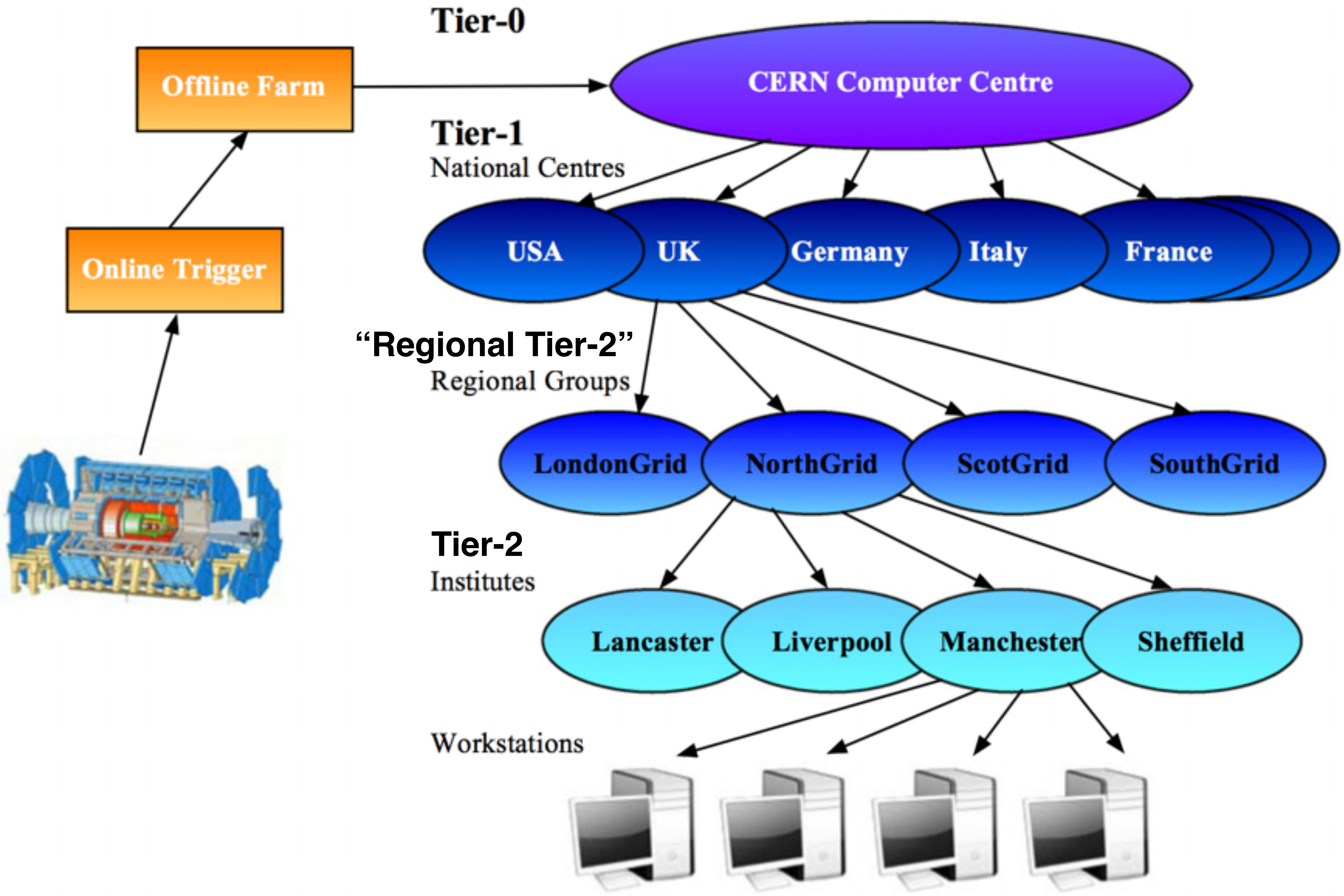
GridPP, WLCG, HEP Data Management

(WLCG?) Axioms

- Data is *Immutable* - “transforms” generate new data.
- Experiments are most important
    - Need/require control/specialisation
    - *Can’t make everything “common”*
    - *Can’t make changes during data taking.*
- Most data *is* in a *‘common’* format *(ROOT)*
- Many sites - need federated/distributed identity.

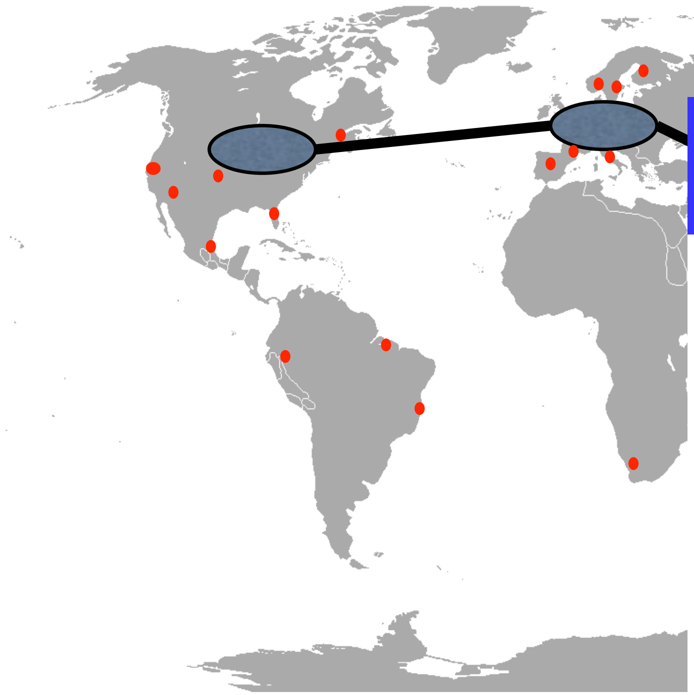
Run 1 WLCG Context



Workflow and Data Management

Big centers for data reduction impacts workflow and data management

- Data selection workflow sits on top of "big data" tools
  - Focusing effort on reproducibility and shared selection criteria
- Data Management involves moving small samples to end sites



Relative Size of Things

Processing

Amazon has more than 40 million processor cores in EC2

Google has ~1M servers so ~20M cores

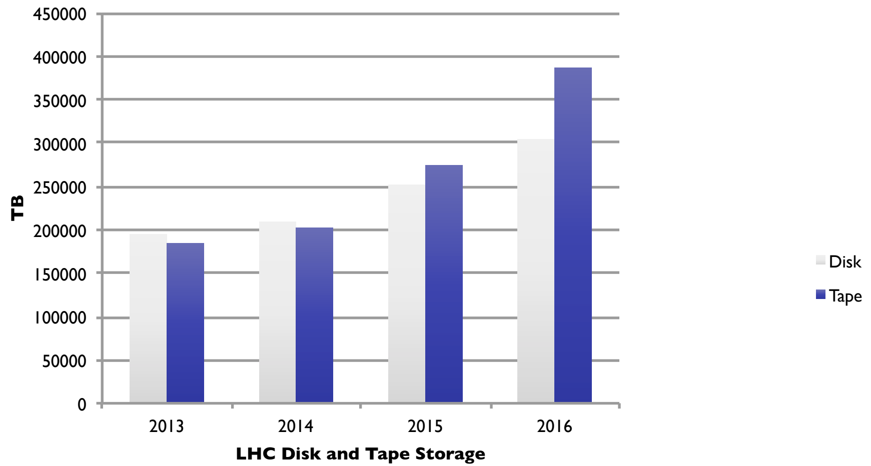
Storage

Amazon supports millions of queries per second

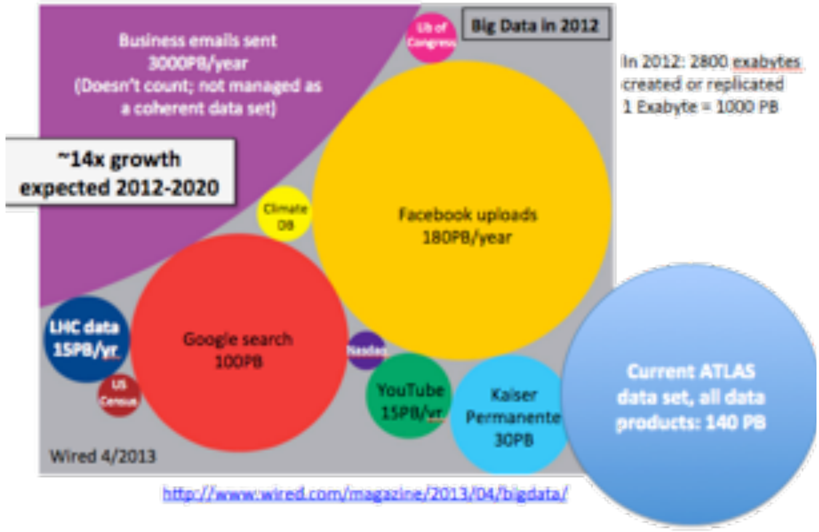
Google has 10-15 exabytes under management

Facebook 300PB

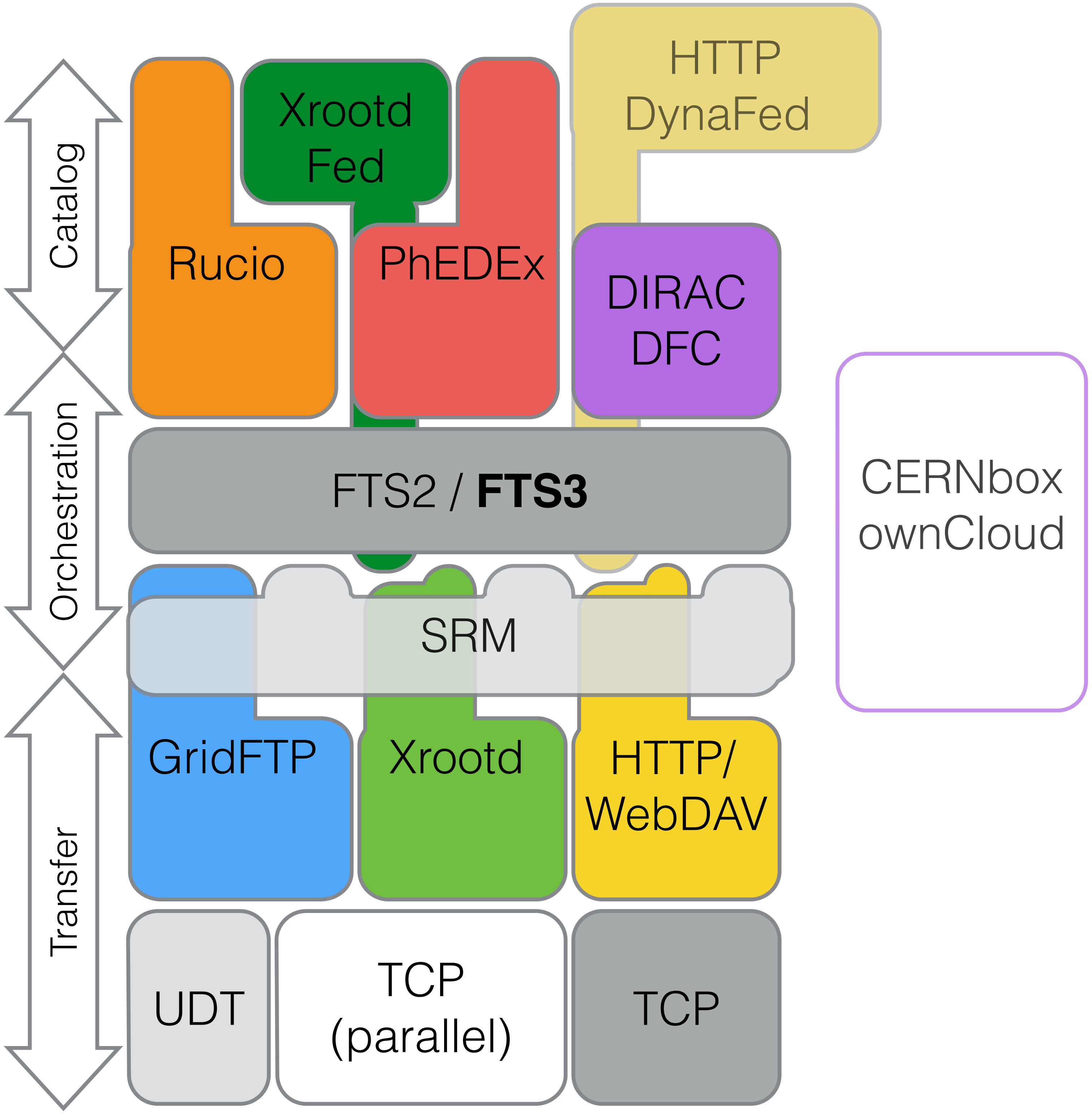
eBay collected and accessed the same amount of data as LHC Run1



Our data and processing problems are ~1% the size of the largest industry problems, but we still distribute more data and lead in the area of data management



WLCG Stack



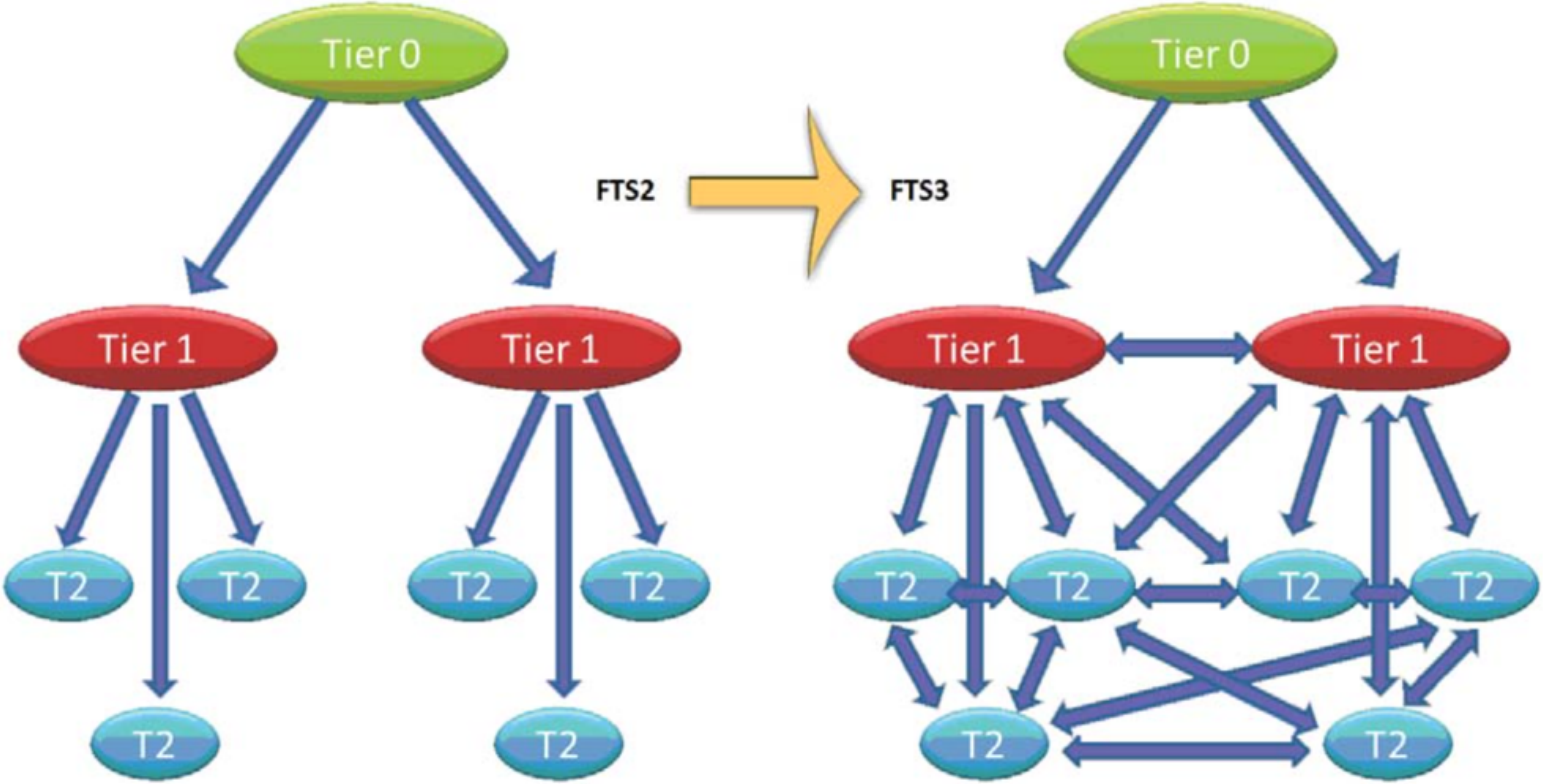
- Run 2
- (2013-)

Lessons

- Practical experiences trump theoretical models.
    - Trade-offs are not static.
    - Strict boundaries can hinder agility.
    - Abstraction layers can be enemies of performance.
    - Engage with common standards.

Lessons

- Trade-offs are not static.
  - In 2004, networks looked slow - emphasis on hierarchy, many copies of data close to compute.
  - By 2011, networks look fast - move to mesh models, remote data access over WAN
  - By 2016, talk of caching at compute, (too much data to have many static copies)


Tier 0
Tier 1
Tier 1
T2
T2
T2
T2
T2
T2
FTS2
FTS3
Tier 0
Tier 1
Tier 1
T2
T2
T2
T2
T2
T2

Lessons

- Strict boundaries can hinder agility.
  - Old hierarchical model made it hard to dynamically respond to "hot" datasets.
    - Unlike Netflix, can't predict this in advance!
    - "Dynamic data placement" - easier with meshy, cachey models.
  - Strict, complicated "space reservations" hard to manage (esp for ATLAS VO which made many different ones).
    - Simplify, consolidate spaces (also supports dynamism)

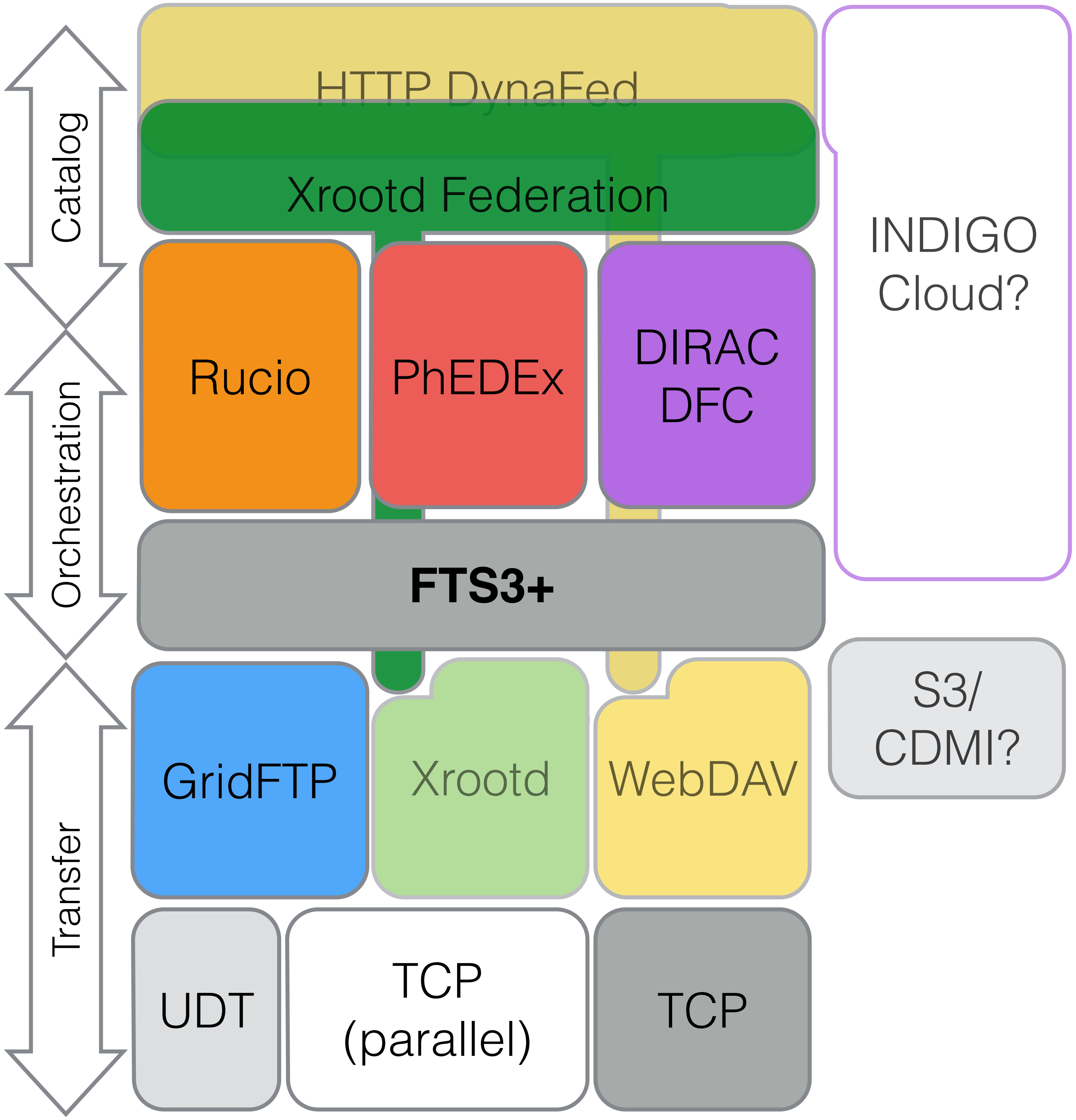
Lessons

- Abstraction layers can be enemies of performance.
    - File replica locations stored in "Logical File Catalogs"
    - Storage systems at sites "abstracted" by (asynchronous) "Storage Resource Manager" API
    - Multiple indirections cripple local access performance.
    - Move to smarter (algorithmic) file location, direct POSIX-like (or Object store) access locally.

Lessons

- Engage with common standards.
    - HEP uses "weird" protocols ["xrootd", X509 with "VOMS" extensions, SRM etc]
    - Leveraging commercial or shared resources is hard when you don't share a common language or tools.
    - Embrace common tools/standards: http/GridFTP, (S3/CDMI?), eduGAIN
        - (This also helps remove *abstractions* in our own layers)

Future WLCG Stack?



- Future (2016++)
- “Run 3+”