NEXUS: Big Data Analytics and Cloud Computing

2017 ESIP Federation Summer Meeting Workshop

Jet Propulsion Laboratory
California Institute of Technology

POC: thomas.huang@jpl.nasa.gov

IN009: Big Data Analytics

IN009:
Big Data Analytics

Submit an Abstract to this Session

Session ID#: 24747
Session Description:
Big Data pose great challenges for Earth and Space sciences. Cloud Computing emerged as a promising solution for supporting Big Data analytics in areas such as climate science, ocean science, atmospheric science, planetary science, and other geoscience domains for model simulation, data management, information mining, decision support, knowledge discovery and visualization. As a follow-on to the 2016 success at AGU, this session is to capture the latest on applying Cloud Computing for Big Data Analytical problems in all Earth and space domains. Topics include experiments, demonstration, studies, methods, solutions and solution discussion on:
Solutions for big data analytics
Big data management and mining
Application of open source technologies
Automated techniques for data analysis
Browser-based data analytics and visualization
Real time decision support
Contributions that fuse participatory social learning into the Geoscience R&D processes are also welcome.

Primary Convener:
Thomas Huang, NASA Jet Propulsion Laboratory, Pasadena, CA, United States
Conveners:
Chaowei Phil Yang, George Mason University Fairfax, Fairfax, VA, United States, **Tiffany C Vance**, NOAA Seattle, Seattle, WA, United States and **Brian D Wilson**, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, United States

Index Terms:

1914 Data mining [INFORMATICS]
1916 Data and information discovery [INFORMATICS]
1928 GIS science [INFORMATICS]
1932 High-performance computing [INFORMATICS]

AGU FALL MEETING

https://agu.confex.com/agu/fm17/preliminaryview.cgi/Session24747

NEXUS Software Architecture

Joe Jacob

Jet Propulsion Laboratory
California Institute of Technology

Agenda

- NEXUS Architecture
- Deployment Scenarios
- Hands-on Labs

System Architecture

ETL System – Tile, ingest and stage data

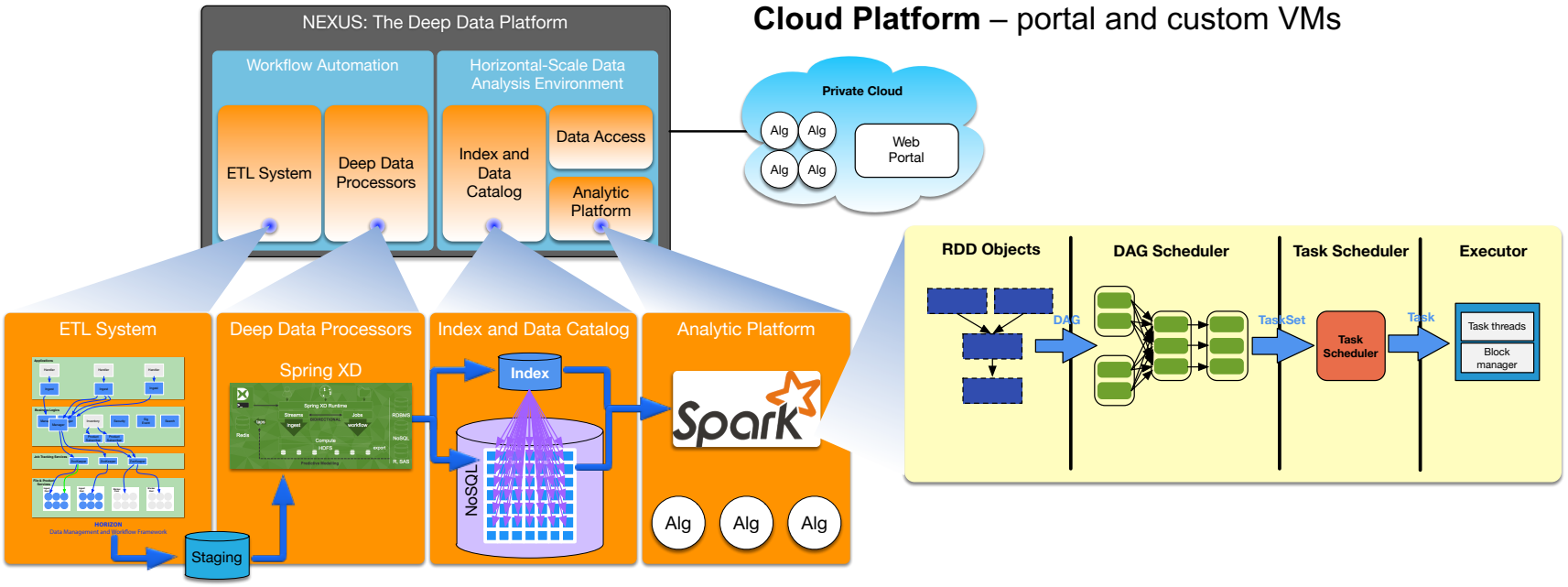
Deep Data Processors – metadata, statistics, and tiles

Index and Data Catalog – horizontal-scale geospatial search and tile retrieval

Analytic Platform – Spark-based domain-specific analytics

Data Access – tile and collection-based data access

Cloud Platform – portal and custom VMs

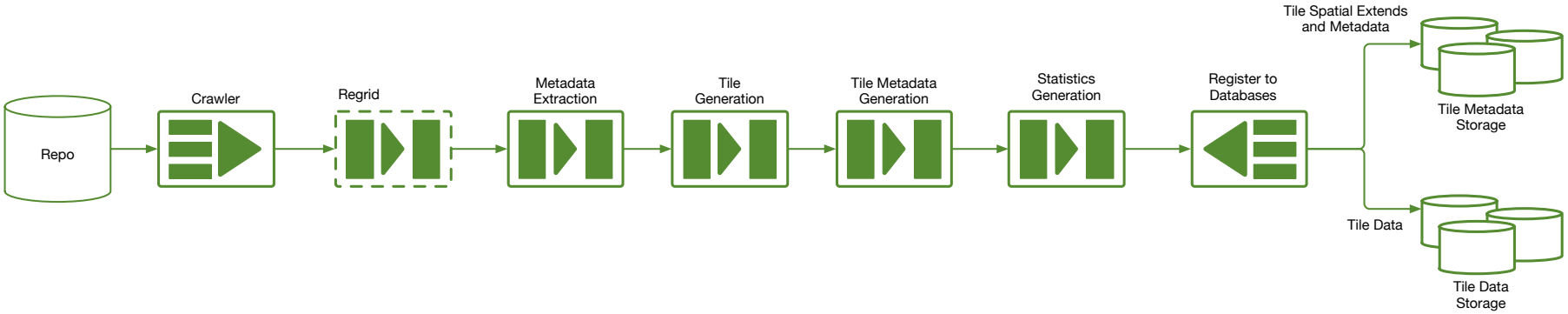


Ingestion and Tiling Cluster

What is a Tile?

- What is a Tile?
  - A collection of nd-arrays containing measurement data and its associated metadata
  - One granule becomes multiple tiles
  - Allows for fast spatial lookup of array data

- Horizontally Scalable Storage
  - Apache Solr Cloud
  - Apache Cassandra, ScyllaDB, Amazon S3

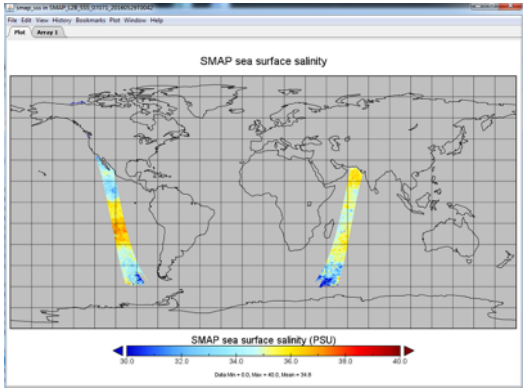


Dataset-specific Tiling Scheme

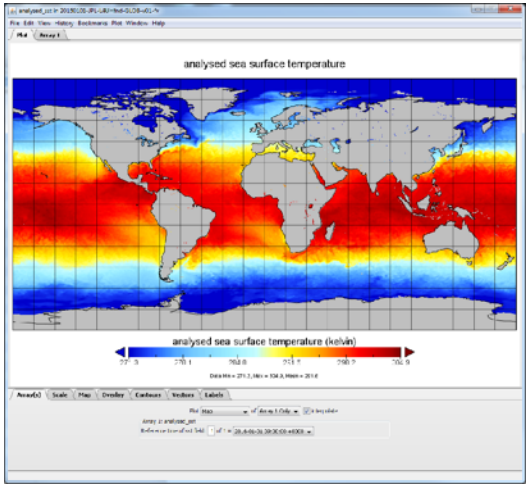
- Ingestion pipeline supports multiple tiling algorithms
  - L2 Swath Data
  - L3/L4 Gridded Data

L3/L4 Grid Tiling Algorithm:

$$c = Number\ of\ tiles\ desired$$
$$d = Number\ of\ dimensions$$
$$L_d = Length\ of\ dimension\ d$$
$$S_d = Step\ size\ for\ dimension\ d$$
$$S_d = \left\lfloor \frac{L_d}{\sqrt[d]{c}} + \frac{1}{2} \right\rfloor$$



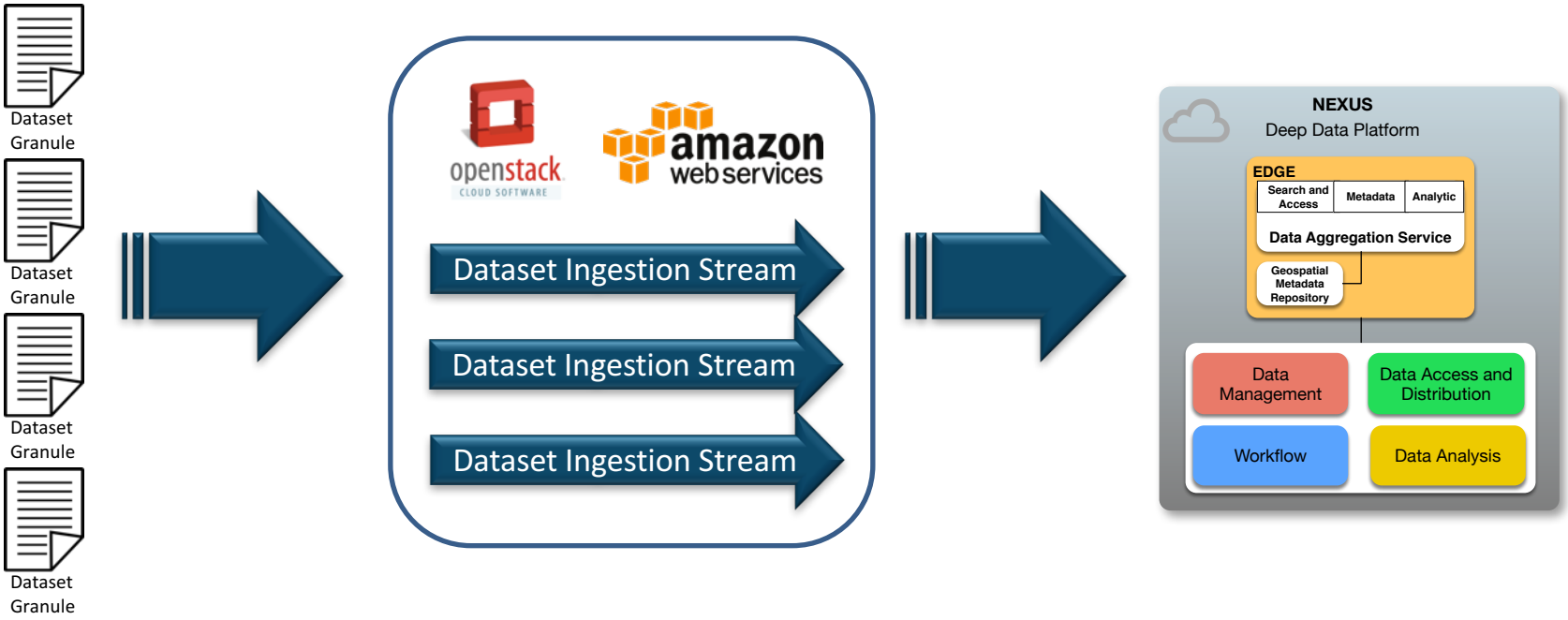
JPL/CAP L2B SMAP Sea Surface Salinity



MUR-JPL-L4-GLOB-v4.1 Analyzed Sea Surface Temperature

Multiple Streams

- Streams can run in parallel
- Individual stream modules can be scaled horizontally
- Streams deployable to the cloud



Pluggable Architecture

- Pluggable validation checks

```
def filter_empty_tiles(self, tile):
    # Only supply data if there is actual values in the tile
    if tile.data.size - numpy.count_nonzero(numpy.isnan(tile.data)) > 0:
        yield tile.data
    else:
        print "Discarding data %s from %s because it is empty" % (tile.section_spec, tile.granule)
```

- Data transformation

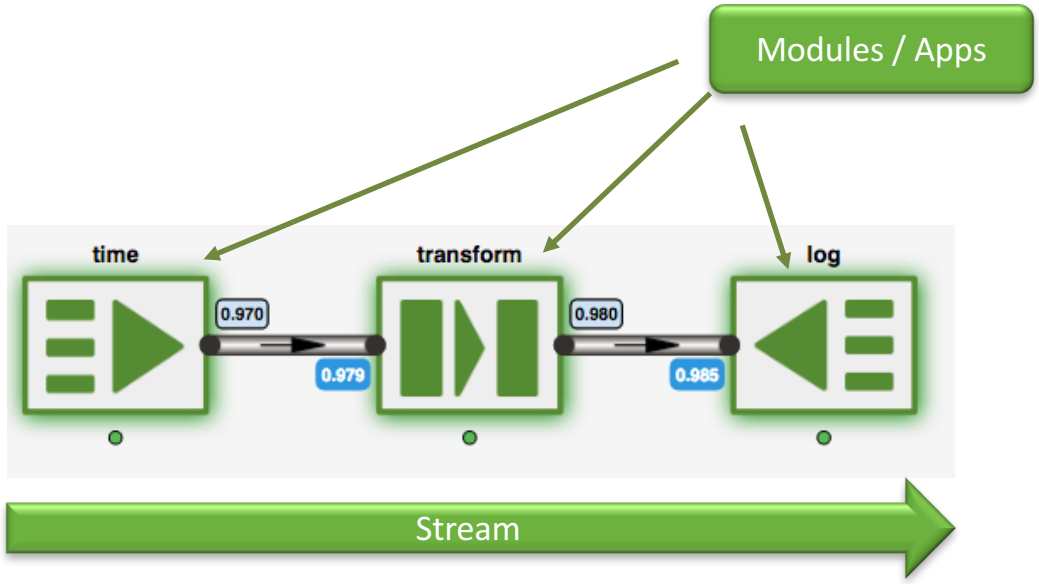
```
def transform(self, tile):

    tile.data.longitudes[longitudes > 180] -= 360

    yield tile.data
```

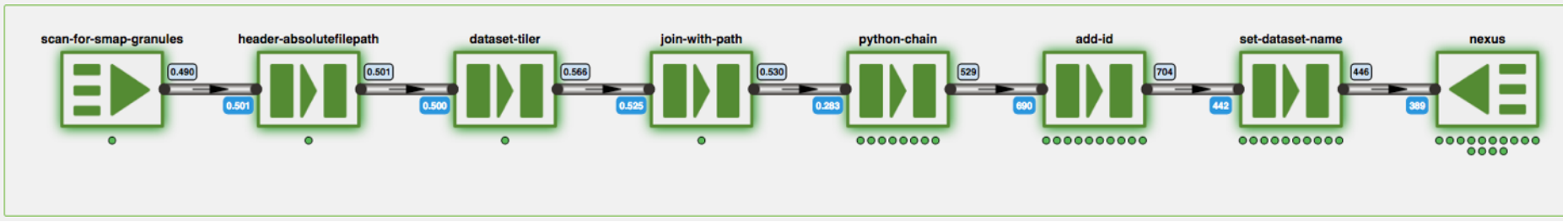
Using SpringXD

- Spring XD
  - http://projects.spring.io/spring-xd/
  - Current production release
  - Additional software components: Zookeeper, Kafka, Redis
- Spring Cloud Data Flow
  - http://cloud.spring.io/spring-cloud-dataflow/
  - Redesign of Spring XD



Ingestion in Summary

- Tested using different environments
  - Bare Metal NASA AIST-funded Deep Data Computing Environment (DDCE) at JPL
  - Mirantis OpenStack at JPL
  - NASA AIST Managed Cloud Environment (AMCE)
- Applications are connected to form ingestion streams
- Configurable to handle different datasets
- Scalable across compute resources
- Resilient to failure

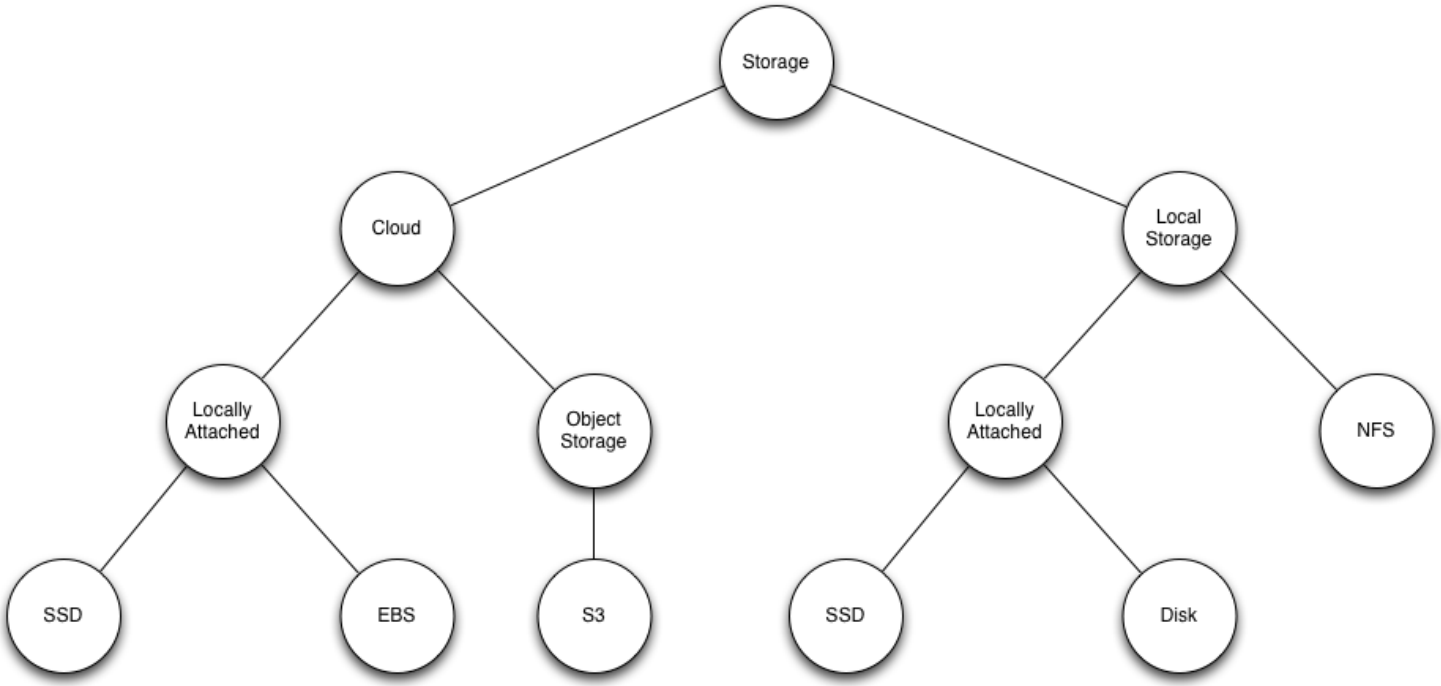


Stream for JPL/CAP L2B SMAP Sea Surface Salinity

Data Management Clusters

NEXUS Storage Options

- Tiles can be stored in NoSQL (e.g. Cassandra) or Object Store
- Cassandra supports SSD, locally attached storage, or NFS (not recommended)
- Storage selection dependents on performance requirement and cost
- NEXUS has an abstract storage architecture



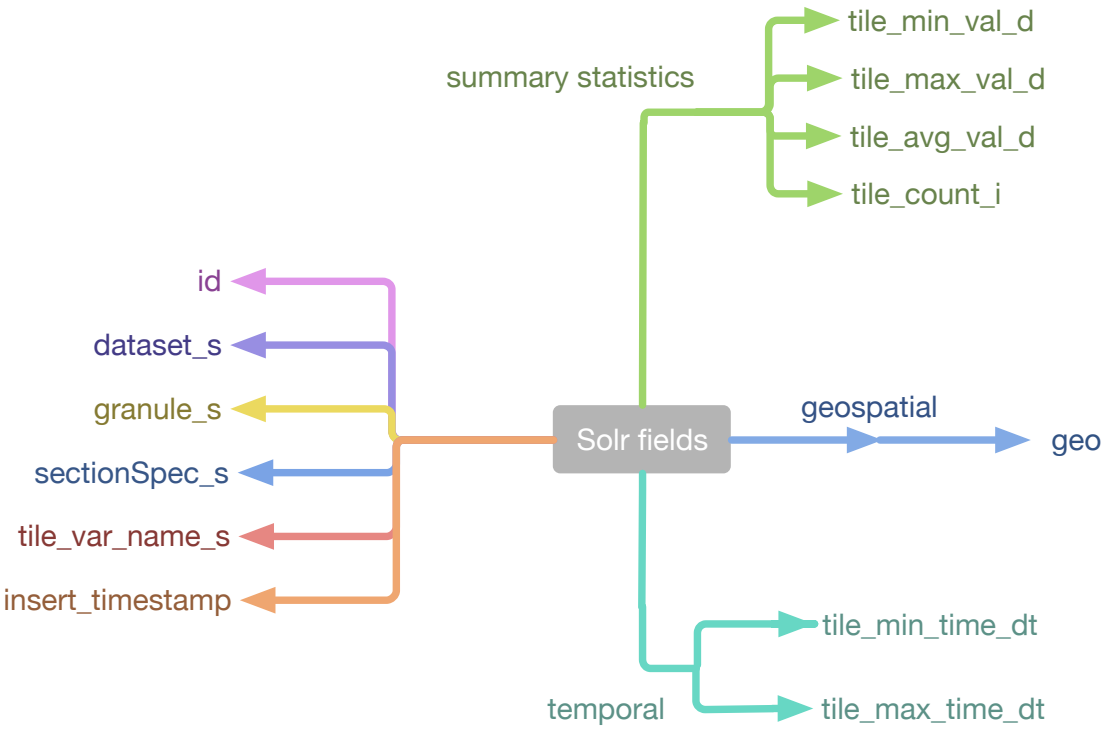
SolrCloud and Cassandra Cluster Architecture

- SolrCloud
  - Distributed search and indexing
  - Uses ZooKeeper to manage cluster state
  - No master node, queries and updates sent to any node
  - Collection, a complete logical index, gets divided into multiple shards
  - Configure to use `compositeId` document router with dataset ID as prefix for all document ID
    - All documents belonging to a dataset gets indexed on same shard
  - Can set up shard replicas for redundancy

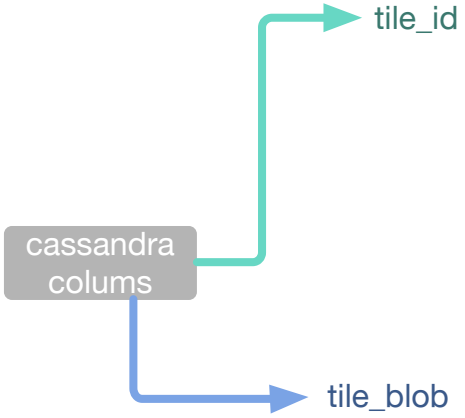
- Cassandra Cluster
  - Uses gossip, peer-to-peer communication protocol, to exchange cluster state information
  - Data is evenly distributed across all Cassandra nodes
  - Can set up replicas to ensure reliability and fault tolerance

Solr and Cassandra Schemas

- Purpose is to keep track of tiles and enable fast retrieval
- The same Solr server can be extended to support additional metadata



Solr Schema

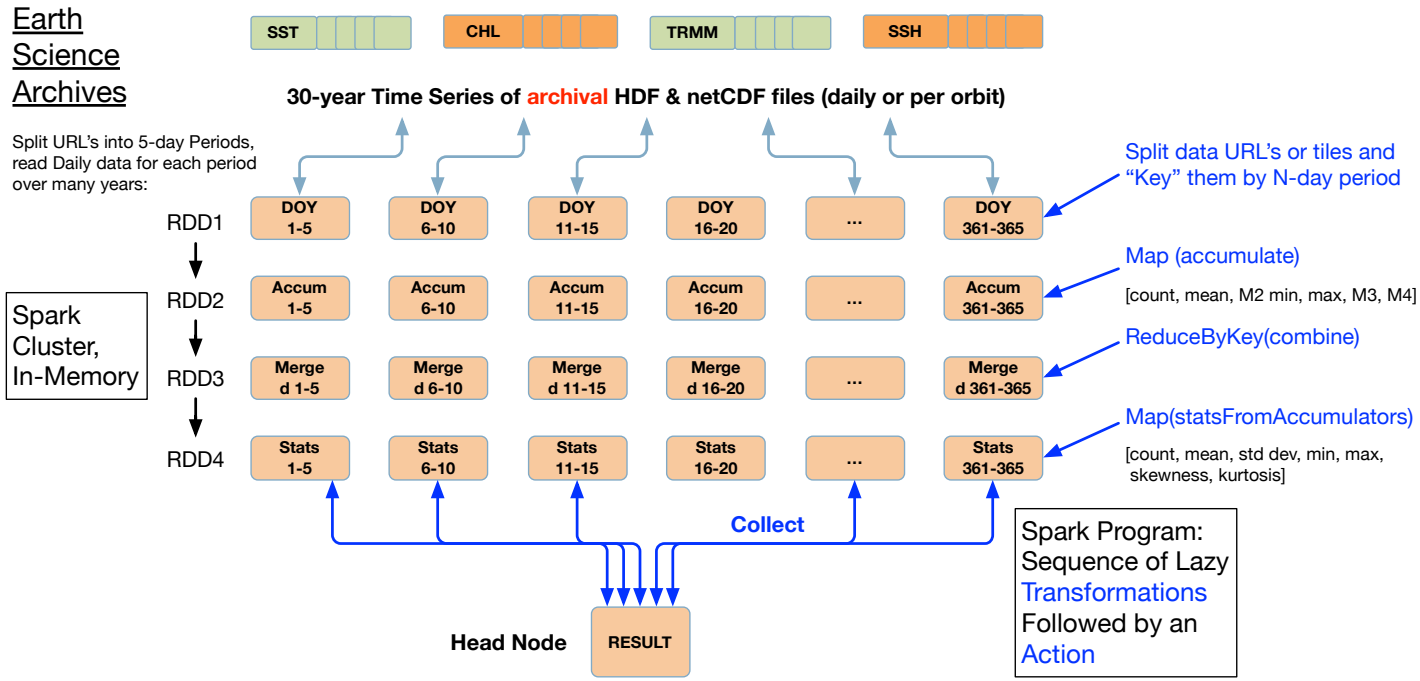


Cassandra Schema

Data Analytic Cluster

Architecture

- Analytics are collocated with Data Management cluster
  - Moving or copying science data costs more (time / money) than the computation itself.
  - Take advantage of data locality for I/O bound analytics.
    - Each processor works on its local data
    - Avoid shuffle operations.
  - Match data parallelism with tiling scheme – 1 data tile is 1 chunk of work
- Analytics driven by Apache Spark
  - In memory map-reduce style computations



- Parallelize over time or space (lat/lon)
- Well-supported: NEXUS can be (has been) deployed to laptop computer, cluster computer, private cloud (e.g., Our Mirantis Cloud/CloudWorks), or public cloud (e.g., AWS)

Tuning Performance

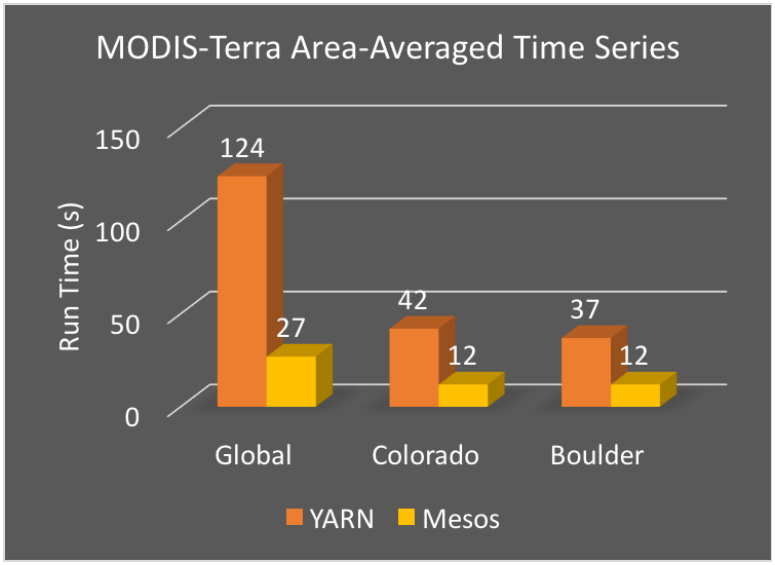
- **Control number of Spark executors and data partitions**
  - Executors are the worker processes that are instantiated when the Spark cluster is initialized and last for the life of the Spark application.
  - A data partition represents a chunk of work that is scheduled for processing on an executor.

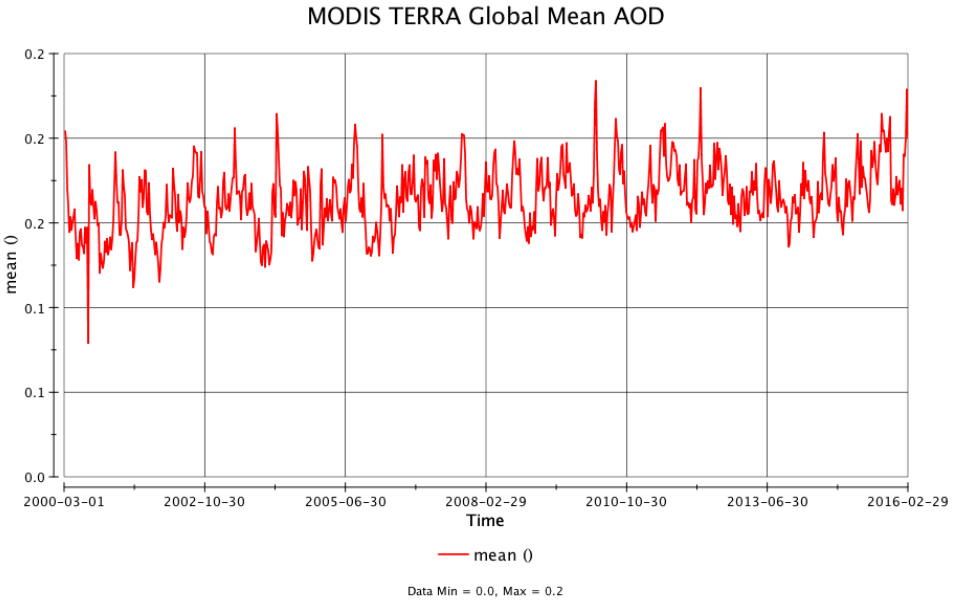
- **Spark performance depends on configuration**
  - Number of executors, E
  - Cores per executor
  - Memory per executor
  - Number of data partitions, P
  - Recommended that 2 <= P/E <= 4
  - > 200 configuration parameters in Spark 2.2.0 documentation
  - Nontrivial to squeeze best performance out of Spark for complex applications.

- **The data partitioning scheme used can impact performance**
  - Calculations on global data or very large subsets have best performance with a few large tiles.
  - Many small tiles are preferred for calculations on smaller subsets.

- **The Scheduler used can impact performance**
  - Spark uses YARN by default
  - Mesos is available as a separate package

Scheduler Comparison: YARN vs Mesos





- NEXUS run on 8-node cluster computer at JPL running Solr, Cassandra, Spark 2.0, with the YARN or Mesos scheduler, as indicated in the plot.
- Area-Averaged Time Series over the indicated spatial subset (Global, State, City) run with 16-way parallelism.
- Variable plotted: MODIS-Terra Aerosol Optical Depth (AOD) 550 nm dark target
- 5,789 daily data granules covering the globe at 1 deg resolution with date range: 3/1/2000 – 2/29/2016 (3 GB input data volume).
- In our experiments, using Mesos consistently yields a speedup of 2 to 4 times over YARN.

NEXUS Spark Analytics Algorithms

Included with NEXUS:

- Area-Averaged Time Series
  - Compute statistics (e.g., mean, minimum, maximum, standard deviation) for each time step within a user-specified spatiotemporal bounding box.
  - Optionally apply seasonal or low-pass filters.
  - Return result in ascending time order in JSON format.
- Time-Averaged Map
  - Compute a geospatial map that averages gridded measurements over time at each grid coordinate within a user-defined spatiotemporal bounding box.
- Correlation Map
  - Computes the correlation coefficient at each grid coordinate within a user-specified spatiotemporal bounding box for two identically gridded datasets.
  - Automatically aligns the time stamps for the two datasets being compared.
- Climatological Map
  - Similar to Time-Averaged Map, but only includes measurements in the time average that are within a user specified month.

Application Specific Extension: Anomaly Detection (OceanXtremes)

- Climatology
  - For each day-of-year (1-366) or month (1-12), computes a ”typical value” for each coordinate grid location.
  - The ”typical value” may be the result of either (1) a standard pixel mean with optional smoothing over time (e.g. 5-day average), (2) Gaussian interpolation [Armstrong and Vazquez-Cuervo, 2001], or Empirical Orthogonal Function (EOF).
- Daily Difference Average
  - Subtract a dataset from its climatology, then, for each time stamp, average the differences within a user-specified spatiotemporal bounding box.
  - Product can be used to search for anomalies compared to the historical norm.

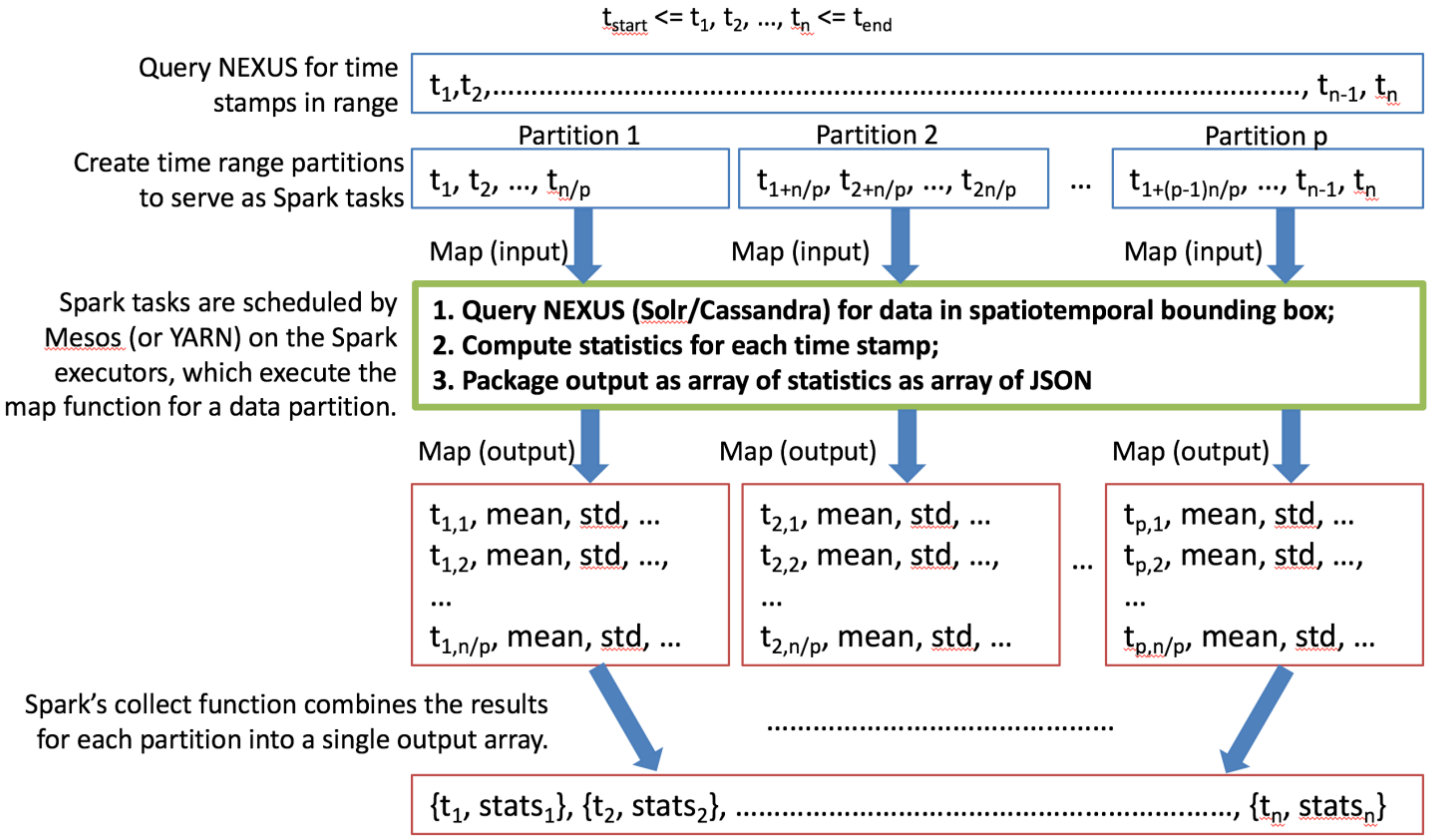
Application-Specific Extension: Distributed Oceanographic Match-up Service (DOMS)

- In Situ Match
  - Discover in situ measurements that correspond with a gridded satellite measurement.

How to implement new algorithm

- Cast your algorithm in Map-Reduce style
  - Map: Independent operations applied to the data elements; e.g., `map()`, `mapPartitions()`
  - Reduce: Combine individual results; e.g., `collect()`, `reduce()`, `reduceByKey()`, `foldByKey`, `combineByKey()`

- Example: Area-Averaged Time Series



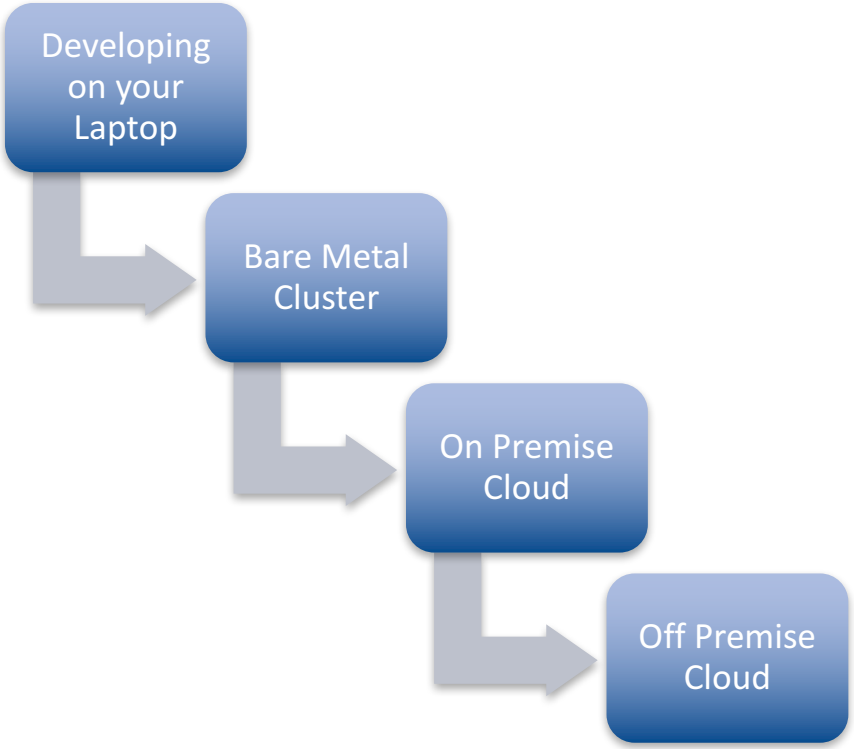
NEXUS Deployment Scenarios

Frank Greguska

Jet Propulsion Laboratory
California Institute of Technology

Overview

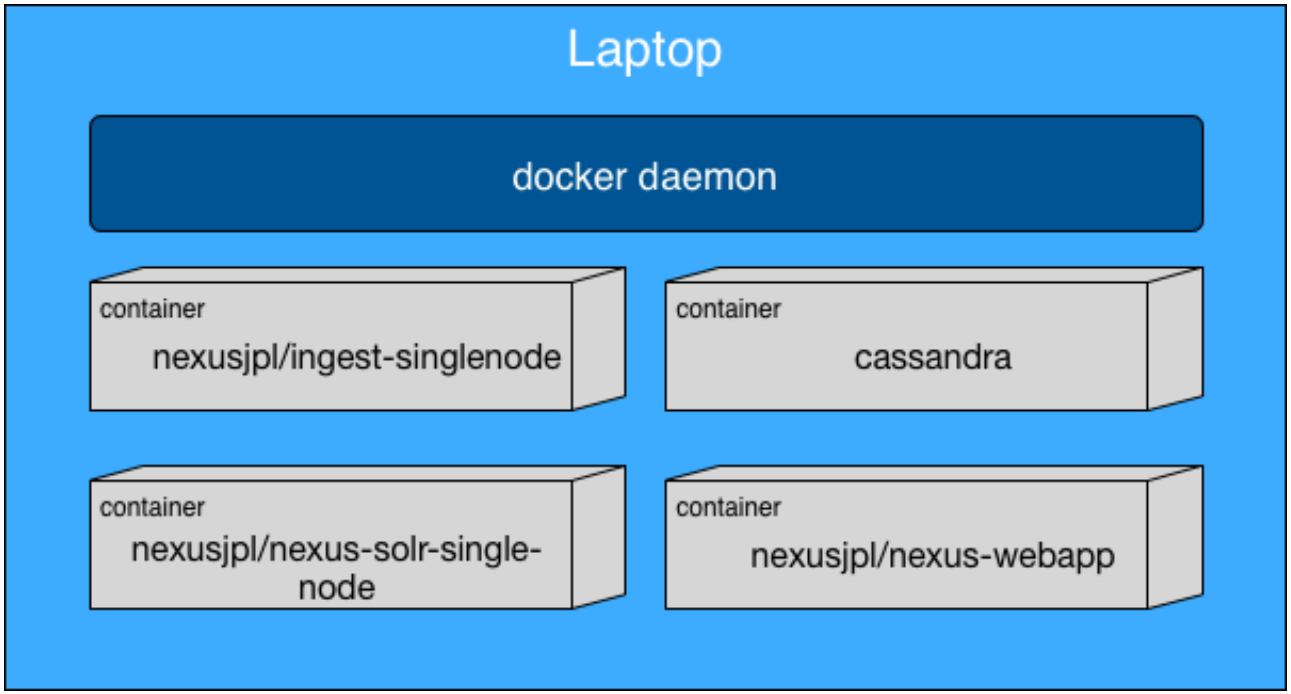
- Installing NEXUS from Local to Cluster to Cloud



- Other Installations

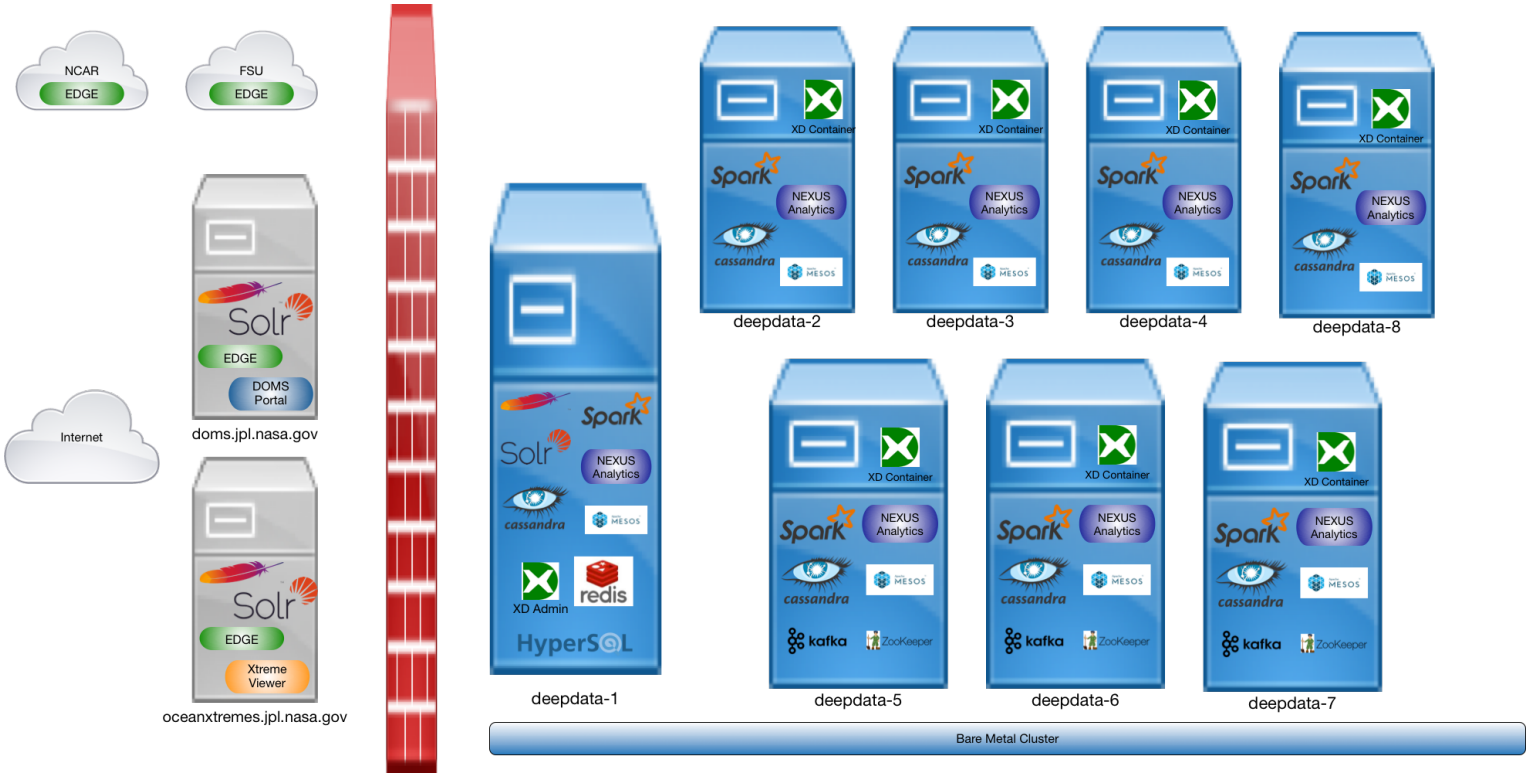
Local

- Running NEXUS on your laptop
  - Native with Vagrant
    - https://github.com/dataplumber/nexus#developer-installation
  - Docker
    - https://github.com/dataplumber/nexus/tree/master/docker
    - https://hub.docker.com/u/nexusjpl/dashboard/



Bare Metal

Bare Metal NASA AIST-funded Deep Data Computing Environment (DDCE) at JPL



Bare Metal

PROS

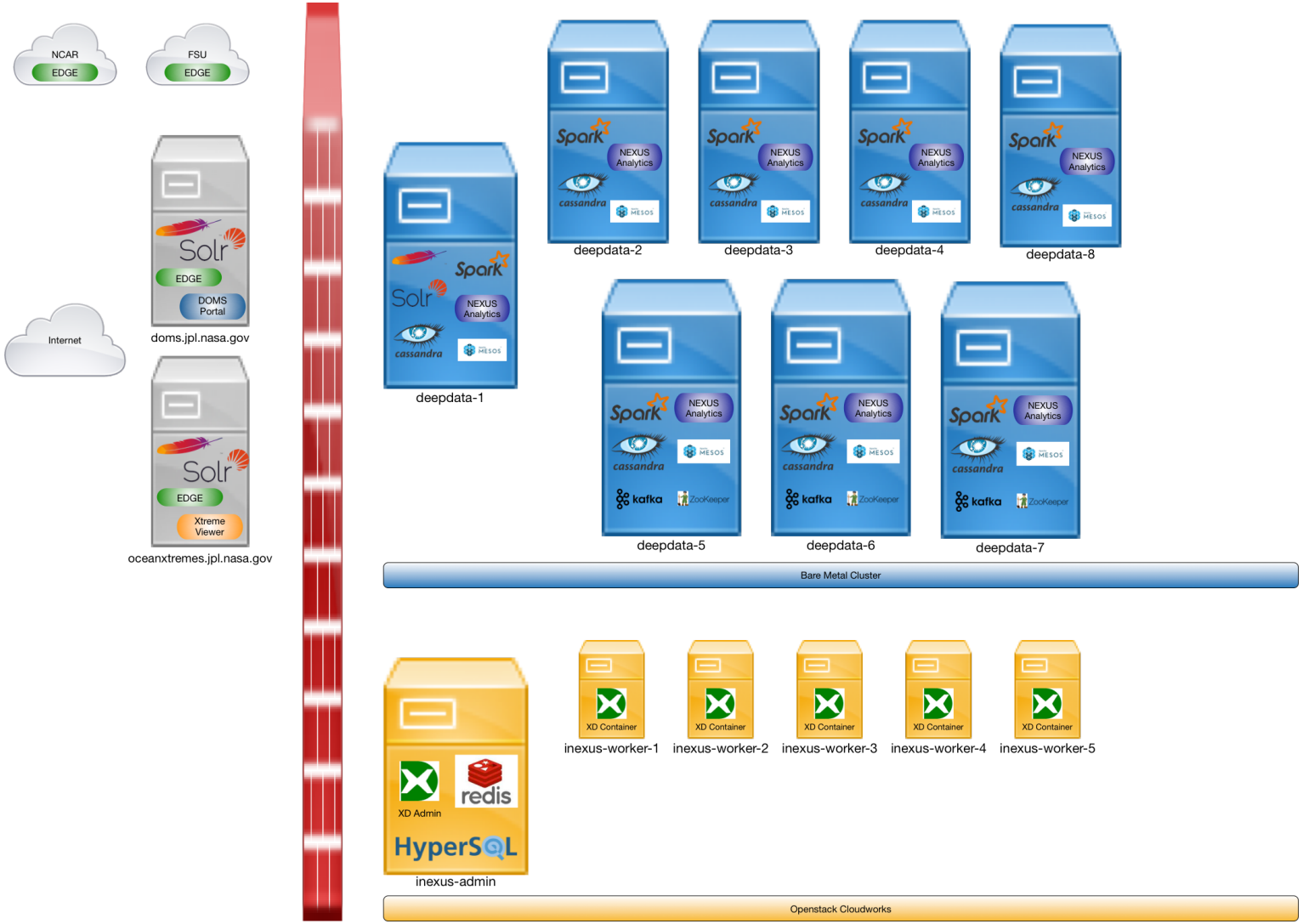
- Full control over operating system, software, and configuration
- No additional software overhead
  - Local disk access
  - Direct network adapter access
- Cost

CONS

- Management is difficult
  - Operating system patches
  - Custom startup scripts
  - Lots of SSH sessions
  - Adding new machines
- Clusters competing for resources
- Cost

On Premise Cloud

Ingestion Cluster moved to OpenStack



On-Premise Cloud

PROS

- Virtual - easier to add new machines
- Similar to bare metal installation

CONS

- Virtualization adds layer of abstraction – i.e. overhead
  - Kafka performance issues
- Similar to bare metal installation

Amazon Web Services (AWS) Elastic Compute Cloud (EC2)

- AIST Managed Cloud Environment (AMCE)
  - Fully Docker-ized Deployment

Amazon Web Services (AWS) Elastic Compute Cloud (EC2)

PROS

- Very easy installation
  - Write Dockerfile once, deploy anywhere Docker can run
  - Host machines only need to be able to run Docker
- Easy to add new containers
- Flexible deployment architecture
  - Choose the size that is right for you
- Cost

CONS

- Container Orchestration is hard
  - Docker Swarm and Docker Stack not ready for primetime
  - Don't kill the swarm manager
- Debugging is harder
  - Especially when using Docker defined networking
- Additional overhead between code and infrastructure
  - In practice, not significant with our workload
- Cost

Other Deployments

- Amazon Web Services (AWS) Elastic Compute Cloud (EC2)
  - Used beefy machines
    - 6 x i2.4xlarge
      - Memory: 122 GB, vCPUs: 16, 4 * 800 GB SSD per instance
  - Compared Cassandra vs. ScyllaDB
  - Similar to bare metal installation

- Sea Level Change Portal
  - Bare metal installation at JPL
  - Small cluster due to nature of data
    - 1 Solr instance
    - 1 Cassandra node
    - No Spark/Mesos

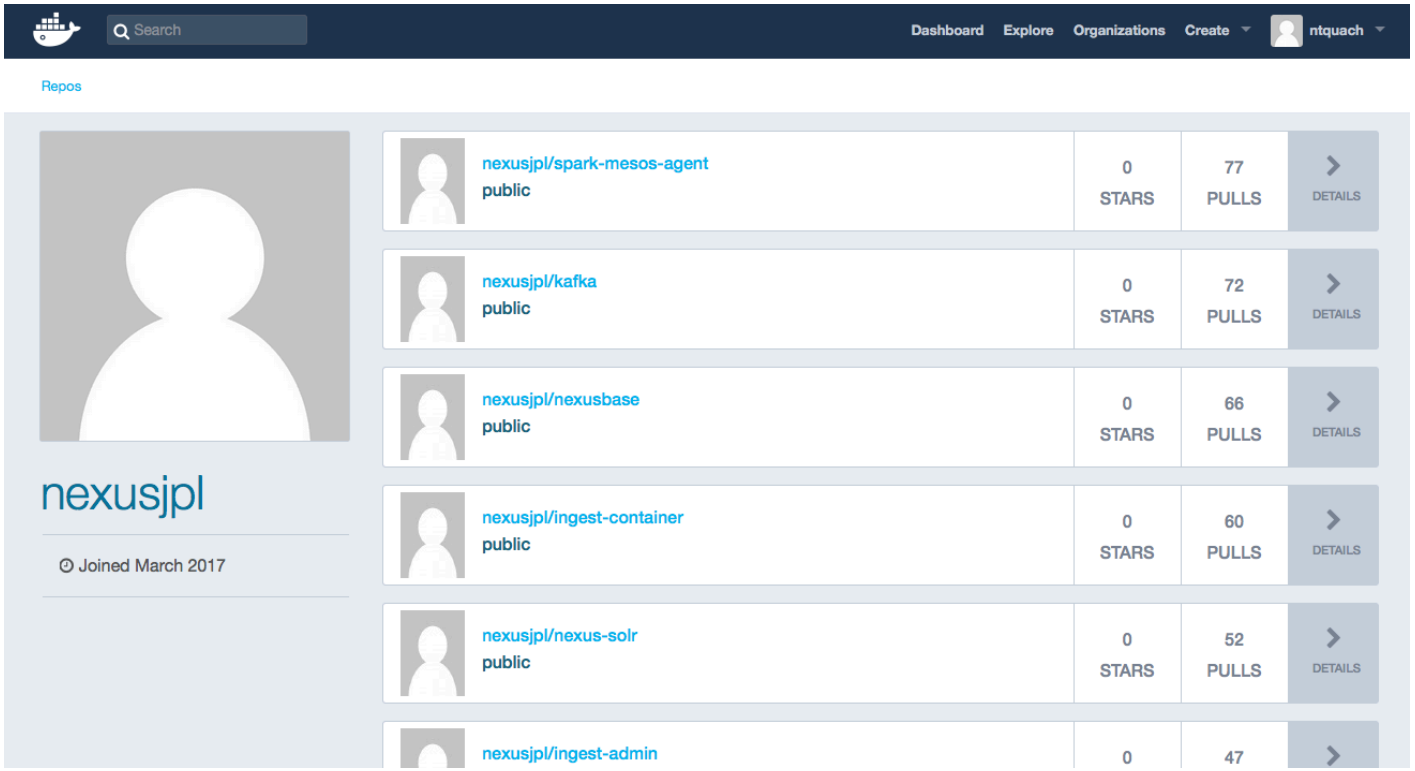
- This Workshop!
  - Single EC2 instance per student (group)
  - ”Mimic” a full cluster deployment

Docker Deployment

Docker Deployment

- What is Docker?
  - Open-source lightweight software container platform consisting of Docker Engine and Docker Registry
  - Pack, ship and run any application as a lightweight container that can run anywhere
  - Container bundles only application and libraries/binaries required by application

- Images for Nexus components pushed to Docker Hub



Docker Compose

- Tool for defining and running multi-container applications
- Single command to start multiple containers
- Applications are defined in YAML file where options passed to `docker run` can be specified
- 3 Compose files for this workshop
  - Infrastructure Compose File
    - Solr
    - Cassandra
    - ZooKeeper
  - Ingestion Compose File
    - MySQL
    - Redis
    - Kafka
    - Spring XD Admin
    - Spring XD Container
  - Analytics Compose File
    - Mesos Master
    - Mesos Agent

Hands-On Labs

Frank Greguska, Joseph Jacob, Nga Quach

Jet Propulsion Laboratory
California Institute of Technology