HSF Event Delivery Working Group Kickoff Meeting - Live Meeting Minutes

Attendees:

- Alexei, Ben G, Brian B, Marc W., Mario L., Nurcan O., Tadashi M., Torre W., Wei Y., Wen G., Marcin, Gancho
- Excused: Alessandro DG.

Minutes:

- Opening introduction from Brian (https://indico.cern.ch/event/836096/contributions/3511324/attachments/1890837/3118236/HSF-EventDeliveryWG-Introduction.pdf)
    - Big picture questions: What do we mean by “event delivery”? What does a successful event delivery system look like? How does it differ from a WFMS like PanDA?
- IDDS presentation from Tadashi (https://indico.cern.ch/event/836096/contributions/3511435/attachments/1890734/3118045/go).
    - What do we aim to do with this? Three items that we orchestrate currently:
        - Sub data block replication.
        - Dynamic data placement and tape carousel
        - Fine grained event processing -- for when processing granularities are smaller than storage granularities.
    - Aim to generalize all three workflows into a single system. Thinking of “input collection”, “output collection”, and transformation.
    - Potential applications:
        - On-demand production of analysis format data. Only trigger creation of data workflow when a user task is present.
            - Simulation: replay the user analysis usage over the past few years, assume DAOD is produced on demand, and completely delete data after some idle threshold. In this case, there is an 80% reduction in disk usage for DAOD and 3x increase of CPU usage.
            - Discussion: Not clear this is a huge win? CPU usage is non-trivial. CPU usage would increase from 1.7M CPU year to 4.9M CPU year with 30 days of DAOD lifetime. CPU allocation is optimized based on physics needs and currently 9M CPU is being used in ATLAS for delivation production. 4.9M would be still acceptable.
        - Remote data transformation / reduction, fine-grained tape carousel.
    - Why a new service?

        - Close orchestration of WFMS & DDMS is believed to be essential for optimizing use of computing & storage resources.
        - Current systems that do this have unclear service boundaries and lack reusability.
    - IDDS aims to have clear service boundaries.
    - Data model.
        - Collections: logical set of events?
        - Storage ID: Storage endpoint?
        - Many to many relationship between requests and transformations; allows recycling of output collections.
    - Interaction with External Entities
        - Inbound - standard REST with apache/nginx. Non-X509 auth.
        - Outbound - push notifications with HTTP hooks or message broker.
    - Current status:
        - Finalized the data model. Creating database in Oracle 19 @ CERN.
        - Hoping to have two distinct versions during 2020.
    - Questions:
        - BB: The IDDS work eventually pushes the production architecture to work with “objects”, not files. How does this reflect on the architectuere? TM: we have been thinking of representing these as independent, transient files. BB: Isn’t XRootd working on a non-file-based storage interface? WY: Are you thinking of SSI? There are some key differences between the current implementation of SSI for LSST and what may be useful here. BB: **ACTION ITEM** - have Andy present at a future meeting.
- ServiceX (Marc W.):
    - ServiceX works to deliver data to clients. Specifically, aims to work with columnar analysis and end-stage analysis systems.
        - Interact with common data management systems.
        - Transform to different data formats (esp. non-ROOT).
    - In general, the aims to be containerized and modular.
        - Try to utilize industry-standard tools. Reproducible, portable deployments.
    - Lots of the effort is from IRIS-HEP R&D in DOMA & AS. Focus is on ATLAS and CMS data types.
    - Why is this better from Classic Analysis Workflow (xAOD->DAOD->flat ntuples->skimmed ntuples->histogram/plots)?
        - Enormous variation in how the last three steps are done. Significant code burden from each group to simply “churn the crank”.
        - Mistakes in one step often means “throw away current results and start over”.
    - Columnar data:
        - Group objects into contiguous columns rather than events.
        - Enables vector processing.

        - Adding event content to the analysis is adding a new column; can keep existing data (no need to throw away).
        - Significant change for analyzer: no for-loops but rather column transforms.
    - Key insight for HEP: concept of “jagged arrays” - store data in a flat array but represent as non-uniform N-dimensional arrays.
    - Goals:
        - Tailored for near-interactive, array-based analyses.
        - On-the-fly data transformation into variety of formats (ROOT, HDF5, Arrow, Parquet)
    - Architecture overview:
        - User specifies metadata tags targeting the data.
        - User receives a token for the request.
        - Internally, ServiceX downloads the data locally to XCache, transforms as necessary, and does bookkeeping.
    - Implementations:
        - API server
        - DID finder
        - Transformer
        - Kafka manager.
    - Early prototype up and running. Rudimentary webpages (see slides for screenshots).
    - Envision both pre-transformation and post-transformation caching. Unclear the final configuration of both.
    - Implemented backpressure for large requests - don’t allow transformer to outrun the available buffer space post-transformation.
    - Current focus:
        - Get full, at-scale demo up and running. Focus on performance under multiple concurrent requests.
        - Stability testing (randomly killing containers)
        -