Fine grained processing with an Event Service

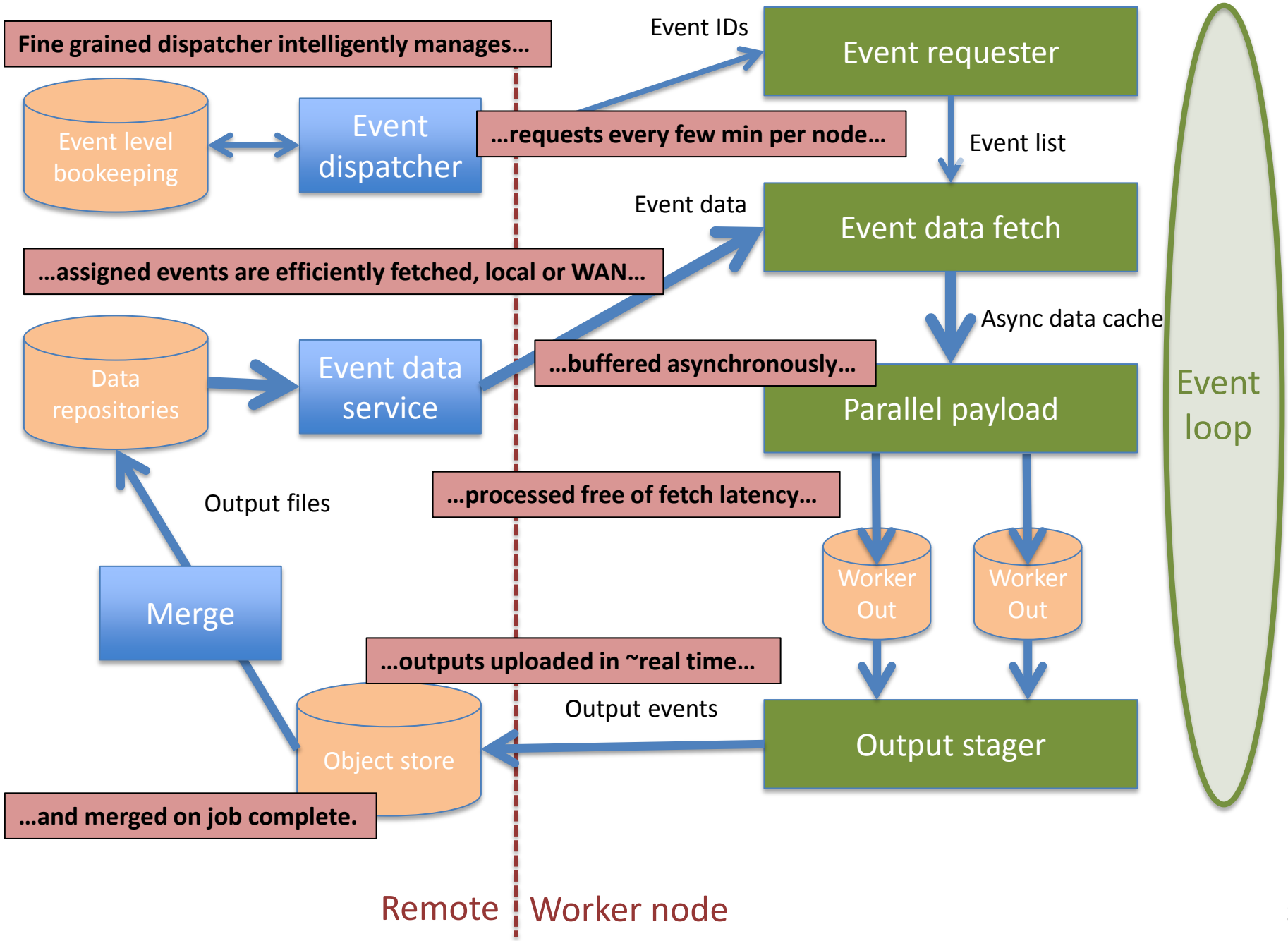
Torre Wenaus (BNL)
Vakho Tsulaia (LBNL)
for the ATLAS Event Service team

Jan 21, 2015
HSF Workshop, SLAC

Event Service

- A new fine grained approach to event processing: near-continuous event streaming through a worker node
- Easily, efficiently, fully exploit workers through their lifetime, whether that is 30 minutes or 30 hours or 10ms from now with no notice
- Decouple processing from the chunkiness of files, from data locality considerations, from WAN access latencies
- Export outputs continuously, negligible losses if the worker vanishes, keeps local storage demands low, promptly places data in a secure standard place
- Great for opportunistic resources
  - 'Full' HPCs are full of big hulking rocks; they still have plenty of room for sand, for those able to efficiently pour fine grained work into the cracks
  - Amazon spot market rewards short-lived, transient workers
  - Volunteer computing (BOINC) rewards robustness against unreliable unpredictable transient workers
- Managers of 'conventional' resources, especially VM/cloud based, love the idea of workloads that can be instantaneously jettisoned with negligible losses

Event Service Schematic



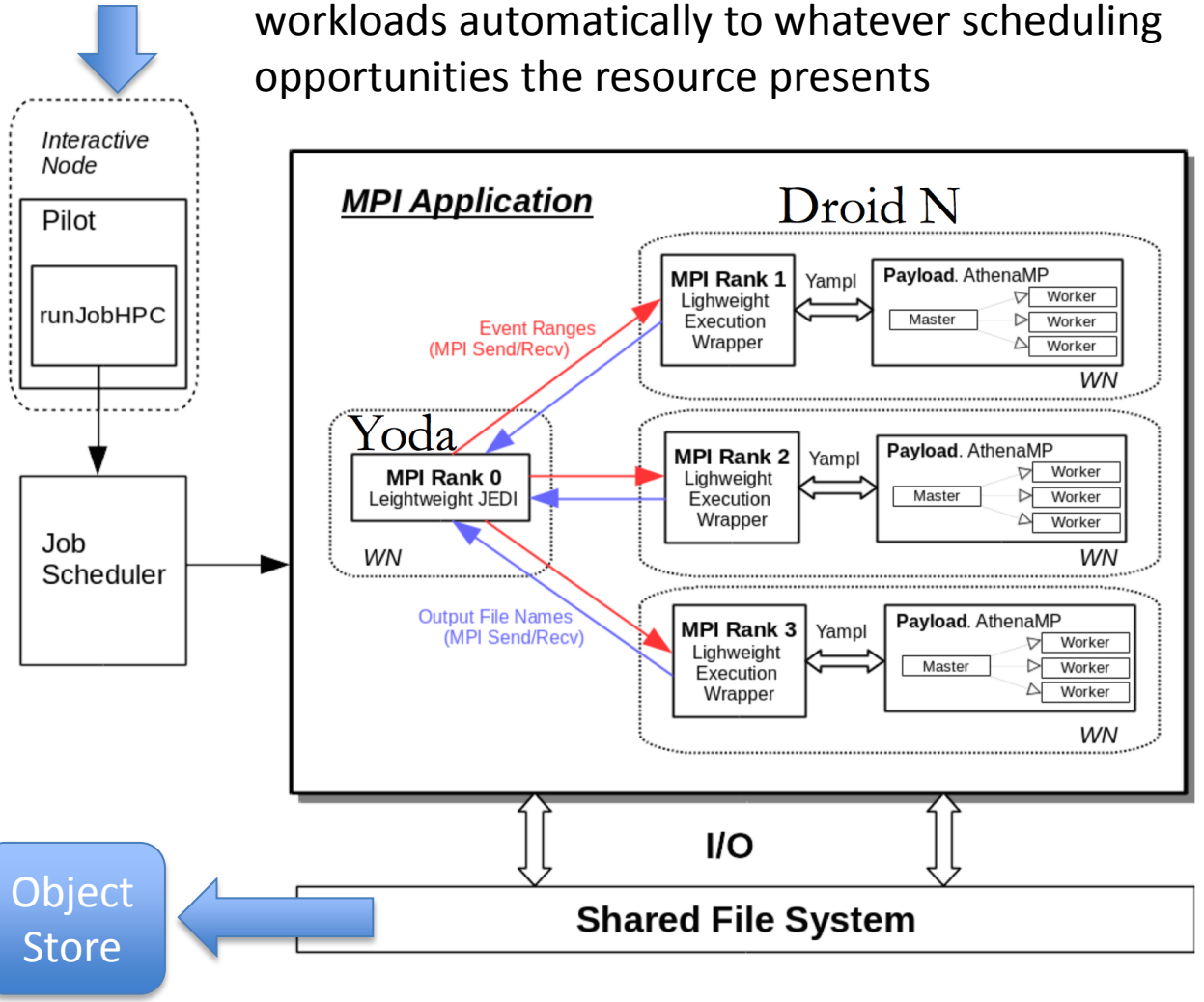
Yoda

PanDA's JEDI based event service miniaturized for HPCs

On HPCs, the MPI-based master/client adaptation 'Yoda' of the Event Service allows tailoring workloads automatically to whatever scheduling opportunities the resource presents

- Work assignments stream in with fine granularity
- Outputs streamed promptly to secure location
- Processing proceeds until slots die, with full utilization

Offers the efficiency and scheduling flexibility of preemption without the application needing to support or utilize checkpointing

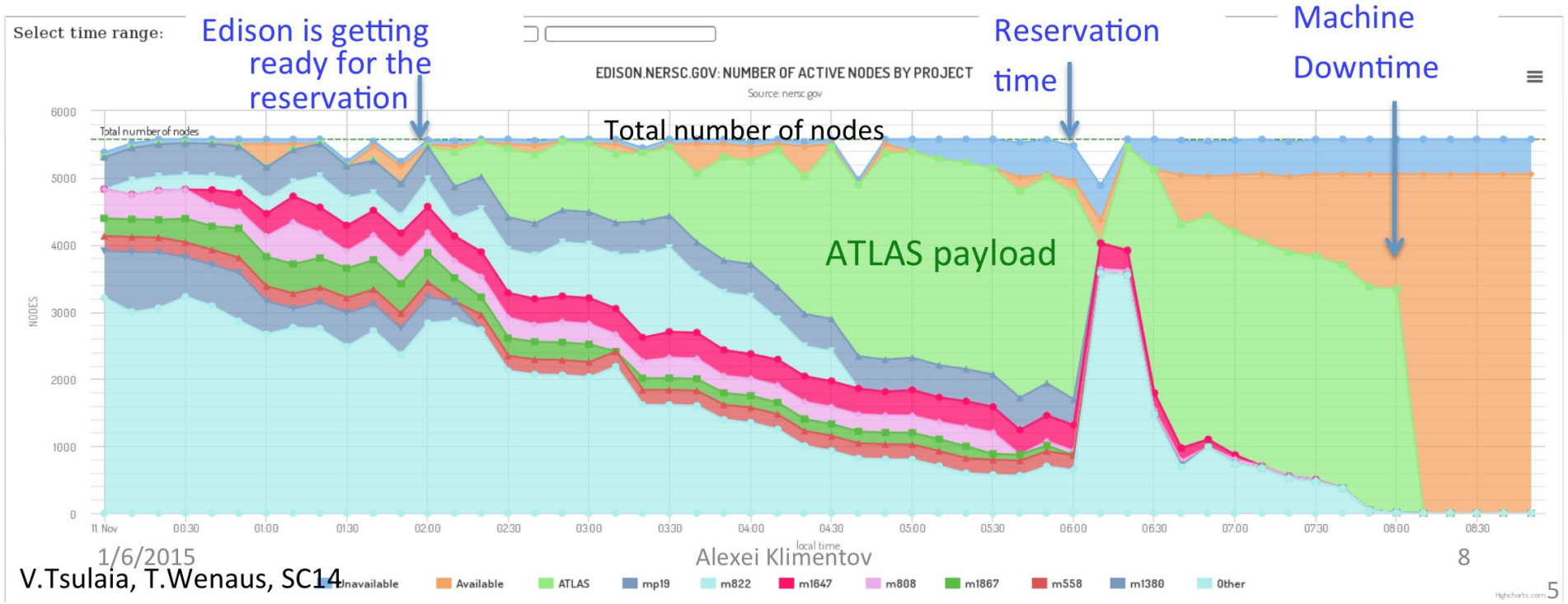


Beneficiary of common project support: DOE BigPanDA for HPC and exascale data intensive computing

Demoed at Supercomputing 2014 as a DOE ASCR Data Demo http://goo.gl/WSdU4a

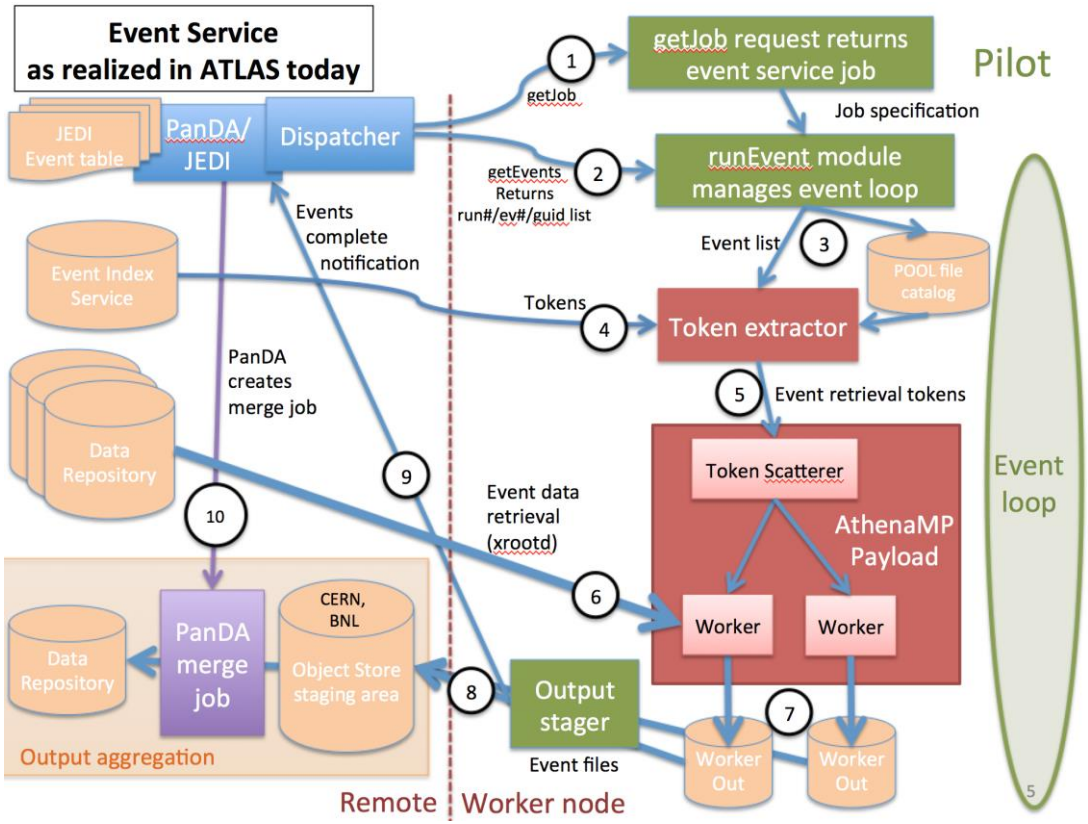
Yoda scavenging resources @ NERSC Edison

- As the machine is emptied for downtime or large usage blocks, a killable queue makes transient cycles available
- Yoda sucks them up efficiently and processes events until the moment they vanish, with negligible losses to the processing
- And refills when they appear again



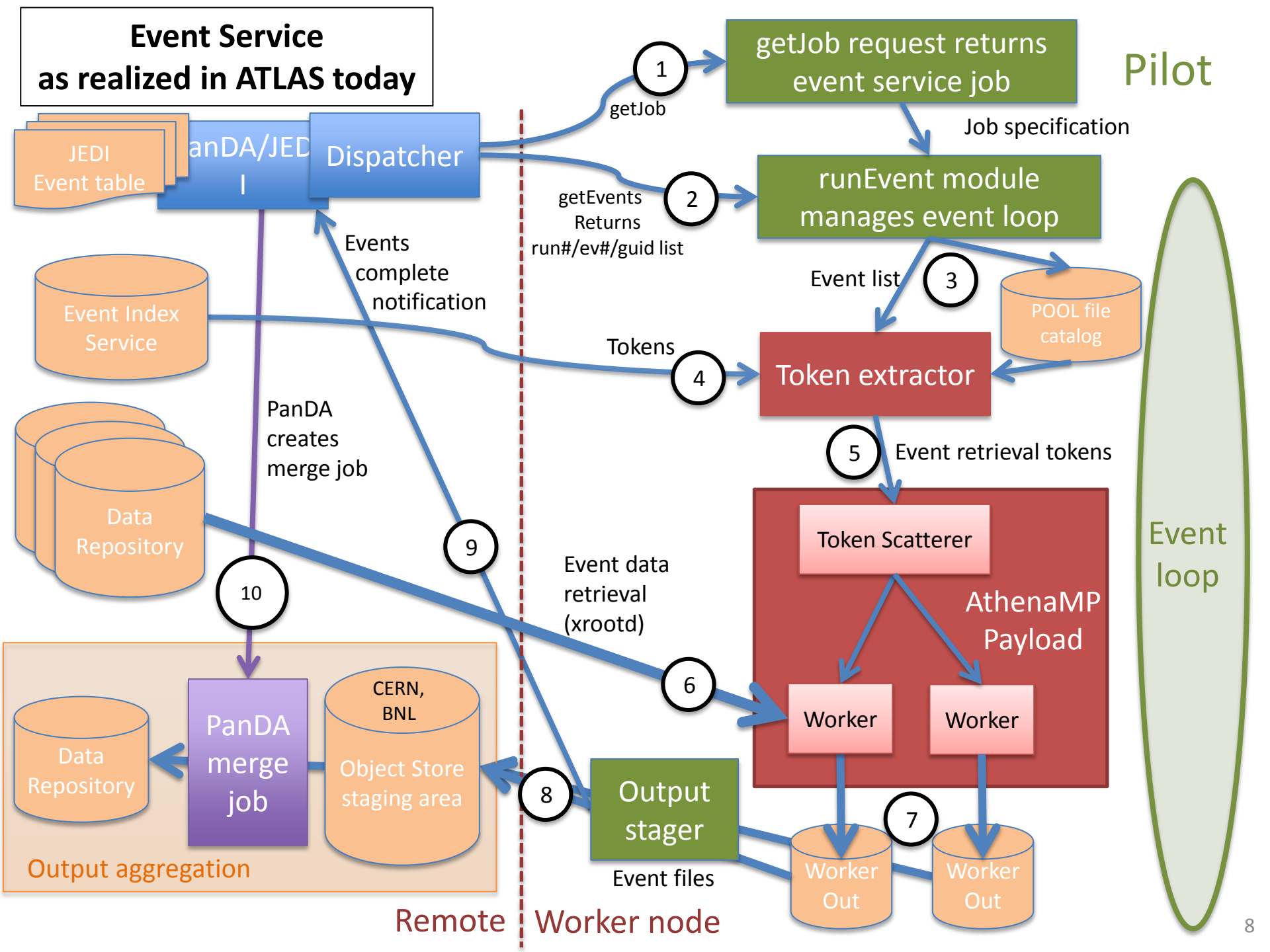
Event Service in ATLAS

- Operational on grid, cloud (Amazon spot market), HPC (NERSC Edison so far)
- Outputs to object stores at BNL, CERN
- No scaling issues seen outside payload-dependent HPC issues
- Entering physics validation and production commissioning on ATLAS grid, clouds
- BOINC underway: ATLAS@home
- Currently simulation-only (the biggest return for the least investment); other payloads expected to follow



The punchline:
ES (or elements thereof) as a common project?

- **Generalize beyond PanDA** as the workload manager?
- Common solution for highly granular, scalable event-level **bookkeeping database**?
- Standardize elements of the **granular workflow**: intelligent and flexible dispatch, brokerage, retry, auto-completion, auto-merge?
- Standardize **MPI parallelization of fine grained workflows on HPCs**?
- Integrate new **payload frameworks** beyond athenaMP?
- Share the work of extending the approach to **new platforms**, more sophisticated workloads, new processing stages (e.g. ROOT analysis)?
- Standardize **object store** based management of fine grained data?
- Applicable to **any scientific processing** that can be finely partitioned (processing and outputs)
- **If you might be interested, talk to us!**


Event Service
as realized in ATLAS today
JEDI
Event table
anDA/JED
I
Dispatcher
1
getJob
getJob request returns
event service job
Pilot
Job specification
runEvent module
manages event loop
2
getEvents
Returns
run#/ev#/guid list
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
9
Event data
retrieval
(xrootd)
Token Scatterer
AthenaMP
Payload
10
6
Worker
Worker
Event
loop
CERN,
BNL
PanDA
merge
job
Data
Repository
Object Store
staging area
8
Output
stager
7
Worker
Out
Worker
Out
Output aggregation
Event files
Remote
Worker node