Event Service Plans and Expectations for 2015

Torre Wenaus, BNL
ADC Facilities Jamboree
Dec 3, 2015

Outline

- Vakho has described where we are with the event service
- Will discuss plans and possibilities going forward
- Some are at the level of 'deploy in 2015'
- Others are at the level of 'develop in 2015' to provide longer term capability
- This will lead into (the longer view part of) the next topic, beyond-pledge scenarios

Moving ES to production

- SC14 demo was a tremendous driver for the effort, and not a diversion – has brought us to the point of having a system we can work on deploying for real ATLAS use
- With the SC14 demo behind us, the effort is now moving towards production readiness
  - At NERSC and Amazon spot market
  - Activate real output merging (in progress)
  - Address scaling issues: Amazon seems OK, some issues at NERSC
  - Begin simulation validation of ES
- Where else can/should we target
  - Conventional grid resources is the easy one, do we use it?
  - Titan? Stampede?
  - Any interest from Euro HPCs?
  - HLT Sim@P1?
  - Euro clouds?
  - ATLAS@Home?
- All dependent on volunteer effort from people connected to the platform

Kudos to our core ES devs who delivered an excellent demo

Wen Guan
Tadashi Maeno
Paul Nilsson
Danila Oleynik
Vakho Tsulaia

ES for cloud, grid production?

- Ability to quickly free a WN from production, if you're draining or if you're switching that WN to another role
  - Single/multi core
  - Production/analysis: quick reassignment of a WN from production to analysis without waiting for a long job to finish. Changeover becomes much more agile
- Sim@P1: quicker draining / less losses when closing down the cloud

ES for ATLAS@Home?

- ES would seem to hold advantages for ATLAS@Home
  - Fine grained workloads extract useful work despite early/frequent interruptions, terminations
    - Deliver useful results back every ~15min (tunable)
  - Take advantage of PanDA/JEDI's ability to do the fine grained bookkeeping and retry
  - Configuration of jobs no longer an issue; workers are fed events to process to advance a task
    - Worker lifetime can be tuned as desired (independently of output frequency), without requiring job length tuning
    - Maximize the CPU to I/O ratio without the risk of making jobs too long to have a high completion rate
      - All but the last 15min is complete and safe

ES for Analysis?

- Opening up opportunistic resources to analysis allows to directly serve urgent spikes, pre-conference crunch, 'human waiting' cases
  - More direct and rapid benefit than using ES for production and applying liberated conventional resources to analysis
  - Some opportunistic resources may be designated particularly for analysis
  - If ES becomes heavily used for production, uniformity of approach could be beneficial
- Better/earlier feedback to users
  - Evaluate task integrity and running parameters with a 15min event cluster rather than subjecting users to the latency of scout jobs (for shorter tasks the scout latency can be felt)
  - Logs are available incrementally and semi-continuously
    - Can play well with more automated error diagnostics, real time analytics
    - DAST comment from last week: "I am quite puzzled that a problem that made the job crash on the very first event, still took two full days from the submission before showing up in any log."

ES for Analysis – Outputs

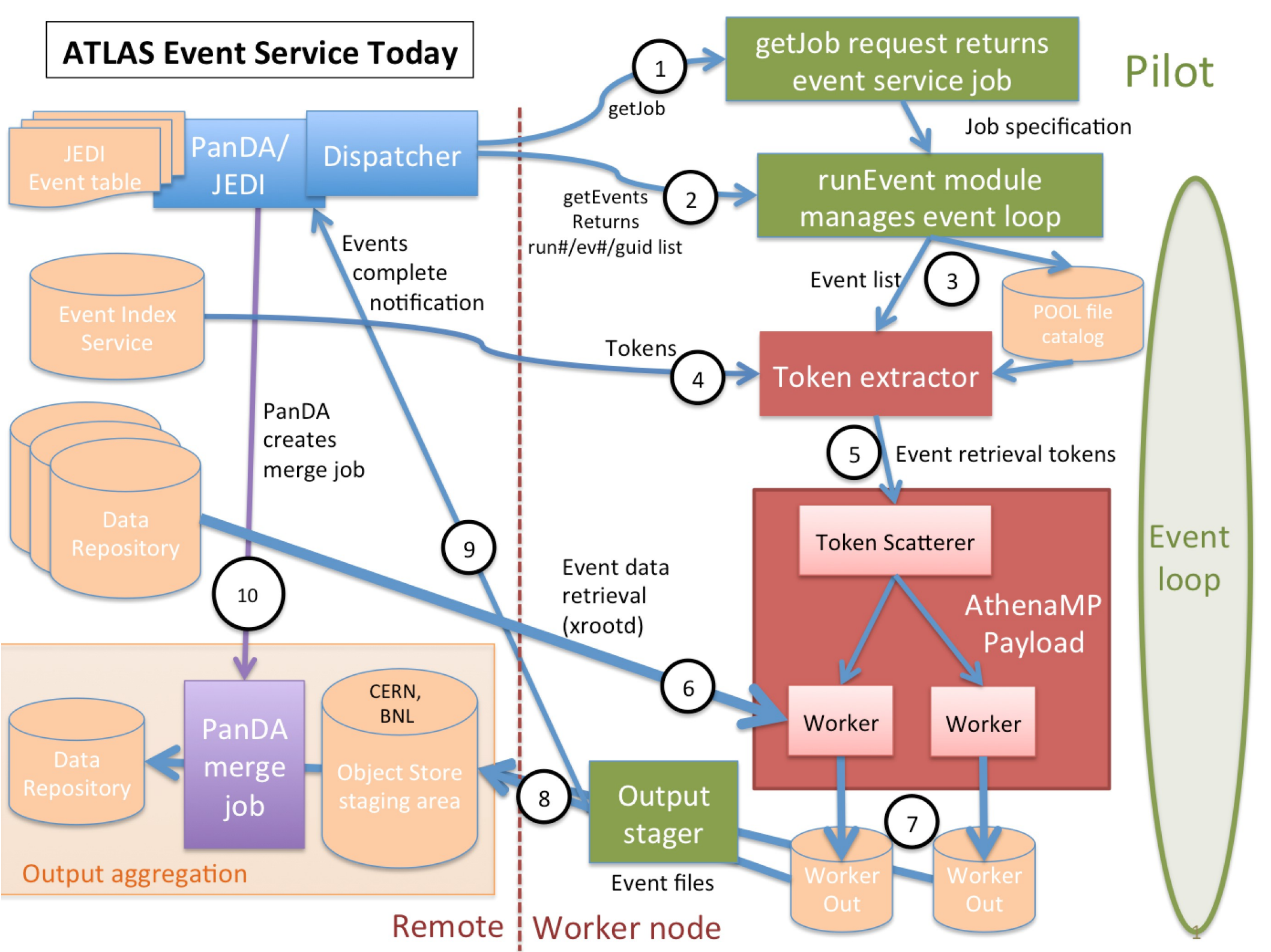
- ES's approach to outputs can have great benefit for analysis – the problem of outputs being trapped at miscellaneous sites
- Outputs stream off the WN semi-continuously, sent to a reliable large scale object store
- Sending outputs is asynchronous with processing, and outputs individually are small – the approach is friendly to small sites without the best networking (the sort that can be at greater risk of disappearing)
- No time window between “running the task” and “securing the outputs at a safe location” during which the site could go down, trapping data
- Many sites can contribute to the processing and aggregation at the destination object store
- Once the task is done, PanDA triggers a conventional merge to create a complete output at a safe location
  - Until task is fully done, PanDA's retry mechanism is working to complete it
- If there are a few percent still to be done, user could finish it off herself at her Tier 3 (next talk)

ES for Analysis – Can it work?

- It may have its in-principle benefits, but can it work?
- Is analysis too I/O intensive? Analysis can take seconds per file, not minutes per event
- But the real point of ES is more 'fine grained' than 'event level'
  - If a 15min quantum happens to represent many files, that is OK too, as long as I/O doesn't kill the concept
- There are several scenarios
  - Limit to analysis use cases that are more CPU-heavy (very limiting)
  - Make data colocation or close proximity in network terms a requirement or strong preference (also limiting)
  - Be as smart and efficient as you can in sending data over the wire, and as latency-independent as you can in the worker
    - The plan to complete the event service has this in mind...

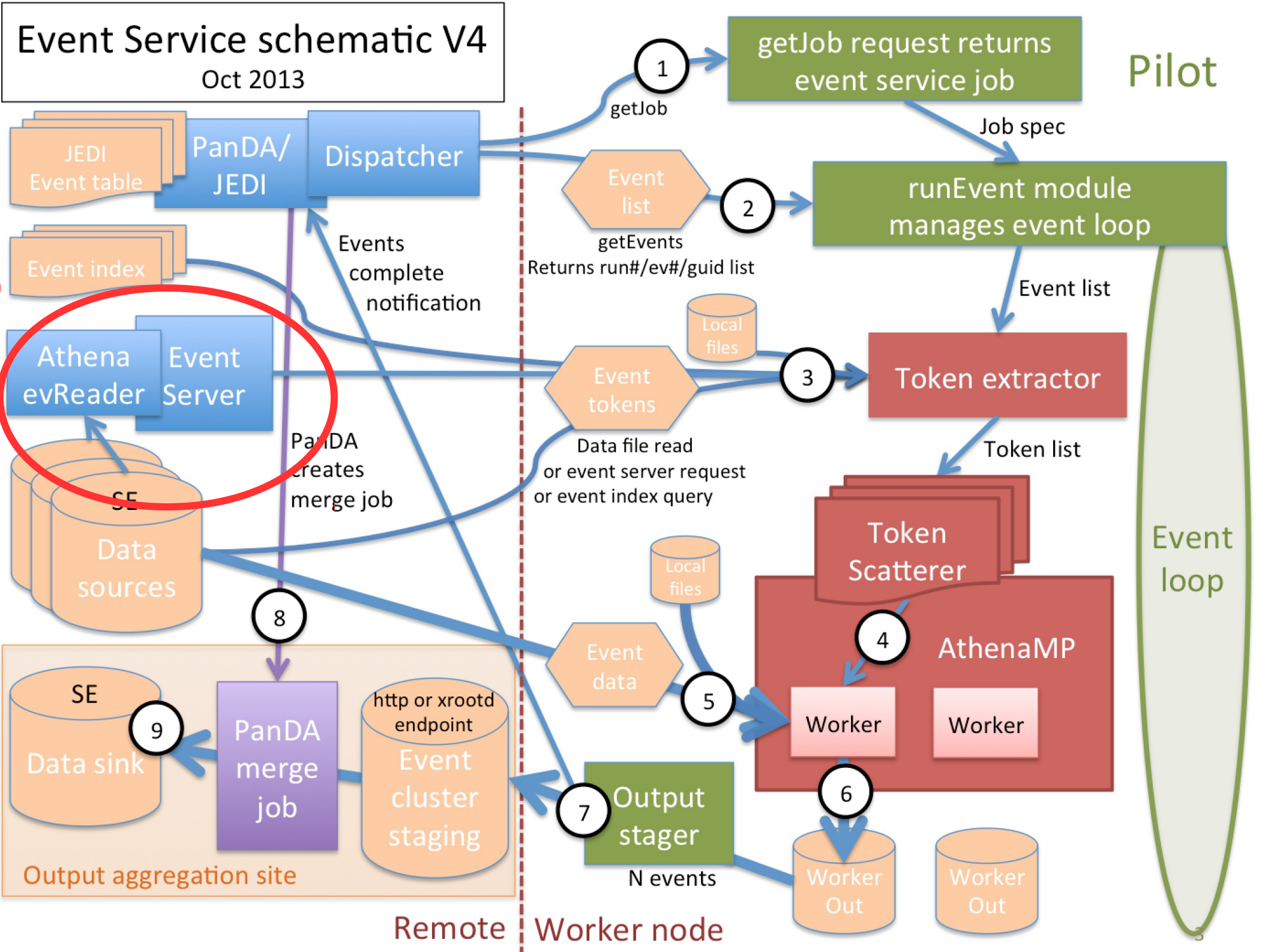
ES event delivery today

- The ES today operates with a 3-step procedure to acquire events to process
  - PanDA/JEDI assigns event ranges to the pilot
  - The token extractor converts event ranges to sets of tokens via local (tag file reading) or remote (event index) mechanisms
  - Worker processes do direct read (local or WAN) of the event data
- This is fine for CPU intensive processing that consumes 100% of its input files, like simu, but it is just a first step towards something more general and capable


ATLAS Event Service Today
getJob request returns event service job
Pilot
1
getJob
Job specification
JEDI Event table
PanDA/ JEDI
Dispatcher
getEvents Returns run#/ev#/guid list
2
runEvent module manages event loop
Events complete notification
Event list
3
POOL file catalog
Event Index Service
Tokens
4
Token extractor
PanDA creates merge job
5
Event retrieval tokens
Data Repository
Token Scatterer
AthenaMP Payload
9
Event data retrieval (xrootd)
Event loop
10
6
Worker
Worker
CERN, BNL
Data Repository
PanDA merge job
Object Store staging area
8
Output stager
7
Worker Out
Worker Out
Output aggregation
Event files
Remote
Worker node

Completing the Event Service

- The ES as originally envisioned provides a true event data delivery service, in addition to the control/bookkeeping service of PanDA/JEDI
- The event delivery process becomes
  - PanDA/JEDI assigns event ranges to the pilot
  - The payload retrieves and pre-stages event data from **event data server**
  - Workers consume the pre-staged event data
- Given knowledge of the data the consumer needs (from the task), the event data server has the opportunity to marshal only the data needed so that only what will be used (within practical limits) goes over the wire
  - A balance to be struck between degree of server-side data marshalling/compression and heaviness of the service – requires some R&D
- Fully asynchronous pre-staging of data in the WN means I/O intensive jobs separated from the data by significant latency can run efficiently
- => ES becomes capable of analysis processing

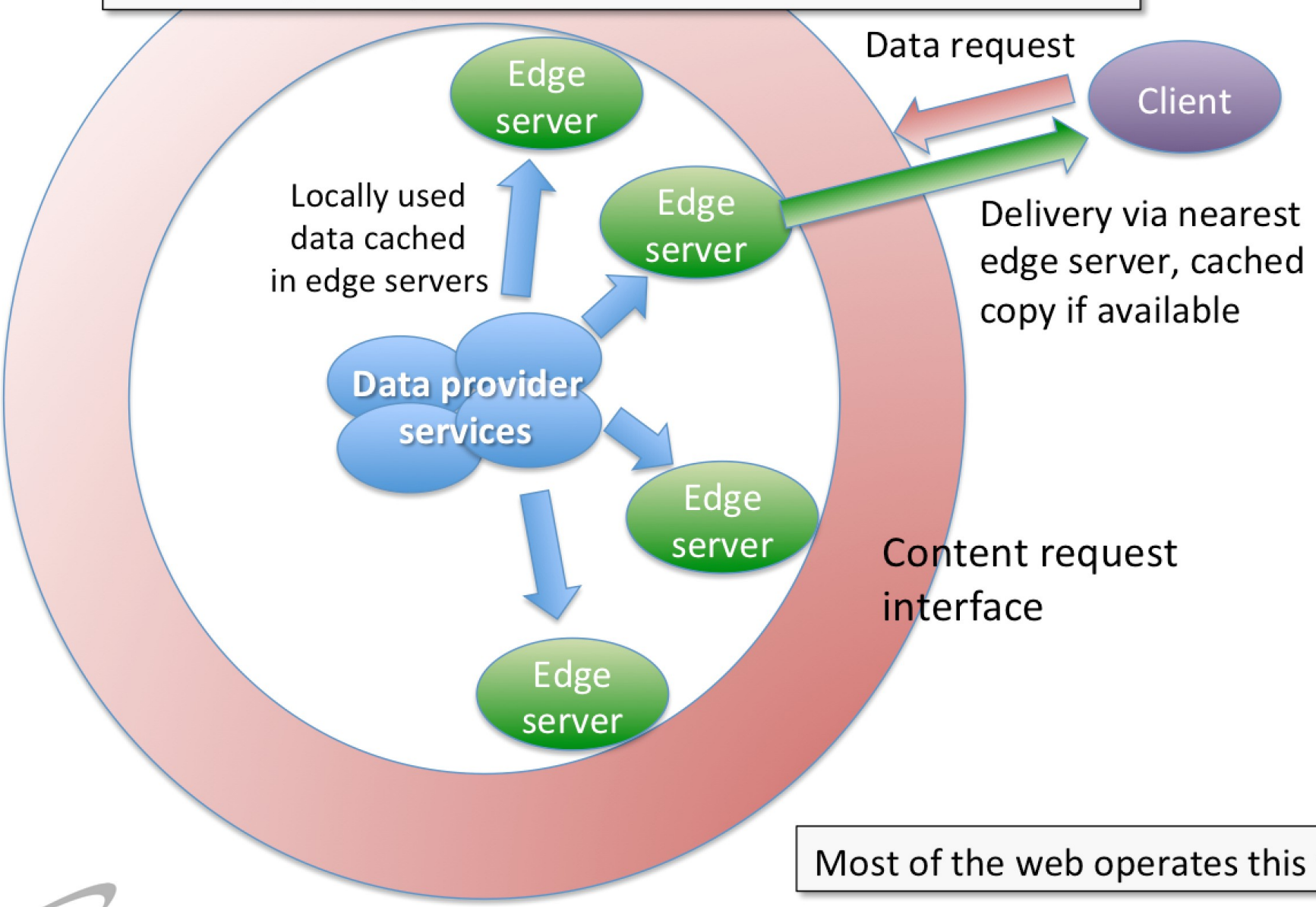

Event Service schematic V4
Oct 2013
Pilot
1
getJob
getJob request returns event service job
Job spec
JEDI Event table
PanDA/ JEDI
Dispatcher
Event list
2
runEvent module manages event loop
getEvents
Returns run#/ev#/guid list
Event index
Events complete notification
Event list
Local files
Athena evReader
Event Server
Event tokens
3
Token extractor
PanDA creates merge job
Data file read or event server request or event index query
Token list
SE
Data sources
Token Scatterer
Local files
4
AthenaMP
8
Event data
5
Worker
Worker
6
SE
9
Data sink
PanDA merge job
http or xrootd endpoint
Event cluster staging
7
Output stager
N events
Worker Out
Worker Out
Output aggregation site
Event loop
Remote
Worker node

Event Data Server (EDS)

- Content delivery network-like services have been a growing trend as HEP steadily moves towards making the most of the web it invented
- Receive a data request from anywhere, return the data by optimal 'close by' means
- EDS acts as an 'event data CDN': an event service that actually delivers events, intelligently, with locality knowledge
- Leverage WAN data delivery tools like xrootd, http
- Leverage caching transparently to the user
- Build data delivery smarts behind a very easy to use exterior
  - Make it a self-contained, well-contained, encapsulated service
- Need to take a close look at how to do the (re)direction to the 'close by' data source the client should use
- Such a service can be very friendly to small sites, Tier 3s; remove need for 'special solutions'

The Content Delivery Network Model

Content delivery network: deliver data quickly and efficiently by placing data of interest close to its clients



Most of the web operates this way

The Content Delivery Network Model

A growing number of HEP services are designed to operate broadly on the CDN model

Service	Implementation	In production
Frontier conditions DB	Central DB + web service cached by http proxies	~10 years (CDF, CMS, ATLAS, …)
CERNVM File System (CVMFS)	Central file repo + web service cached by http proxies and accessible as local file system	Few years (LHC expts, OSG, …)
Xrootd based federated distributed storage	Global namespace with local xrootd acting much like an edge service for the federated store	Xrootd 10+ years Federations ~now (CMS AAA, ATLAS FAX, …) *See Brian's talk*
Event service	Requested events delivered to a client agnostic as to event origin (cache, remote file, on-demand generation)	ATLAS implementation coming in 2014
Virtual data service	The ultimate event service backed by data provenance, regeneration infrastructure	Few years?

Event Data Server (2)

- An ES job has an associated, optimally chosen EDS instance best able to serve up the job's data
- Client (agent of the payload) makes its data requests to the EDS CDN front end, transparently redirected to the chosen instance
- The EDS instance delivers data to the client, transparently optimizing the delivery
  - Local file source if available, local cache if available, nearest replica in network terms, on-demand generation in the 'virtual data' future...
- This is a sketch; it needs to be designed and developed if we agree it is well motivated

ES 2015 Summary & Conclusions

- The SC14 hard deadline and intense efforts by our excellent cadre of developers have brought the ES to the threshold of production deployment
- Time to seriously address where and how we deploy it
- And address where we go next
- Analysis is a big open territory, with substantial potential benefits, requiring substantial work for the most general use cases
- Rewards could extend to improving support for analysis users 'at home': beyond pledge resources, Tier 3s, laptops...
- Which brings us to the next topic of beyond pledge scenarios