ExaO: Traffic Optimization for ExaScale Science Applications

Justas Balcas[2], Greg Bernstein[3], Haizhou Du[4], Azher Mughal[1], Harvey Newman[1], **Qiao Xiang**[5], Y. Richard Yang[5], Jingxuan Zhang[4]

[1] California Institute of Technology, [2] CERN, [3] Grotto Networking, [4] Tongji University, [5] Yale University

March 31st, 2017, IETF98, Chicago

Next-Gen Integrated Systems for Exascale Science

- **Vision:** resource in worldwide-distributed environments should be deployed flexibly to meet the demands of exascale science applications.
- **Goal:**
  - Production deployments of a new class of intelligent, software-defined global systems for data-intensive science programs involving a worldwide ensemble of sites and networks.
  - A game-changer for next-generation science data flow orchestration that shapes the future architecture and operational modes of exascale computing facilities.
- **Members**: worldwide multi-organizational collaboration between Caltech, CERN, Tongji University and Yale University.

Next-Gen Integrated Systems for Exascale Science

- **Timeline:**
  - – Pre-production deployment in 2017.
  - – Production deployment in ~2018-19.
  - – Worldwide deployment of such systems in ~2020-24.

LHC: Large Hadron Collider

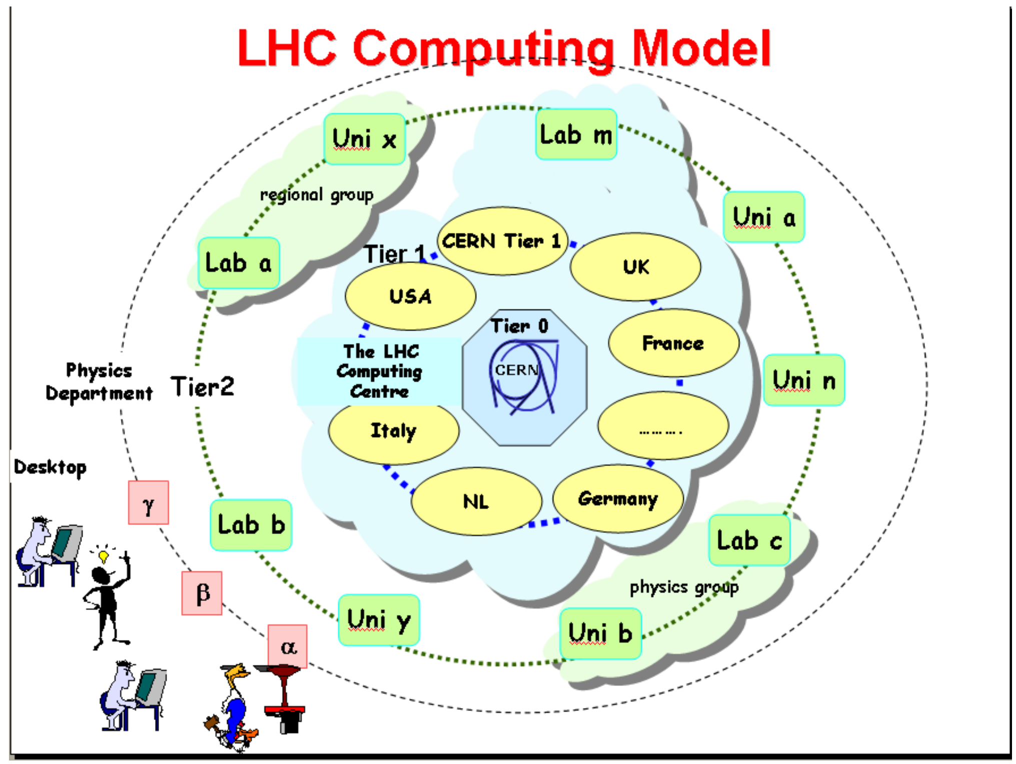


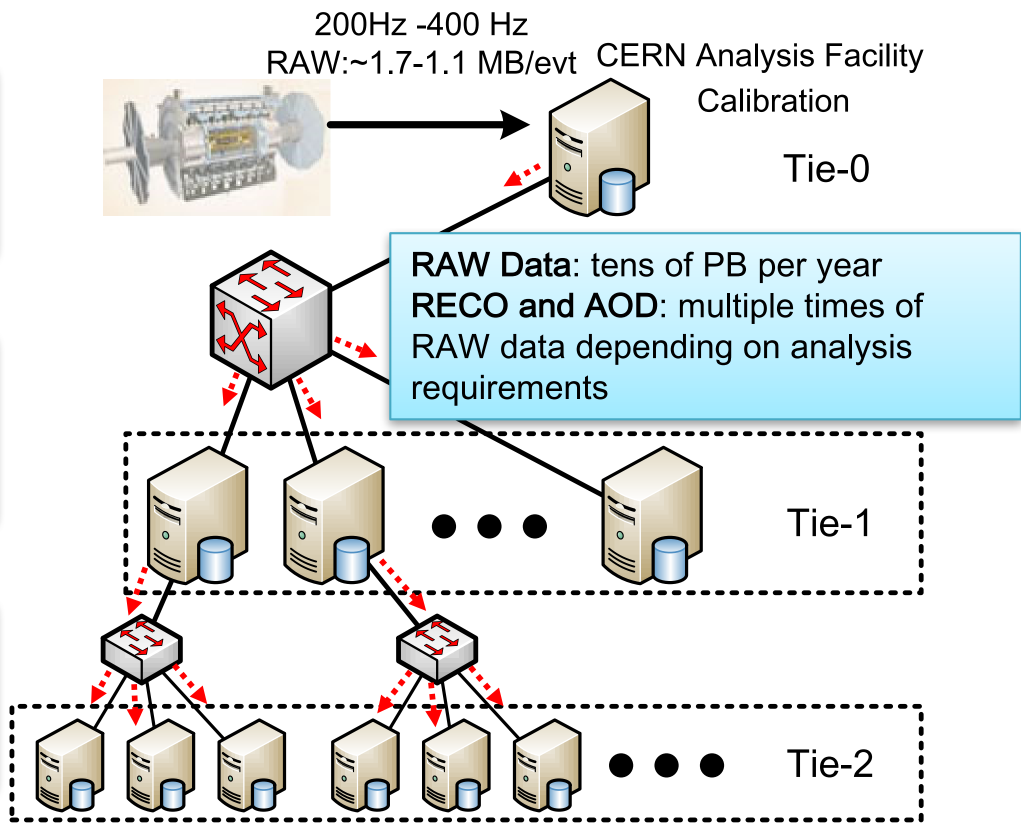
Figure source: cern.ch

The Compact Muon Solenoid Computing Model

Large raw datasets from LHC at the Tier-0 site

RECO and AOD datasets are distributed to Tier-1 sites

RECO, AOD and simulation datasets are transferred among Tier-1~3 sites for analysis.



Problem Settings

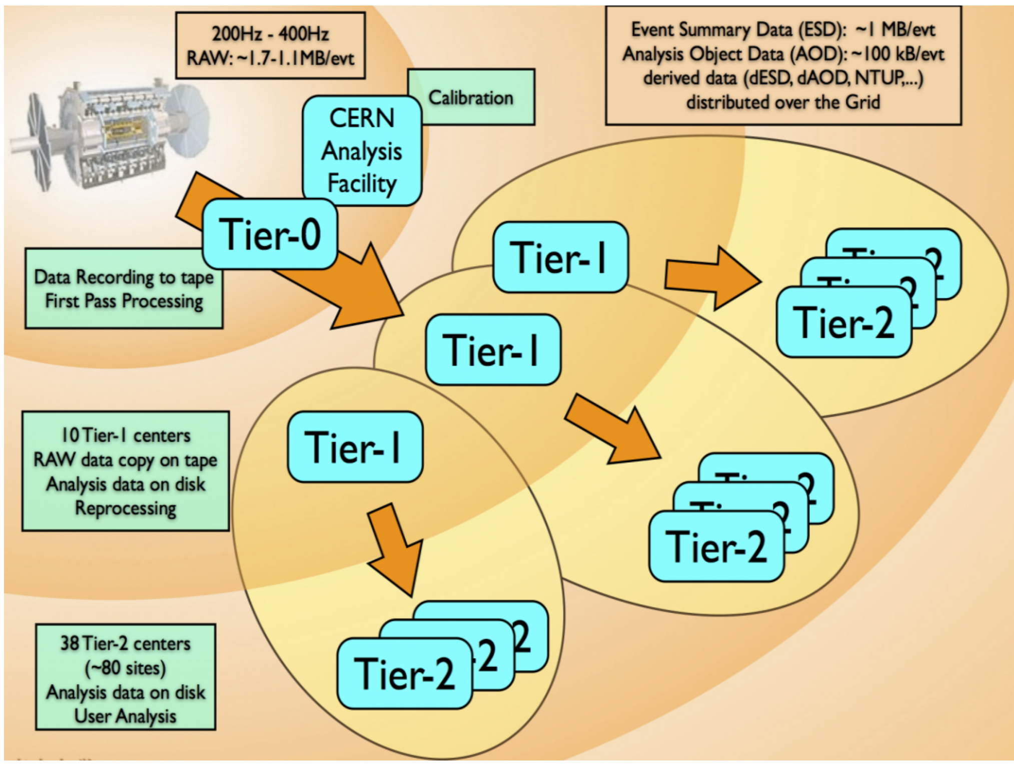


Figure source: cern.ch

- A multi-domain data center network
- Heterogeneous resource /system provision
- Various jobs
  - Exascale dataset transfers
  - MapReduce analytics
  - MPI analytics
  - etc.

Takeaway from Version 00

- Use ALTO topology services (path-vector and routing state abstraction) to provide on-demand fine-grained network information from different sites/domains.
- Use such information to orchestrate dataset transfer scheduling.
- Prototype demo at SC'16.
- Currently under production development.

Update in Version 01

- Dataset transfer is only one application in CMS.
- ALTO has the potential to improve the performance of analytic applications.
  - In CMS DC networks, network resource is not always the bottleneck.
- ExaO: a multi-resource orchestrator for CMS applications.

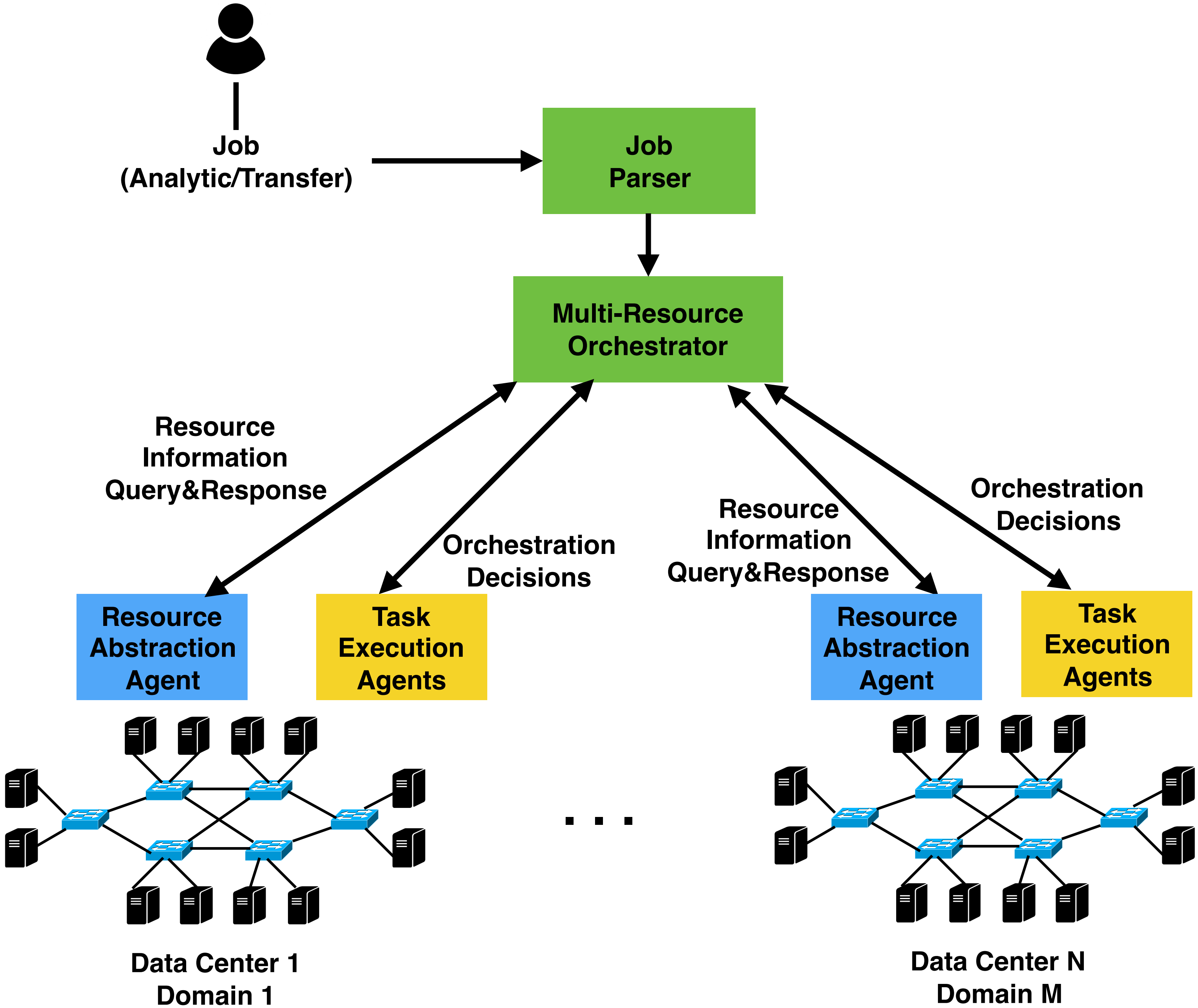
Impact on ALTO Working Group

- ExaO is a representative ALTO application in data center networks, which is a major use case of ALTO listed in the WG Charter.
- We expect the success of ExaO to provide key insights and experience in deploying ALTO services and developing ALTO applications.

Overview

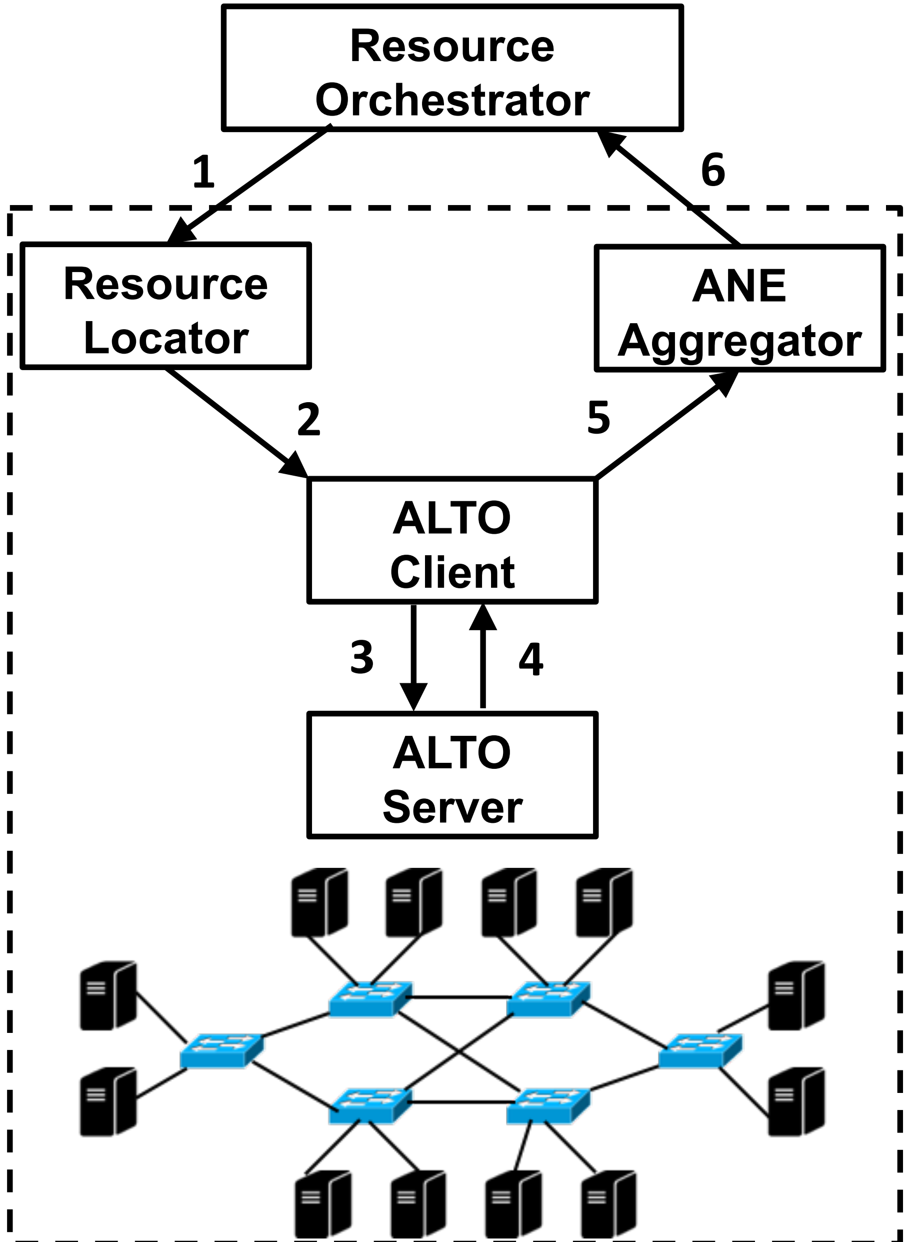
- Deploy ALTO clients and servers at sites and networks to retrieve endpoint properties and topology information.
- Expand the capability of abstract network element (ane) to provide an abstract view of computing, storage and networking resources.
- Use such views for deep site orchestration among virtualized clusters, storage subsystems and subnets to successfully co-schedule CPU, storage and networks.

ExaO Architecture



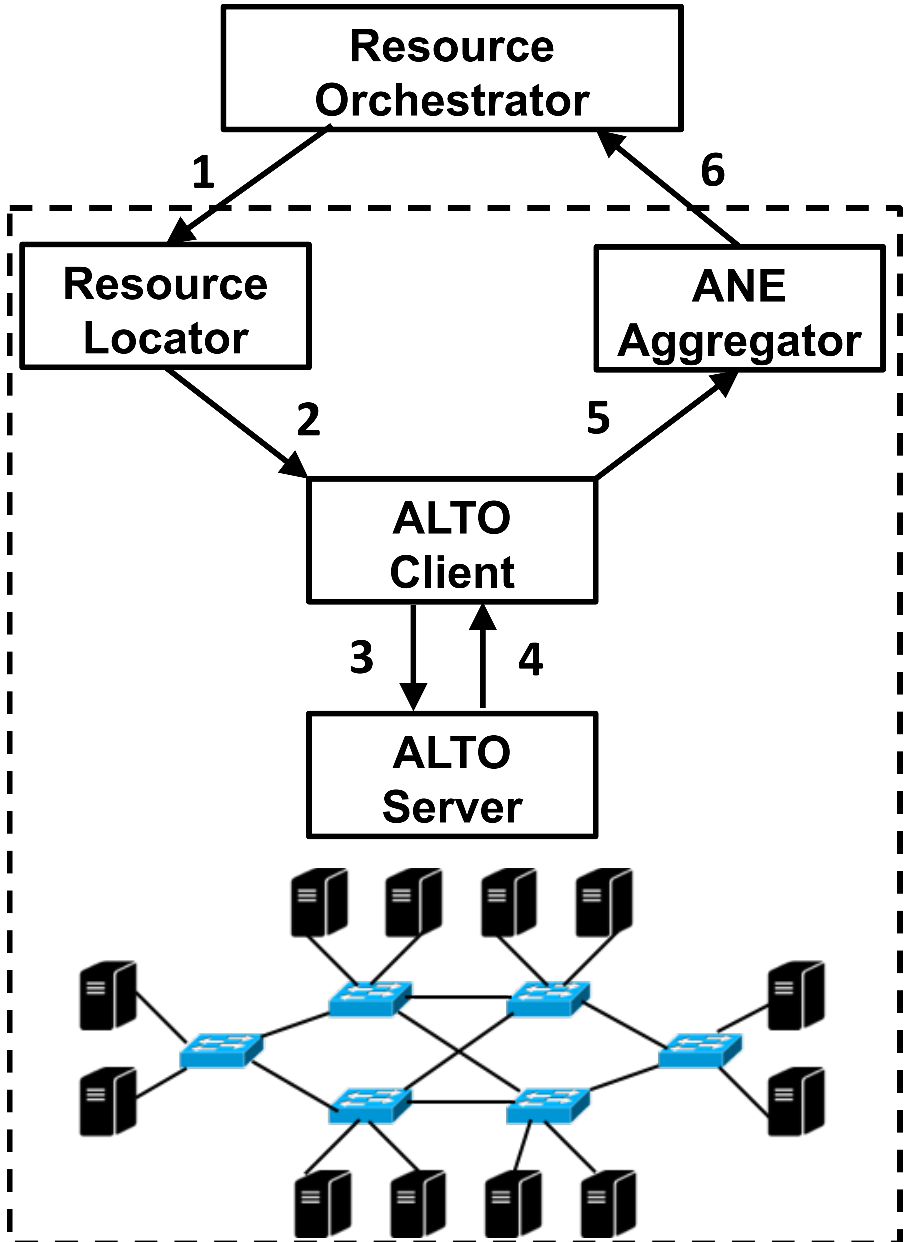
Resource Abstraction Agent

1. Orchestrator sends input dataset name and job information to resource locator.
2. Locator talks to resource management system to get endpoint addresses of available resources for the job and sends to ALTO client.
3. ALTO client issues queries to ALTO server.
   - EPS query to get properties of resource nodes.
   - PV-based query to get properties of ane connecting resource nodes.



Resource Abstraction Agent

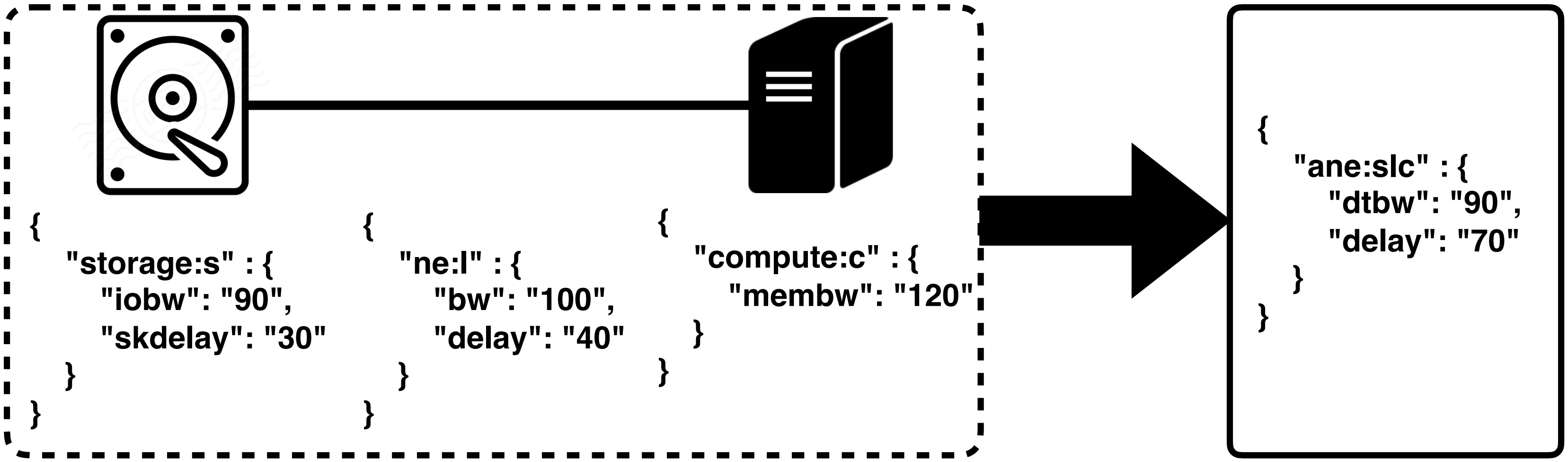
4. ALTO server executes the query and send the response to ALTO client.
5. ALTO client sends the information of resource nodes and ane to ANE aggregator.
6. ANE aggregator encodes the received information into a single ane and sends back to the orchestrator.



ANE Aggregator: Provide a One-Big-ANE View

- **Motivation**: existing resource node abstractions, e.g., HTCondor and YARN, only provide coarse-grained resource information, e.g., # of cores and size of memory.
- **Basic idea**:
  - Abstract the set of resource nodes and the connecting network available to the job into a single ane.
  - The output ane provides an abstract view of computing, storage and networking resources.
  - Each property of each resource is encoded into a property of ane.
  - Properties from different resources are merged to reduce information overhead and privacy exposure.

ANE Aggregator: An Example



- ane.dtbw = min{s.iobw, c.membw, l.bw}
- ane.delay = s.skdelay+l.delay
- Property merging is motivated by ALTO-PV and ALTO-RSA and is crucial for privacy preserving.

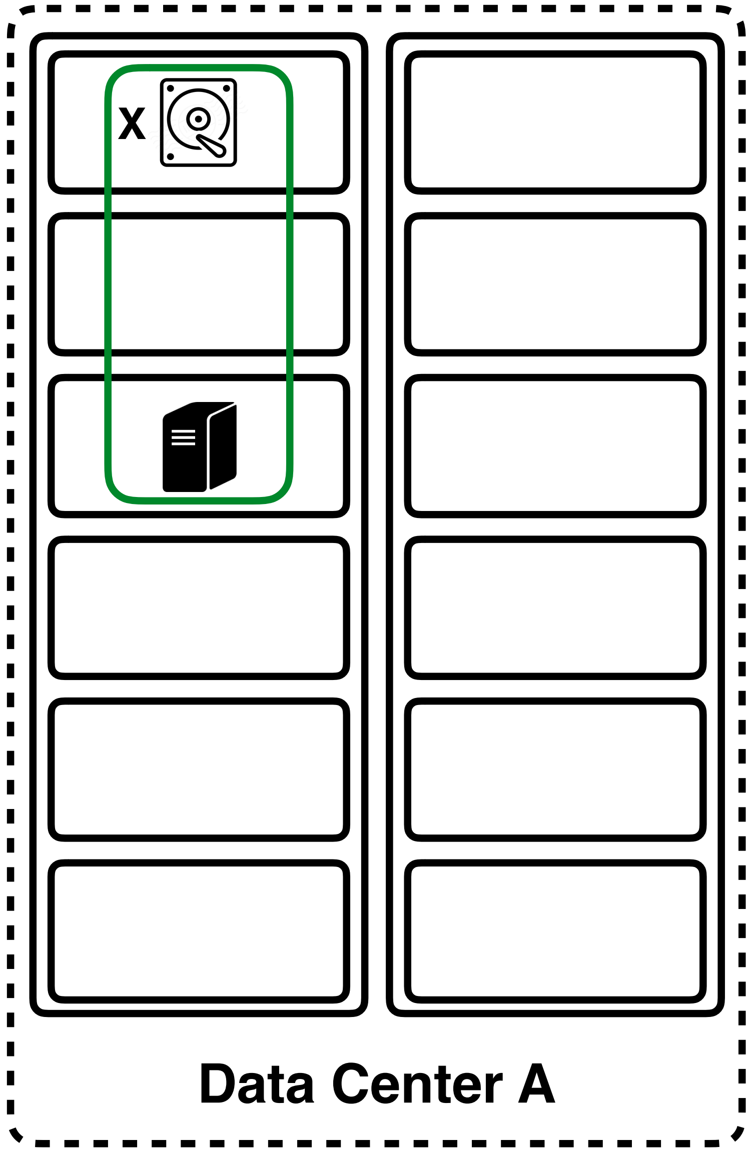
Design Issue: Why ANE Instead of PID?

- The abstraction requested by job is data-oriented and highly dynamic.
- PID is a static abstraction decided only by sites.
- ANE is a dynamic abstraction decided by both sites and applications.
  - Sites: policy, regulation, etc.
  - Applications: dataset name, job properties, preferred resource, etc.

Resource Abstraction Agent: An Example

- Job J needs dataset X as input.
- Data center A and B eachh has a copy of X.

- Resource locator in data center A finds J would be placed on a different node from X's location. Both nodes are in the same rack.



Data Center A: EPS Query of Storage Node

- **Request**


```
{
    "endpoints": ["ipv4:10.0.0.1"],
    "properties": ["iobw", "skdelay"]
}
```


- **Response**


```
{
    "meta" : {
        "dependent-vtags" : [...]
    },
    "endpoint-properties": {
        "ipv4:10.0.0.1" : {
            "iobw": "150",
            "skdelay": "30"
        }
    }
}
```

Data Center A: EPS Query of Computing Node

- **Request**


```
{
    "endpoints": ["ipv4:10.0.0.5"],
    "properties": ["membw"]
}
```


- **Response**


```
{
    "meta" : {
        "dependent-vtags" : [...]
    },
    "endpoint-properties": {
        "ipv4:10.0.0.5" : {
            "membw": "200"
        }
    }
}
```

Data Center A: PV-based ECS Query

- **Request**

```
{
    "cost-type": {
        "cost-mode": "path-vector",
        "cost-metric": "ane"
    },
    "endpoints": {
        "srcs":["ipv4:10.0.0.1"],
        "dsts":["ipv4:10.0.0.5"]
    }
}
```

- **Response**

```
{
    "meta" : {
        "vtag": [...
            "query-id":"query_0"],
        "dependent-vtags": [...],
        "cost-type": {
            "cost-mode": "path-vector",
            "cost-metric": "ane"
        }
    },
    "endpoint-cost-map": {
        "ipv4:10.0.0.1" : {
            "ipv4:10.0.0.5":  ["ane:l1"]
        }
    }
}
```

Data Center A: PV-based ANE Property Query

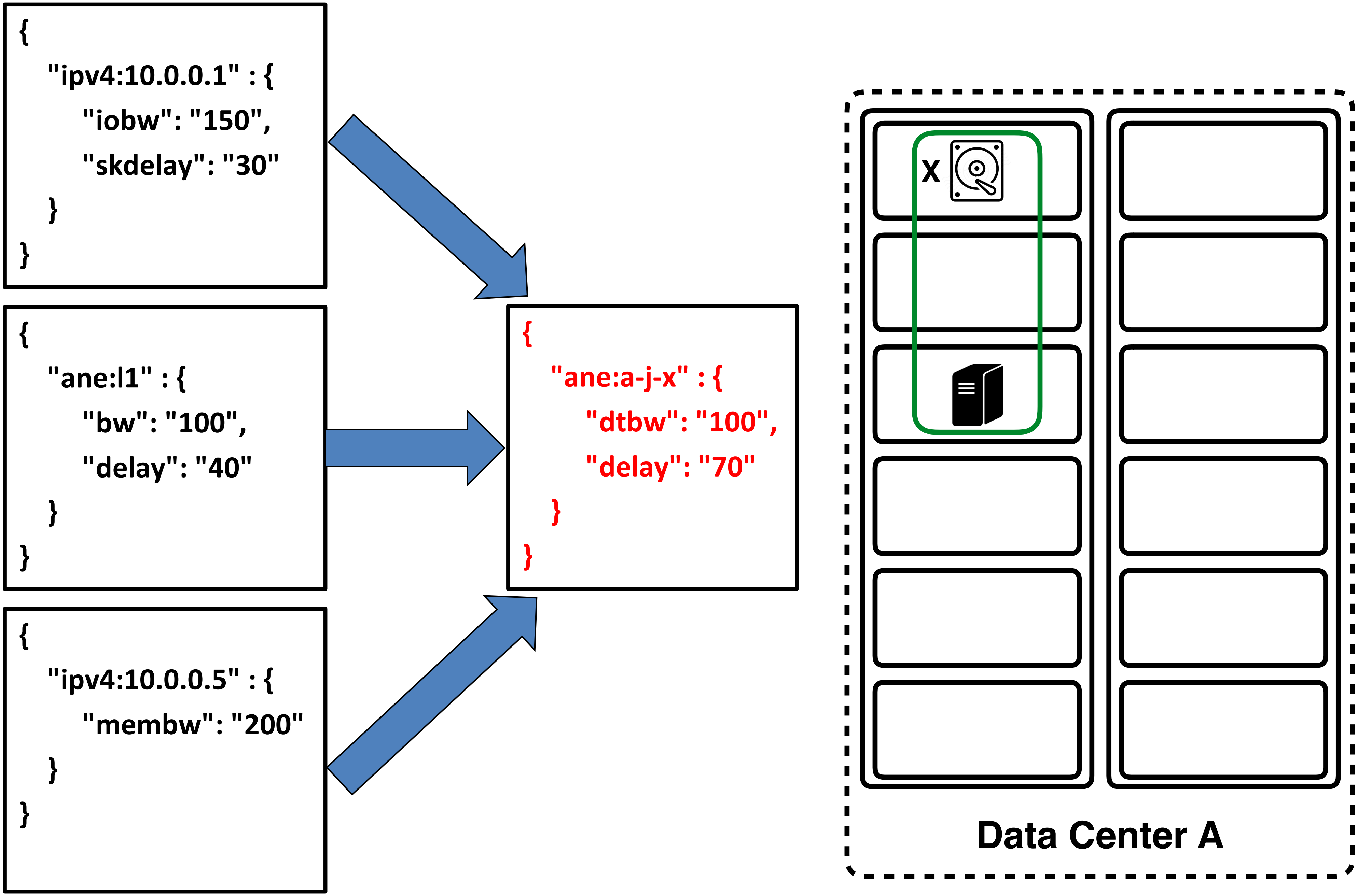
- **Request**

```
{
    "query-id": "query_0",
    "entities": ["ane:l1"],
    "properties": ["bw", "delay"]
}
```

- **Response**

```
{
    "meta" : {
        "dependent-vtags": [...]
    },
    "property-map": {
        "ane:l1": { "bw": "100",
                    "delay": "40"}
    }
}
```

Data Center A: ANE Aggregator

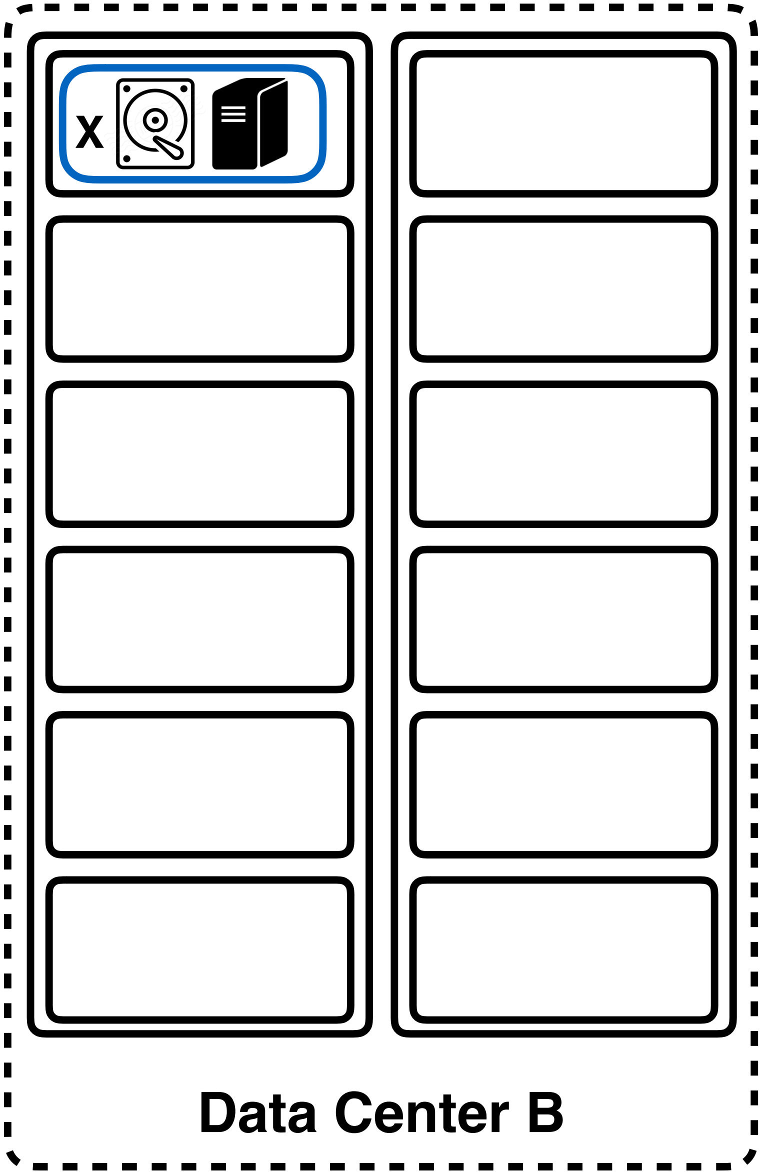


Data Center B: ALTO Query + ANE Aggregator

- Resource locator in data center B finds J would be placed on the same node where X is stored.

```
{
   "ipv4:10.0.0.1" : {
      "membw": "200",
      "iobw": "150",
      "skdelay": "20"
   }
}
```

```
{
   "ane:b-j-x" : {
      "dtbw": "150",
      "delay": "20"
   }
}
```



J should be placed at data center B because it provides a better data transfer bandwidth and delay.

Design Issue: Aggregator Inside/Outside ALTO

- ANE property aggregation is motivated by ALTO-RSA.
- How different resources are further encoded into ane is dynamic and application-specific.
- ANE aggregator as an ALTO service?
  - Pro: enrich ALTO's control capability on privacy leakage.
  - Con: ALTO is supposed to be agnostic of applications.
- **Current design**
  - Resource locator only passes endpoint address to ALTO client.
  - ANE aggregator works as an independent module instead of an ALTO service.
  - This design is modular.

Make and Enforce Resource Orchestration Decisions

- Decisions include
  - where to place analytic processes, and
  - where to transfer/store intermediate/final results.
- Under certain cases, decisions also include copy a hot dataset to another location before placing analytic processes.
- Decision enforcers are implemented on top of current resource management systems in each site.
- ALTO provides fine-grained resource information to improve the performance of these two components.

Summary

- ExaO expands the capability of abstract network element (ane) to provide an abstract view of computing, storage and networking resources, which supports the efficient resource orchestration of data-intensive applications.
- We are also exploring the feasibility and benefit of applying cost-calendar and flow cost service in supporting next generation science data flow orchestration.
- **Milestones**
  – Pre-production deployment of ExaO by IETF 100.
  – Production deployment by IETF 102-103.

Backup Slides

CMS Science Network

- Geographically distributed
- Multi-domain
  - Each domain has its own policy.
  - Resource information is private.
- Heterogeneous

Heterogeneity

- Heterogeneous resources
  - CPU: AMD, Intel (various specs…)
  - Storage: Tape, SATA hard-drive, SSD, etc.
  - Networking: 1/10/40/100Gbps
- Heterogeneous systems
  - File systems: HDFS in US sites, dCache/NFS in European sites, EOS/CephFS at CERN, etc.
  - Schedulers: even Hadoop provides different scheduling policies
- Heterogeneous jobs
  - Dataset transfers
  - MapReduce analytic
  - MPI analytic

Example 2: File System Becomes Bottleneck

- A MapReduce job J needs dataset X as input
- X has a copy in site A using HDFS and another copy in site B using NFS
- J should run at site A since HDFS provides a better data throughput, which is a key factor for MapReduce job latency.

Example 3: Copy Data First?

- Two jobs J1 and J2 need a dataset X as input.
- X only has one copy at site A and A has abundant resources.
- Scheduling Option 1: executing J1 and J2 sequentially.
- Scheduling Option 2: assign the nodes storing X to J1, and assign nearby nodes to J2.
- Scheduling Option 3: make an additional copy of X to a set of idle nodes in A. J1 and J2 can then run at the same time.
- Which option is better depends on
  - How long would each job run?
  - How large is X? (In other words, how long does it take to make another copy)