
BNL, SLAC and UChicago Shared Analysis Facilities

Fengping Hu on behalf of the UChicago Maniac Lab Team, Ofer Rind, Doug Benjamin, Wei Yang

US ATLAS Computing Facilities Face-to-Face at SLAC
2022.11.30

ATLAS
EXPERIMENT

The ATLAS Experiment logo, featuring a stylized 'M' shape composed of small blue squares, positioned at the bottom right of the slide.

Outline

- Overview of the US AFs
- Hardware at each site
- Common Capabilities
 - Identity Management
 - Batch Access
 - Jupyter Access
- Unique Features



Overview of US Analysis Facilities

US ATLAS has three shared Tier 3 analysis facilities providing software & computing

- Resources that fill gaps between grid jobs and interactive analysis on local computers
- Interactive ssh login, local batch, non-grid storage, LOCALGROUPDISK, PanDA
- GPU resources available

Launched Oct 2021



BNL Facility

~2000 cores, part of a larger shared pool,
opportunistic access up to 40k cores

User quota: 500GB GPFS plus 5TB dCache

~200 users



SLAC Facility

~1200 cores, part of larger shared pool,
opportunistic access up to 15k cores

User quota: 100GB home, 2-10TB for data

~100 users



UChicago Facility

~3000 cores, co-located with MWT2,
opportunistic access up to 16k cores

User quota: 100GB home, 10TB for data

~170 users

Hardware specifications @SLAC

SLAC operates 3 computing environment:

- LSF/AFS based cluster and farm (to be decommissioned by Aug 2023)
 - Lots of slots. Some users still heavily using them
- Interim (SLAC Data Facility, or SDF) SLURM Cluster and Lustre storage
 - US ATLAS owns:
 - 512 AMD EYPC 7702 cores in 4 nodes. 512GB RAM on each node.
 - 64 AMD EYPC 7542 cores + 4 Nvidia A100 GPU, 1TB RAM on one node
 - ~830TB Lustre space
- SLAC Shared Scientific Data Facility (S3DF) SLURM Cluster and Weka storage
 - Available to general usage by Jan 2023
 - Major experiments such as LCLS and Vera C. Rubin are adding hardware here.
 - All US ATLAS resource in SDF will be moved to S3DF.



Hardware specifications @BNL

BNL has several storage systems and technologies available to the users

- NFS Posix home area with small quota (100 GB) on storage appliance
- Each user has dCache R/W disk space
- ATLAS users can be granted access to GPFS storage
- New Lustre storage (2 PB) available to all SDCC ATLAS users
- CERN EOS R/W access via fuse mount and CERN kerberos credentials

Two different computer clusters – Shared HTCondor pool and Institutional Cluster

- Shared pool:
 - 56k cores. ATLAS users have 2.2k core quota but able to burst much higher
 - 2 nodes dual NVidia A100–80 GPU's – MIG used to subdivide GPU's
- Institutional Cluster (6 dedicated nodes with dual P100 GPU's)



Hardware specifications @UChicago (excluding GPU)

Node Type	Number of Nodes	Processor Per Node	Cores Per Node	Memory Per Node	Storage Per Node	Notes
Hyper-converged storage & CPU	19	Two AMD EPYC 7402 CPUs at 2.8 GHz	48C/96T	512 GB DDR4 SDRAM	Two 960 GB SSDs, four 2TB NVMe and twelve 16 TB spinning disks.	3 nodes provided by, and for, UChicago ATLAS group
Fast Compute & storage	16	Two Intel(R) Xeon(R) Gold 6348 CPU at 2.60 GHz	56C/112T	384 GB DDR4 SDRAM	Ten 3.2 TB NVMe	Provided by, and for, IRIS-HEP SSL
Interactive Nodes	8	Two AMD EPYC 7402 CPUs at 2.8 GHz	48C/96T	256 GB DDR4 SDRAM	Two 960 GB SSDs	2 nodes provided by, and for, UChicago ATLAS group
Xcache Node	1	Two Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz	24C/48T	192 GB DDR4 SDRAM	Twenty Four 1.5 TB NVMe	



Hardware specifications @UChicago GPU Nodes

Node Type	Number of Nodes	Processor Per Node	Cores Per Node	Memory Per Node	GPUs Per Node (Mem)	Storage Per Node	Notes
GPU A	2	Two AMD EPYC 7543 32-Core Processor	64C/128 T	512 GB DDR4 SDRAM	Four NVIDIA A100 (40G)	One 1.5TB NVMe	MIG(4*7*5G) configured on one of the two nodes
GPU B	1	Two Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz	24C/48T	96 GB DDR4 SDRAM	Four Tesla V100 (16G)	Three 220GB SSD	
GPU C	3	Two Intel(R) Xeon(R) Gold 6146 CPU @ 3.20GHz	24C/48T	192 GB DDR4 SDRAM	Eight NVIDIA GeForce RTX 2080 Ti (12G)	Six 450GB SSD	
GPU D	1	Two Intel(R) Xeon(R) CPU E5-2687W v4 @ 3.00GHz	24C	128 GB DDR4 SDRAM	Eight NVIDIA GeForce GTX 1080 Ti (12G)	Six 450GB SSD	



Identity management at BNL/SLAC

- SLAC

- Porting NERSC IRIS to SLAC for account management
- Experimenting Dex IdP for token based SSH access
- Progress are slow due to effort on S3DF

- BNL

- Interactive access requires an SDCC account
- SDCC federated account can be created via CERN/SLAC/FNAL MFA login with a manual account verification procedure (no ssh access)
 - Documented in US ATLAS Tier 3 documentation ([Link](#))



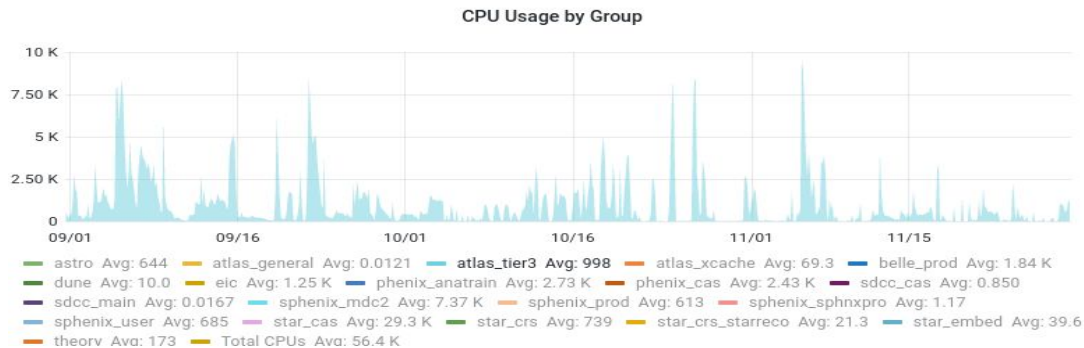
Account Management @UChicago

- In-house user management system (leveraging [CI Connect](#) component)
 - Users can sign up for SSH or Notebook access via globus oauth(CILogon)
- Openid connect(dex running on kubernetes)
 - Power users can access K8S directly via [Dex IdP](#) configured to utilize CILogon
- CERN IAM for atlas(oauth)
 - Coffea Casa users utilize the ATLAS IdP at CERN



Batch Access at BNL

- Users with SDCC accounts can access HTCondor batch system in the Shared Pool via the command line on the spar* nodes.
- Users with SDCC accounts wanting to run jobs on the GPU machines in the Institutional cluster can access via the command line on icsubmit* nodes.
- All users with Federated-account or SDCC account can use the jupyter-lab interface to launch jobs from the Terminal or via Dask (Shared Pool).
- All ATLAS users can direct PanDA jobs to an opportunistic PanDA queue

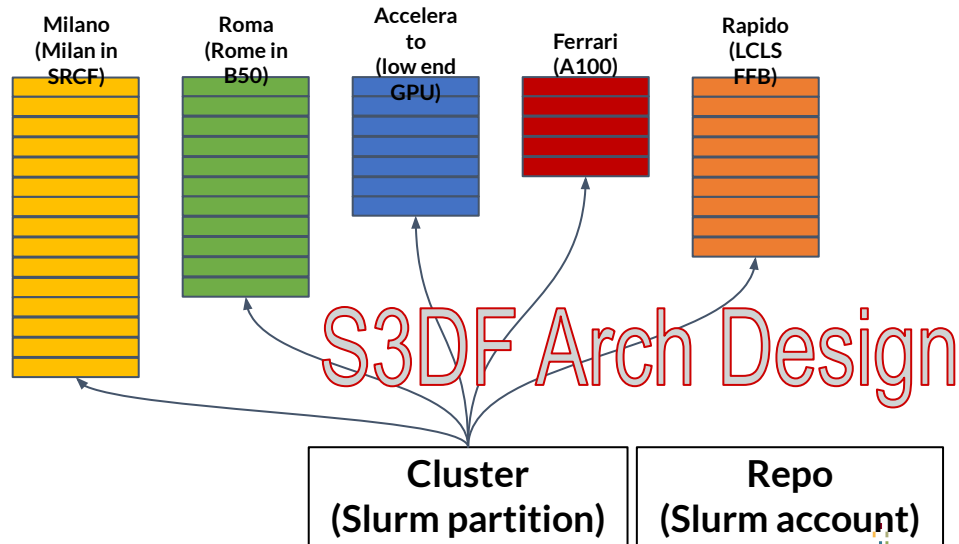
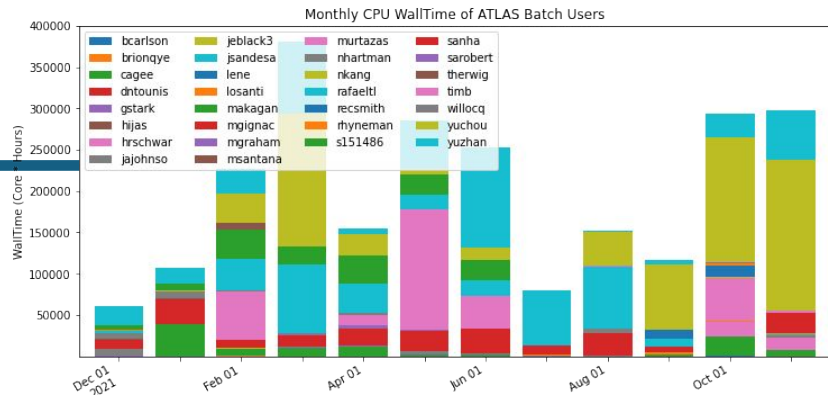


ATLAS AF batch activity for last 90 days peaks to 10k cores can be seen. Usage very spiky – much higher than default quota – 2.2k cores.

Batch Access at SLAC

SLAC has three computing facilities

- ATLAS users currently use the two older ones
 - one uses LSF, the other uses Slurm
- Will soon migrate all users to the latest one (S3DF)
 - Slurm



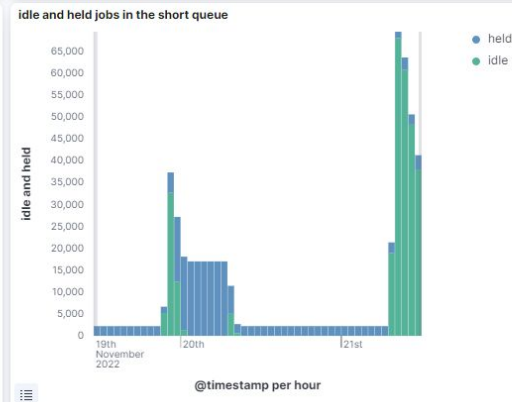
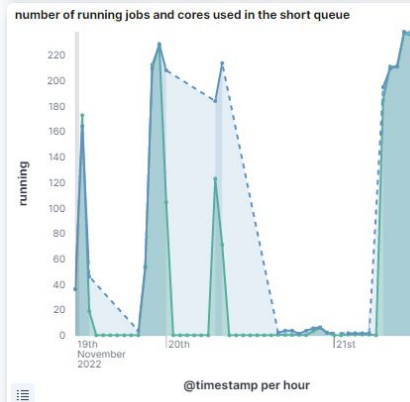
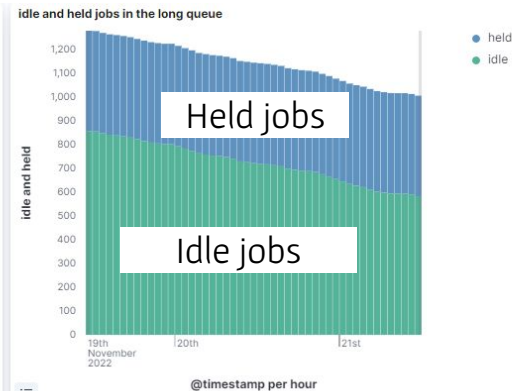
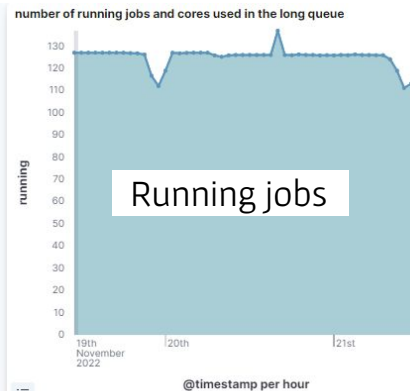
Batch access @UChicago – a synergy of cloud native Kubernetes and traditional batch system HTCondor

- Kubernetes is for containerized workloads and it minimizes the time and effort required to provision and manage infrastructure resources, but it has limited support for batch workloads
- HTCondor as deployments in Kubernetes
- Resource allocation at two layers
 - Kubernetes allocates resources to pods– dynamically auto scale with Horizontal Pod Autoscaling (HPA) to match demands
 - htcondor job claim resources from the htcondor pod
- Prometheus monitoring of real time resource usage
- Custom monitoring scripts and Elastic Search / kibana for HTCondor queue status monitoring



Batch Access @UChicago

- Short queue
 - Shorter walltime limit, quick turnaround
- Long queue
- MWT2 queue
 - Potential for handling bursty interactive load (Dask support in the works)



Jupyter Access @SLAC

Home / My Interactive Sessions / Jupyter

Interactive Apps

Desktops

✳ Cryosparc Desktop

🖥 Desktop

Services

Jupyter

Jupyter version: bc52c1d

This [app](#) will launch a customizable [Jupyter](#) server on our cluster and automatically present its interface on this webpage. You are free to create your own instances in [Conda/Singularity](#) etc on our clusters.

Jupyter Instance

Which Jupyter image to run

Commands to initiate Jupyter

```
ls /cvmfs/oasis.opensciencegrid.org /cvmfs/atlas.cern.ch
/cvmfs/atlas-condb.cern.ch /cvmfs/atlas-nightlies.cern.ch
```

Use JupyterLab instead of Jupyter Notebook?

JupyterLab is the next generation of Jupyter, and is completely compatible with existing [Jupyter Notebooks](#). Note this requires [JupyterLab](#) to be installed.

Disable JupyterLab extensions (Run with --core-mode)

Partition

Slurm [Partition](#) to launch Jupyter job on

Number of hours

Number of hours for which the Jupyter instance will run for

Number of CPU cores

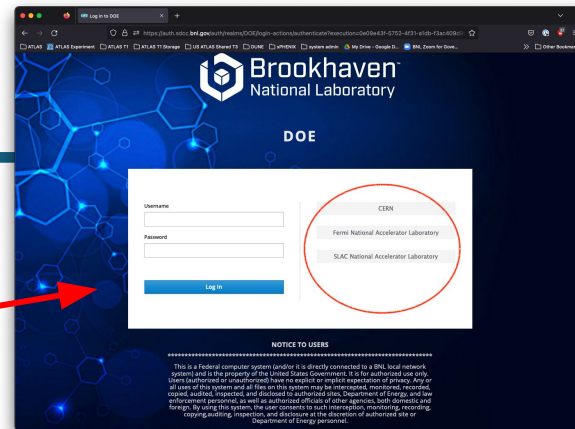
Jupyter web portal based on Open OnDemand

- A pull down menu of Jupyter environments in Singularity or Conda
- Or “bring-your-own-jupyter”
- Jupyter instance run as Slurm batch job
- Use Slurm account to choose between US ATLAS purchased resources or shared/opportunistic resources



Jupyter Access @BNL

- Federated ID Jupyter Hub at SDCC
 - Allows ATLAS users to use their CERN/FNAL/SLAC credentials as well as local credentials
- Users can use their own custom container



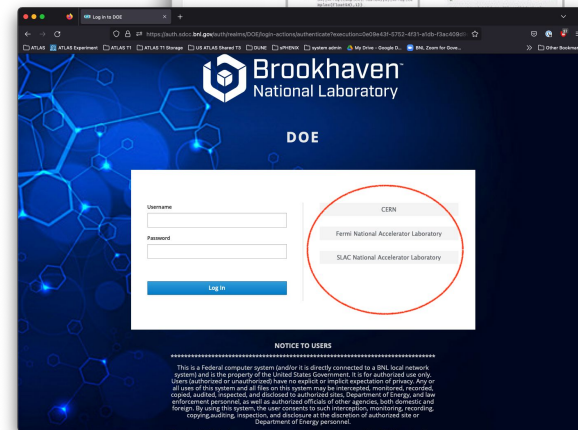
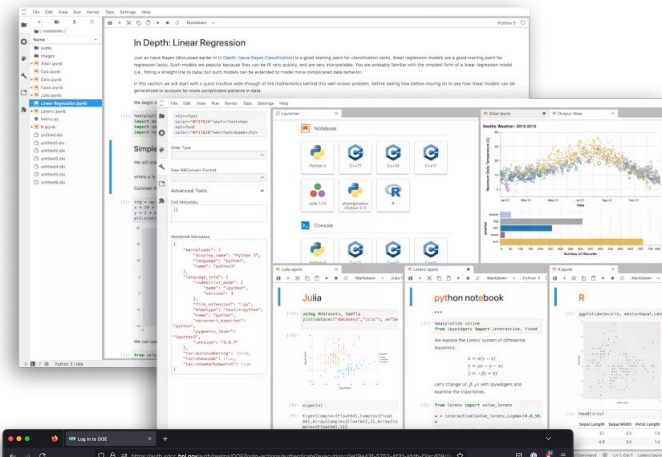
HTC cluster

HPC cluster

Evolution of User Analysis Tools

US ATLAS, NP, Belle II, NSLS II...

- Pythonic Big Data tools being used increasingly at Data centers
 - *JupyterLab allows users to access compute resources from within a web browser, instead of via traditional ssh command line interface (CLI)*
- Federated ID Jupyter Hub at SDCC
 - *Allows ATLAS users to use their CERN/FNAL/SLAC credentials as well as local credentials*
- Our users can access storage and compute farm through this mechanism.
 - *Leverage tools developed and maintained by a larger community outside of HEP*



Jupyter Access @UChicago

Configure a Jupyter notebook ([instructions](#))

Notebook name

CPU cores

Memory (GB)

GPU instances

GPU memory (MB)

Duration (hours)

Image

4864
11019
11178
16160
40536

Configure a Jupyter notebook ([instructions](#))

Notebook name

CPU cores

Memory (GB)

GPU instances

GPU memory (MB)

Duration (hours)

Image

ml-platform:latest
ml-platform:conda
ml-platform:julia
ml-platform:lava
ml-platform:centos7-experimental

Unique features @SLAC

- A large pool of GPU to support opportunistic usage
 - 80x Nvidia A100 (including 4 owned by US ATLAS)
 - 350+ older Nvidia GPU models
- Likely will have a large pool of K8s
 - Vera C. Rubin is k8s on steroids
- Weka Posix storage and large CEPH deployment
 - CEPH is both a Weka posix cold data tier, and
 - A standalone CEPH storage



Unique features @BNL

- Large pool of CPU nodes available to users in shared pool
 - ATLAS users can burst up to 10k cores
- Access to 38.1 PB ATLAS disk storage
- OKD cluster for new services
 - Reana instance
 - ServiceX instance (ATLAS XAOD)
- CVMFS Stratum-0 for publishing software

Unique Features @UChicago – Coffea Casa

- Deployed via Fluxcd
- Docker images served from [OSG Harbor](#)
- User authentication via Indico IAM for Atlas
- Dask scale out to htcondor in AF
- Dask scale out to MWT2 to handle bursty interactive workload(in the works)
- Supports multiple data access methods(servicex, local filesystem(/data) etc)



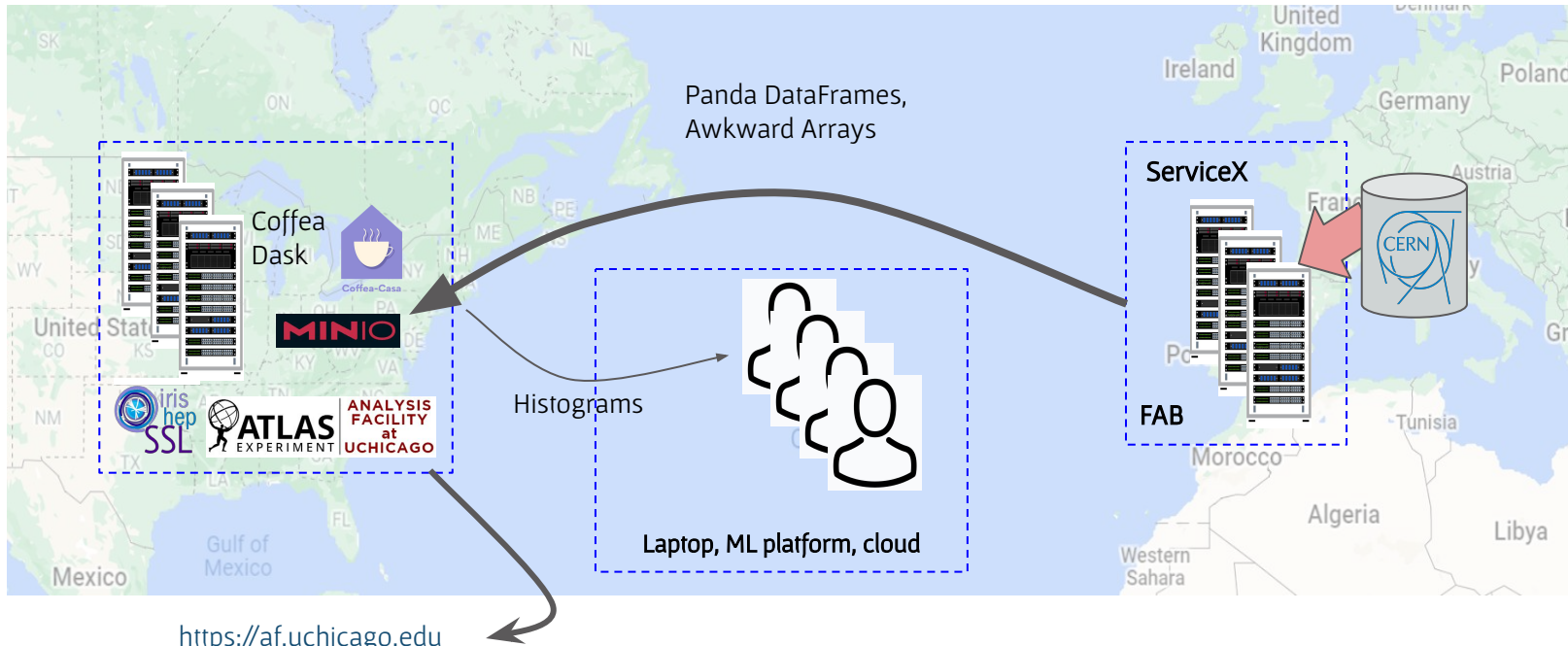
The screenshot shows a web browser window with the URL `coffea.af.uchicago.edu/hub/login`. The page features a dark red header with the text "Coffea Casa Service for US ATLAS". Below the header is the ATLAS Experiment logo, which includes a stylized figure holding a globe and the text "ATLAS EXPERIMENT". To the right of the logo is the text "ANALYSIS FACILITY at UCHICAGO". The main content area is white and contains the following text: "Authorized ATLAS Users Only!", "To login into the Coffea-Casa Analysis Facility, you will need to get a ATLAS OAuth token.", and "To get a token you need to a) be member of ATLAS and b) register with the OAuth service at: [atlas-Auth.web.cern.ch](#)". Below this is a section titled "Useful Links" with two links: "Coffea-Casa Support Page" and "Coffea-Casa Docs". There is also a "News" section with the text "Watch here for announcements!". At the bottom of the page is a dark red button with the text "Authorized ATLAS Users Only: Sign in with ATLAS SSO".

Unique Features @UChicago – ServiceX

- 2 Instances are deployed at AF
 - [ATLAS xAOD](#)
 - [ATLAS uproot](#)
- Endpoints are publicly accessible
- Authentication via Globus Auth. Account approval via a Slack channel `#servicex-signups`

R&D Activities @UChicago – ServiceX over FAB

Working with **FAB** (FABRIC Across Borders) to demonstrate ServiceX deployment at CERN, delivery of analysis objects to analysis facilities in the U.S.



Systems integration effort highlights @UChicago

- HTCondor on Kubernetes
 - Pioneered in running a production HTCondor system on Kubernetes
 - Integrated and adapted HPA(Brian's)
- coffea-casa
 - Worked closely with coffea-casa team to support Analysis Grand Challenge
 - Integrated coffea-casa with local account management system: local file system access, fair share.
 - Improved coffea-casa HTCondor integration: created and maintain a HTCondor chart – image prepulling, ephemeral-port-reservation fixes etc.