CMS Data Management Updates

Nicolo' Magini,
substituting for Brian Bockelman

Historical Space Management in CMS

- Space management in CMS has historically been a site operation:
  - (Run II) Site informs CMS the number of TB of central data they can host.
    - CMS is responsible to keep PhEDEx-based data below this threshold.
    - Site admin is responsible for keeping non-central (e.g., user) areas under control.
  - (Run I, obsolete) Site approves all transfers and disapproves when site is full.
- **Space management is entirely delegated**: concepts such as space tokens, quotas, etc, are only used by the site - and only if that is the admin's preferred toolset!

Space Management in CMS

- This model assumes the admin actively engages with CMS! We believe there is *interest* in offloading some responsibility back to CMS:
    - **If** we utilize CRIC in the future (high-level storage data), we could automatically reduce centrally-managed data volume based on storage availability.
    - The existing SpaceMon project (using storage catalog dump) aims to help admins discover dark data at their site.
- Based on CMS computing norms, participation at sites will be OPTIONAL.
    - Of course, if sites don’t participate, we are more likely to overrun their storage!

Data Access in CMS

- CMS currently supports: rfio, dcap, xrootd, POSIX, HTTP, and arbitrary callout scripts natively.
    - Additionally, “whatever else ROOT supports”.
    - CMS native support adds better statistics gathering, improved multithreading, improved caching over ROOT’s TFile.
- Outside POSIX, the most refined / tested remote IO API is Xrootd.
    - Starting in 2017, HTTP/DAVIX should be an option for the most recent releases. Current releases simply fork/exec curl.
    - Intent: nudge sites to start deprecating rfio and dcap.
    - Going forward, best remote IO implementation remains Xrootd. HTTP (DAVIX implementation) likely has similar performance but less reliability over the data federation.

Data Access in CMS

- CMS currently relies on remote IO for DIGI-RECO workflows.
  - It allows us to operate in a mode that reduces time-per-event by >2x.
  - We simply don't have sufficient CPU to run without it.
- This is different from RunI, where use of AAA was a slight optimization to the system.

Data Transfers in CMS

- Data Transfers (copy file from Site A to Site B) are driven by two applications: PhEDEx and ASO.
  - Both heavily utilize a FTS3 backend; ASO exclusively utilizes FTS3. PhEDEx has other backends that are not commonly used.
  - As long as both ends of the transfer have one FTS-supported transfer protocol in common, CMS can perform data transfers.
    - Preference is currently GridFTP due to widespread availability inside the WLCG and out.
      - As sites must interoperate with each other, this makes GridFTP a de-facto requirement. If we cannot move data, we must use remote streaming techniques to the site - a decrease in efficiency.
      - There are some special cases where another protocol can be used. Still seems to be the exception. Do not currently foresee a future where we migrate from GridFTP.
- **Editorializing from Brian**: Based on their technical architecture, I have serious doubts about the viability of the WebDAV or Xrootd-based third-party transfers. They look a lot like the (unsuccessful) srmCopy mode instead of the (successful) GridFTP appraoch.

SRM in CMS

- SRM is not necessary for CMS disk-only endpoints:
  - Two sites (Nebraska and Bristol) have already retired their endpoints.
    - CERN is finishing up this transition.
    - Several other US sites will likely retire their endpoints this calendar year.
- SRM *theoretically* could be retired for tape archives: no archive sites seem to be pushing for this.