme



- 5 database applications still originate online, replicated to offline
- Data File Catalog, SAM reside on primary offline server
- DFC only replicated from primary to replica offline server



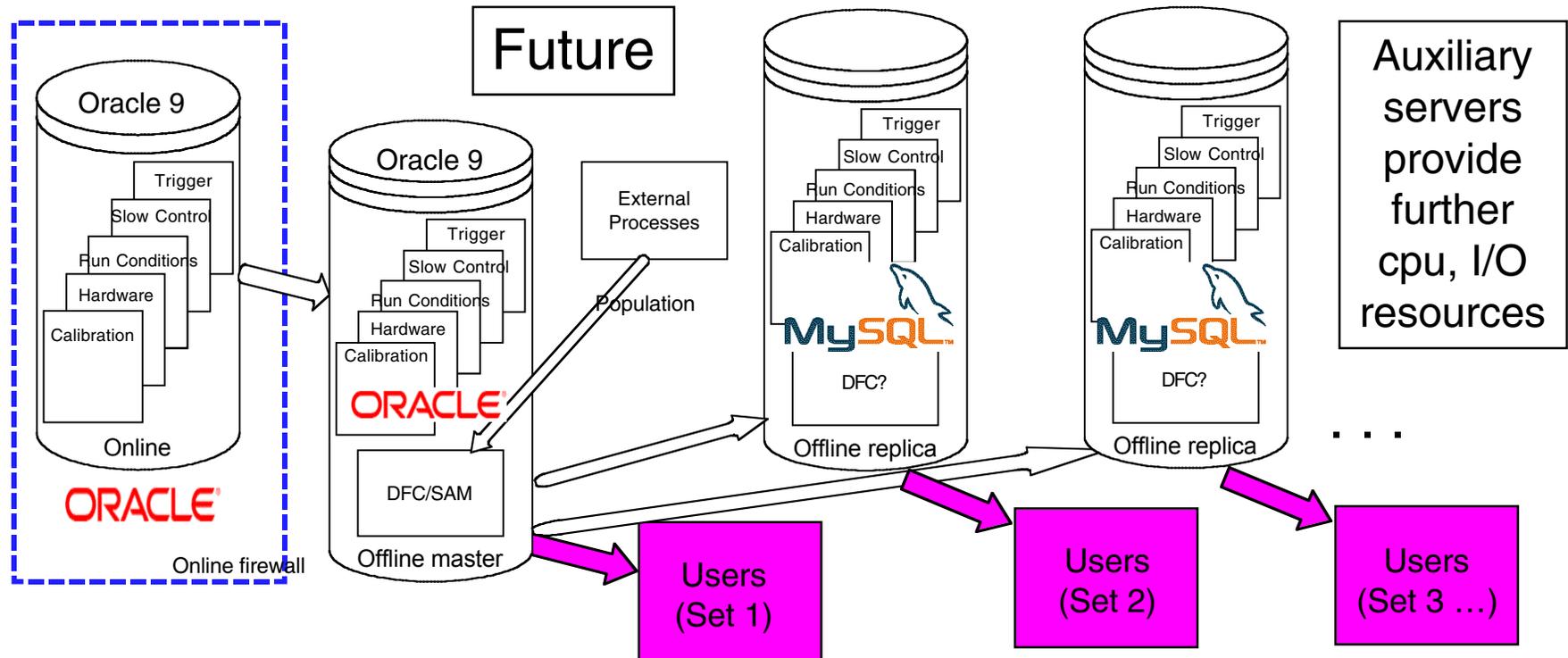
Long term Oracle distribution



- Oracle 9iv2 “datastreams” replication allows many choices.
- SAM takes over DFC? One copy, or many? (One in present scheme)
- Connection broker to mitigate and assign servers to user jobs?



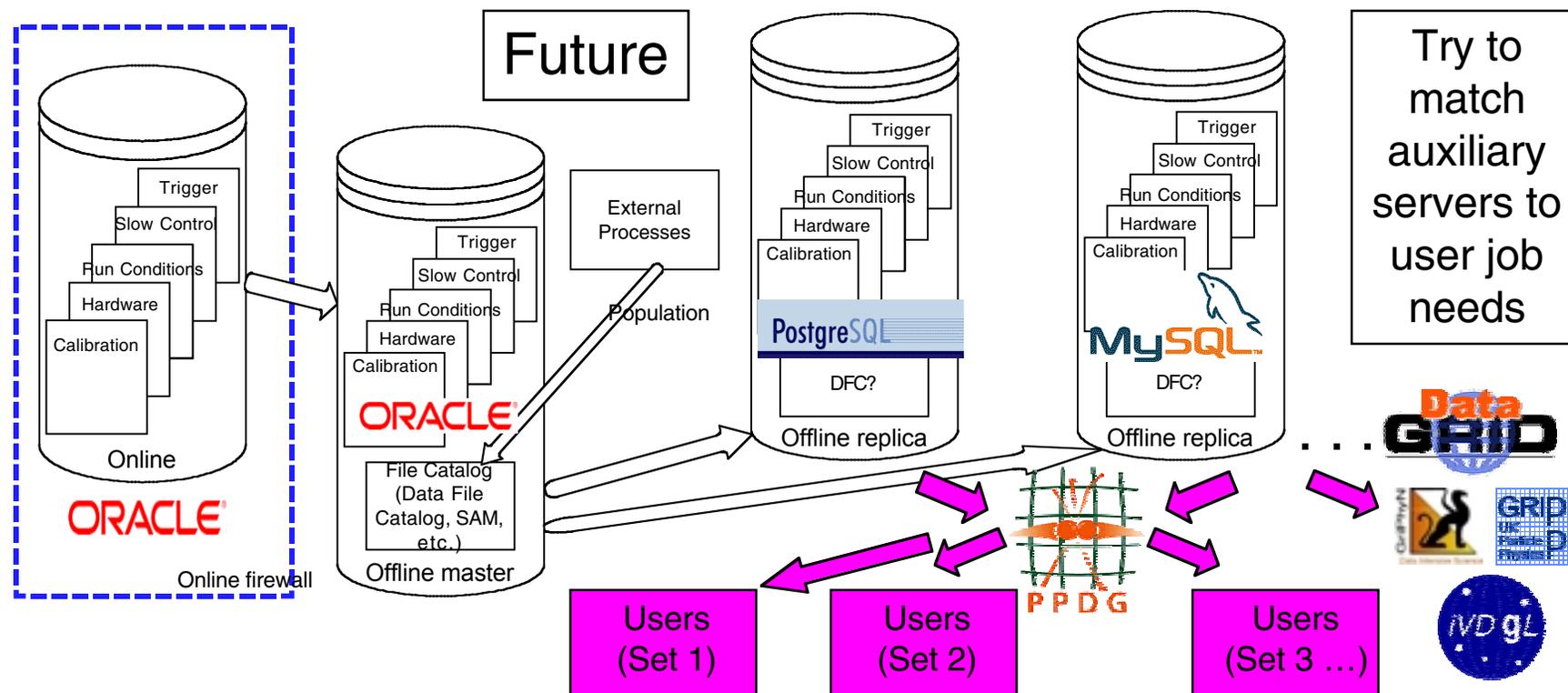
Alternative: Freeware?



- Oracle 9iv2 “datastreams” replication allows calls to external scripts!
- MySQL, PostgreSQL, etc. are possibilities. Export can be triggered.
- Connection broker still an issue. Migration to grid tools? Content?

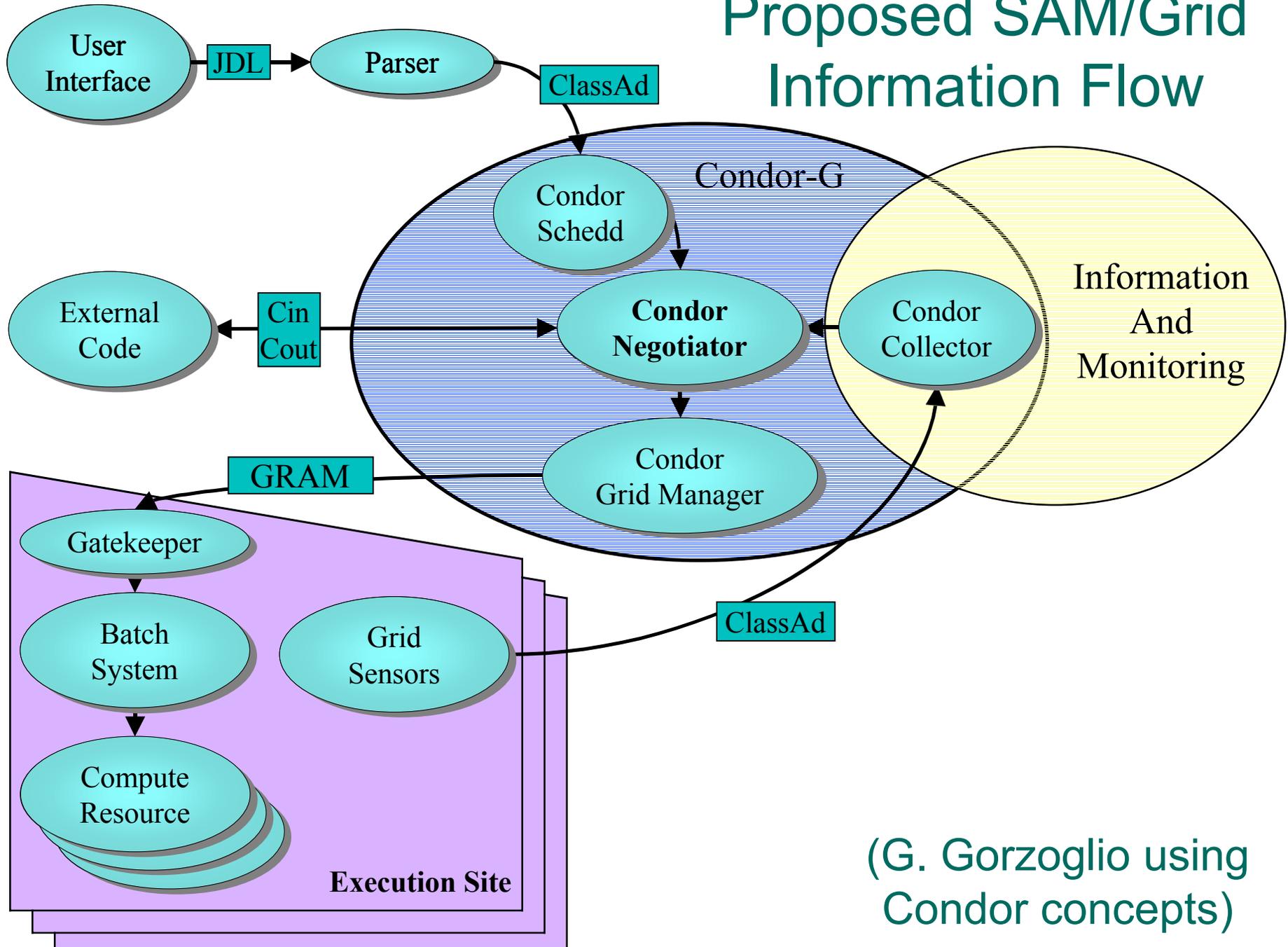


Alternative II: Freeware + Grid



- Grid provides many opportunities for growth, both mixed and homogenous.
- Again, since export can be *triggered*, can keep in sync from master copy.
- Ideally, match local copy to user jobs... Robust grid tools definitely needed.

Proposed SAM/Grid Information Flow



(G. Gorzoglio using Condor concepts)



Time to Act is Now

- None of this is going to happen without user involvement.
 - Need to act on every level (debugging, user job profiling, API and code design & maintenance, server design & configuration, etc.)
 - Our job as organizers is to break things down into manageable separate jobs that relate to long range goals,
 - Then match people to jobs until they are done.
- Some of this is admittedly hard to define:
 - We don't know how the Grid is going to evolve exactly.
 - We do know that the code will become more mature and sophisticated, that people are going to want to analyze data and do simulations, are already beginning to do so on site and off-site with large numbers of cpus, and that we are already beginning to hit intrinsic limitations of our existing servers in a serious way.
- Way forward is to get started and have a clear road map!



A small sampling of tasks that an individual user or group can do

- Database monitoring - connections, tables, durations, etc.
 - Analyze usage patterns to help set designs and policies.
 - Develop validation for DB access in new versions of CDF code
 - Profile programs of various sorts using new tools (simulation, etc.)
- Freeware port of database and replication
 - Participate and help with comparison of freeware choices
 - Scripts to help with population and updating of freeware db
 - Validation of contents to check that they are the same as main db
- API design, coding and testing
 - Extend calibration-API-like features & design to other databases
 - Use “physicist insight” to determine what we need for analysis
- SAM/Database/Grid test stand
 - Small-scale array to test grid concepts for distributing & connecting to database in various ways



A small sampling of tasks that an individual user or group can do (II)

- Connection broker / negotiator design & testing
 - We may not WANT a 3rd tier! (May not work for off-site - latency problems...)
 - However some kind of broker or negotiator is likely needed.
 - Help design this, test, and make sure that it works off-site as well as on.
- Study of slow controls and monitoring system
 - Tables need to be classified and redesigned with eye toward analysis
 - Decide what to save & store for archival vs. replicate (see below)
 - Accessors and methods to use data in analysis programs (B field, etc.)
- Replication procedures and design
 - Do we want or need “pull on demand?” (Secondary sourcing)
 - Prefetching? Fetching in batches at begin job rather than by run?
 - Should all data be sent to all servers, or only a subset?
- Profile of online usage - optimize length, type of connections



CDF Run II Database Plans

Let's avoid this problem...



... and instead build the tools that are really needed!