Analysis Grand Challenge

Alexander Held (UW-Madison)
Oksana Shadura (UNL)

IRIS-HEP / Ops Program Analysis Grand Challenge Planning

Aug 23, 2022: https://indico.cern.ch/event/1173741/

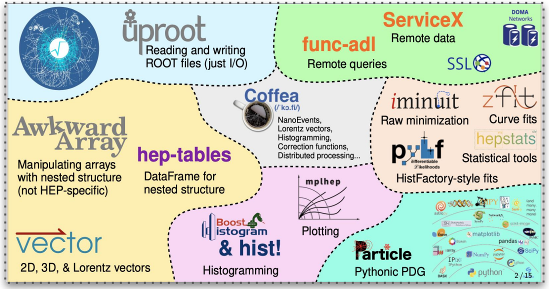
This work was supported by the U.S. National Science Foundation (NSF) Cooperative Agreement OAC-1836650 (IRIS-HEP).

Analysis Grand Challenge

Motivation:

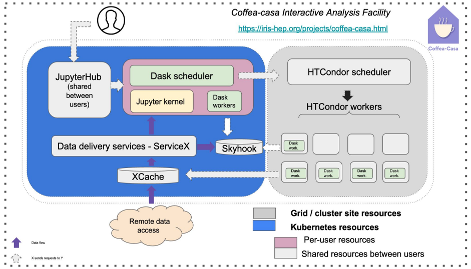
- Allow coping with HL-LHC data sizes by **rethinking data pipeline**
    - Evaluating the new Python “datascience” analysis tools, integrating differentiable analysis pipeline
- Provide flexible, easy-to-use, low latency **analysis facilities**

Analysis Tools



Analysis Facilities

(coffea-casa AF or any other facility matching technical requirements)


Coffea-casa Interactive Analysis Facility
https://iris-hep.org/projects/coffea-casa.html
Coffea-Casa
JupyterHub (shared between users)
Dask scheduler
Jupyter kernel
Dask workers
HTCondor scheduler
HTCondor workers
Data delivery services - ServiceX
Skyhook
XCache
Dask work.
Remote data access
Grid / cluster site resources
Kubernetes resources
Per-user resources
Shared resources between users
Data flow
X sends requests to Y

Execution of AGC analysis benchmark

The AGC is being conducted during **2021–2023**, leaving enough time for tuning software tools and services developed as a part of the IRIS-HEP ecosystem before the start-up of the HL-LHC and *organized together with the US LHC Operations programs, the LHC experiments and other partners.*

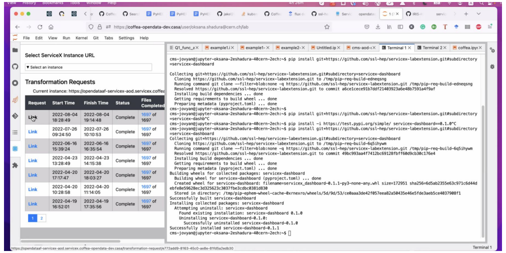
Recent news:

Fellow projects, CMS Open Data Workshop

AGC related IRIS-HEP Fellow projects

- **Durbar Chakraborty:** Metrics to define user activities and engagement on the various coffea-casa Analysis Facility deployments (https://iris-hep.org/fellows/durbar2003.html) - finished
- **Haoran Sun:** Enabling support for MiniAOD Transformer for ServiceX Data Delivery Service (https://iris-hep.org/fellows/hswhhe.html) - in progress
- **Andrii Falko**: documentation of AGC analysis task & implementation in ROOT's RDataFrame (https://iris-hep.org/fellows/andriiknu.html) - in progress
- **Jake Li:** Jupyterhub plugin that will provide users access to a ServiceX dashboard directly from AF (https://iris-hep.org/fellows/jakel2014.html) - finished
    - Collecting list of all ServiceX instances: please see Slack (or send us URLs)

ServiceX JupyterHub plugin



https://coffea-opendata-dev.casa/user/oksana.shadura@cern.ch/lab
File Edit View Run Kernel Git Tabs Settings Help
Q1_func_a
example1.i
example1-
example2-
Untitled.ip
cms-aod-
Terminal 1
Terminal 2
coffea.ipyı

Select ServiceX Instance URL

▼ Select an instance

Transformation Requests

Current instance: https://opendataaf-servicex-aod.servicex.coffea

Request	Start Time	Finish Time	Status	Files Completed
Link	2022-08-04 18:28:49	2022-08-04 19:14:48	Complete	1697 of 1697
Link	2022-07-26 09:24:50	2022-07-26 10:10:53	Complete	1697 of 1697
Link	2022-06-16 15:39:24	2022-06-16 16:35:54	Complete	1697 of 1697
Link	2022-04-23 13:28:49	2022-04-23 14:15:38	Complete	1697 of 1697
Link	2022-04-20 17:17:47	2022-04-20 18:03:27	Complete	1697 of 1697
Link	2022-04-20 10:28:58	2022-04-20 11:14:05	Complete	1697 of 1697
Link	2022-04-19 16:52:01	2022-04-19 17:35:56	Complete	1697 of 1697

1 2

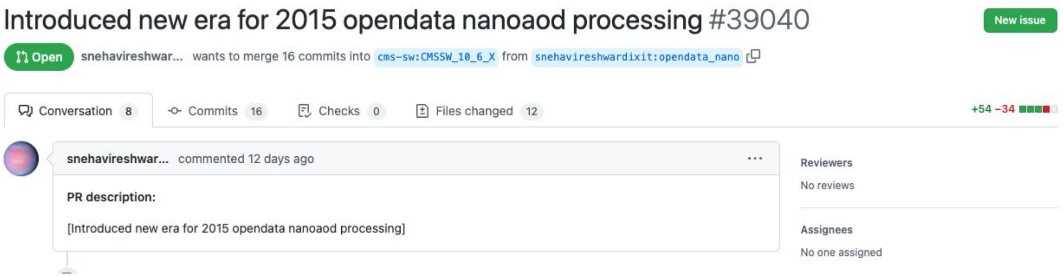
```
cms-jovyan@jupyter-oksana-2eshadura-40cern-2ech:~$ pip install git+https://github.com/ssl-hep/servicex-labextension.git#subdirectory
=servicex-dashboard

Collecting git+https://github.com/ssl-hep/servicex-labextension.git#subdirectory=servicex-dashboard
  Cloning https://github.com/ssl-hep/servicex-labextension.git to /tmp/pip-req-build-edneqsng
  Running command git clone --filter=blob:none -q https://github.com/ssl-hep/servicex-labextension.git /tmp/pip-req-build-edneqsng
  Resolved https://github.com/ssl-hep/servicex-labextension.git to commit a6ce1cece91b7ddf21403923a6e48b7591a4f9af
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
cms-jovyan@jupyter-oksana-2eshadura-40cern-2ech:~$
cms-jovyan@jupyter-oksana-2eshadura-40cern-2ech:~$ pip install git+https://github.com/ssl-hep/servicex-labextension.git#subdirectory
=servicex-dashb^C
cms-jovyan@jupyter-oksana-2eshadura-40cern-2ech:~$ pip install -i https://test.pypi.org/simple/ servicex-dashboard==0.1.0^C
cms-jovyan@jupyter-oksana-2eshadura-40cern-2ech:~$ pip install git+https://github.com/ssl-hep/servicex-labextension.git#subdirectory
=servicex-dashboard
Collecting git+https://github.com/ssl-hep/servicex-labextension.git#subdirectory=servicex-dashboard
  Cloning https://github.com/ssl-hep/servicex-labextension.git to /tmp/pip-req-build-6q5ihywm
  Running command git clone --filter=blob:none -q https://github.com/ssl-hep/servicex-labextension.git /tmp/pip-req-build-6q5ihywm
  Resolved https://github.com/ssl-hep/servicex-labextension.git to commit 49bc993aa4f7412bc69128fbff60d9cb30c176e4
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Building wheels for collected packages: servicex-dashboard
  Building wheel for servicex-dashboard (pyproject.toml) ... done
  Created wheel for servicex-dashboard: filename=servicex_dashboard-0.1.1-py3-none-any.whl size=172951 sha256=6d5ab2355e63c971c6d44d
ebfe8e59628ec3d325623c3037fbe3cdbc0381d830
  Stored in directory: /tmp/pip-ephem-wheel-cache-0xrnexro/wheels/5d/9d/53/ce0aaa3de42f057eea82a50435e46e5fde3aeb5ce4037908f1
Successfully built servicex-dashboard
Installing collected packages: servicex-dashboard
  Attempting uninstall: servicex-dashboard
    Found existing installation: servicex-dashboard 0.1.0
    Uninstalling servicex-dashboard-0.1.0:
      Successfully uninstalled servicex-dashboard-0.1.0
Successfully installed servicex-dashboard-0.1.1
cms-jovyan@jupyter-oksana-2eshadura-40cern-2ech:~$ 
```

https://opendataaf-servicex-aod.servicex.coffea-opendata-dev.casa/transformation-request/e772add9-8163-45c0-ae8e-81fd5a2edb30

Terminal 1

AGC related US.CMS Fellow projects

- **Holly Wingren:** AGC component benchmarking - finished
- **Sneha Dixit:** Data generation (NanoAOD) from CMS Opendata 2015 (recent release), https://github.com/cms-sw/cmssw/pull/39040 - finished
    - Will replace AGC input data (flat ntuples) by NanoAOD files produced in this official way
    - Expect that these NanoAOD files will also be provided officially/centrally by CMS



Introduced new era for 2015 opendata nanoaod processing #39040

Open snehavireshwar... wants to merge 16 commits into cms-sw:CMSSW_10_6_X from snehavireshwardixit:opendata_nano

snehavireshwar... commented 12 days ago

PR description:

[Introduced new era for 2015 opendata nanoaod processing]

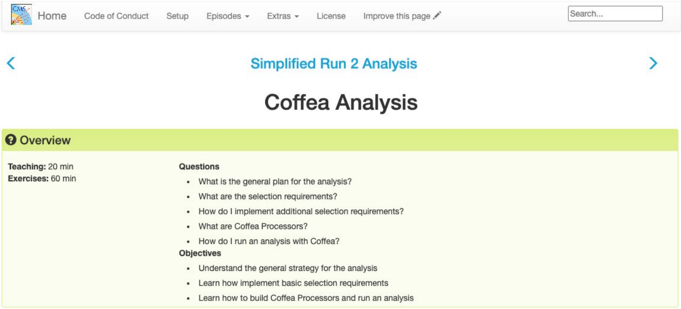
Reviewers
No reviews

Assignees
No one assigned

ICHEP 2022

- Presented **AGC at ICHEP** (slides)
- There is interest in adding **ML inference** to the workflow
  - Clearly frequently used in practice
  - Less clear how to best integrate it: efficient use of accelerators, likely many site-dependent differences
  - We are looking into including this in the pipeline
- Less in scope, but also interesting: **ML training workflow**
  - Could become a standalone project?

CMS Open Data workshop



Simplified Run 2 Analysis

Coffea Analysis

Overview

Teaching: 20 min
Exercises: 60 min

Questions

- What is the general plan for the analysis?
- What are the selection requirements?
- How do I implement additional selection requirements?
- What are Coffea Processors?
- How do I run an analysis with Coffea?

Objectives

- Understand the general strategy for the analysis
- Learn how implement basic selection requirements
- Learn how to build Coffea Processors and run an analysis

- Aug 1-4: **CMS Open Data workshop** (https://indico.cern.ch/event/1139022/)
  - Hands-on tutorials of how to use CMS Open Data
  - Included demonstration of using Run-2 Open Data, heavily based on AGC (tutorial page)
    - the AGC analysis is turning into an “official” CMS Open Data demonstration
- CMS **re-processed input files** for analysis (more precise calibration, similar ntuple structure)
  - Will investigate switching to this, or directly to NanoAOD format

AGC updates:

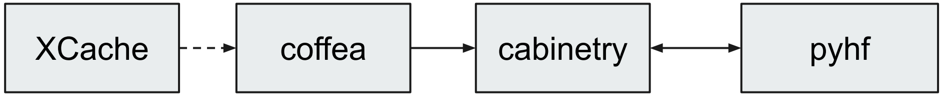
PyHEP + ACAT, benchmarking plans, site status

Next targets: PyHEP + ACAT

- PyHEP 2022 (https://indico.cern.ch/e/PyHEP2022)
  - Presenting a tutorial, focused on the analysis pipeline & interactions between the libraries used
  - Expect participants to be able to follow along via Binder (using small set of input files)

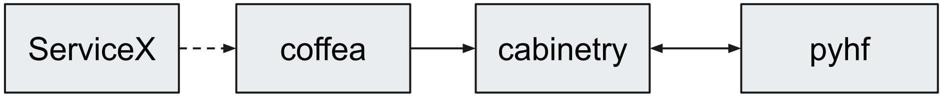
- ACAT 2022 (https://indico.cern.ch/event/1106990/)
  - Talk focused on first performance benchmarking results
  - Want to measure multiple setups:
    - Comparison of processing remote files read through XRootD (with / without XCache) and local files
    - At UNL: scaling to Tier-2 HTCondor queue vs HTCondor in k8s (running on new hardware at UNL)
    - Scaling with fraction of data accessed in files
    - How I/O limited is the setup (by removing event processing & comparing to theoretical limits of what the hardware could provide)
    - Happy to hear other suggestions!
  - ServiceX use in this context relies on next release (thanks to Burt Holzman for finding / resolving minio bug!)

Pipelines for performance testing



Columns from NanoAOD or flat ntuples processed by coffea

Starting out with plain coffea, reading inputs through XCache from remote



Columns from NanoAOD or flat ntuples from ServiceX

Want to also compare effect of serving data through ServiceX

Towards AGC at various sites across the US

- Investigate **`XCache` setup and configuration** for AGC execution
  - Deployed already / planned? Restrictions to which kinds of files can be cached?
- **Deploy and test `coffea`** at your facility
  - Please feel free to report issues to us, we can help relay them
- Investigate **scaling for `coffea`** (various backends available)
  - e.g. latency, performance with number of workers, workers a single user can get, …
- **Deploy and test `ServiceX`**
  - AGC only uses uproot transformers currently
  - Expect significant performance improvements from upcoming release
- How to **support additional dependencies** (on interactive nodes + workers)
  - Should users bring images?

- **Helps us help you**: **which issues are you running into** that we should prioritize / communicate to the respective teams?

What currently runs where? (please help us fill in the gaps!)

	BNL	FNAL	SLAC	UNL	UChicago
basic coffea (e.g. IterativeExecutor) -> notebook with USE_DASK = False	✓	✓		✓	✓
coffea scaling (e.g. with Dask) -> notebook with default settings*		✓		✓ (using HTCondor @ Tier2, planning to switch to k8s)	✓ (occasional segfaults at scale)
standalone ServiceX -> notebook (no configuration)	✓ (?)	✓		waiting for new release	✓
ServiceX+coffea+scaling -> notebook with PIPELINE = "servicex_processor"				waiting for new release	✓
XCache support	✓	✓ (some performance caveats, to be understood)	✓	✓	✓

* may need site-dependent Dask cluster configuration, see implementation, please get in touch in case of questions

AGC: timeline & plans

- **Nov 2021:** first demonstration of toolchain at AGC workshop (agenda)
- **April 2022:** second iteration of AGC workshop, new ttbar analysis pipeline (agenda)
- **Summer 2022:** benchmarking of system components in the AGC context
  - Results to be shown at ACAT 2022
- **Spring 2023:** execution of AGC at full scale

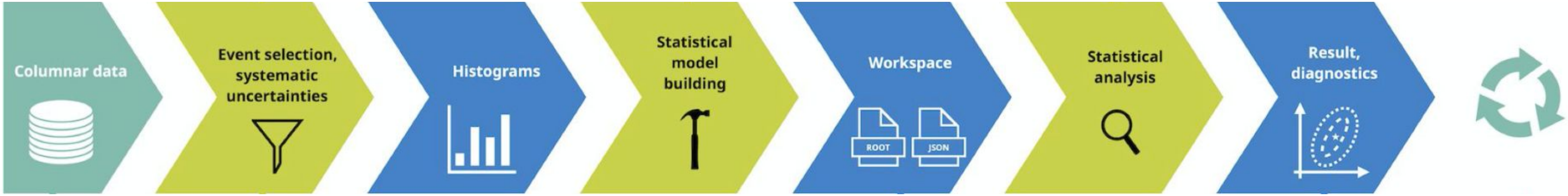
- Proposal for **next meeting**: **Sep 27, 2022** (9:00 PT / 11:00 CT / 12:00 ET / 18:00 CERN)

Stay in touch via analysis-grand-challenge@iris-hep.org (sign up: google group link), and please also feel free to contact us if you'd like to get involved or have any questions!

Backup slides

What do we mean by “analysis”?

- In view of the HL-LHC: “analysis” **starts** from centrally produced **DAOD_PHYSLITE**
  - Includes **all subsequent steps** to produce results needed for publication
    - (Re-) calibrate objects & calculate columns for systematic variations
    - Filter events & calculate observables
    - (Optional) histogramming
    - Construction of statistical model + statistical inference
  - Do all these steps in a **reproducible** way



The AGC analysis setup

- Main AGC analysis example is based on **Run-2 CMS Open Data** (~400 TB of miniAOD available)
    - **Open Data is crucial**: allows everyone to participate!
    - Input data currently ntuples derived from miniAODs, will switch to nanoAOD once available
        - nanoAOD setup will more closely correspond to PHYSLITE workflows

- Generic single-lepton **ttbar** selection, simple top reconstruction, x-sec measurement
    - Including on-the-fly evaluation of systematic uncertainties
    - Flexibility to also turn setup into a BSM search
    - Currently using ~4 TB of ntuple inputs (~1B events before cuts) → through Rucio and xrootd(https) @ UNL

- Goal of setup is showing **functionality**, **not** discovering new physics
    - Want to capture workflow, but can made-up tools for calibrations & systematic uncertainties

Goals of the AGC

- Demonstrate method for **handling HL-LHC data pipeline requirements**
    - Large data volumes of O(100 TB) + bookkeeping
    - Handling of different types of systematic uncertainties
    - Use of PHYSLITE / nanoAOD formats, aligned with LHC experiments

- **“Interactive analysis”**: turnaround time of ~minutes or less
    - Made possible by highly parallel execution in short bursts, low latency & heavy use of caching

- **Specify all analysis details** to allow for **re-implementations** and re-use for benchmarking

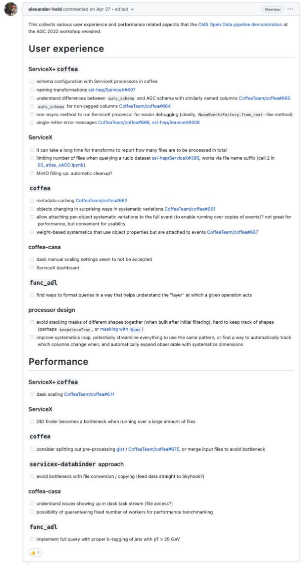
- Stretch goal: **differentiable analysis pipeline** (see e.g. https://indico.cern.ch/event/1096431/)
    - Investigate end-to-end analysis optimization, evaluate usefulness vs cost of gradient information

Status

- **Demonstrated analysis pipeline** at last AGC workshop: https://indico.cern.ch/e/agc-tools-2
  - Workshop also features details of all tools & services used in pipeline
  - Pipeline setup
    - `ServiceX` delivers columns following declarative `func_adl` request
    - `coffea` orchestrates distributed event processing & histogram production
      - Using `uproot`, `awkward-array`, `vector`
    - Visualization with `hist` & `mplhep`
    - Statistical model construction with `cabinetry` & inference with `pyhf`

- Pipeline demonstrator highlighted possible **usability & performance improvements**
  - Everything tracked in agc#64
  - Main performance bottlenecks are being addressed

- **Everything is openly developed** (AGC repository)
  - Including categorization of datasets in terms of role in AGC demonstrator (AGC repository)



alexander-held commented on Apr 27 • edited

This collects various user experience and performance related aspects that the CMS Open Data pipeline demonstration at the AGC 2022 workshop revealed.

User experience

ServiceX+ `coffea`

- □ schema configuration with ServiceX processors in coffea
- □ naming transformations ssl-hep/ServiceX#407
- □ understand differences between `auto_schema` and AGC schema with similarly named columns CoffeaTeam/coffea#665
- □ `auto_schema` for non-jagged columns CoffeaTeam/coffea#664
- □ non-async method to run ServiceX processor for easier debugging (ideally, `NanoEventsFactory.from_root` -like method)
- □ single-letter error messages CoffeaTeam/coffea#666, ssl-hep/ServiceX#408

ServiceX

- □ it can take a long time for transforms to report how many files are to be processed in total
- □ limiting number of files when querying a rucio dataset ssl-hep/ServiceX#395, works via file name suffix (cell 2 in 03_atlas_xAOD.ipynb)
- □ MinIO filling up: automatic cleanup?

`coffea`

- □ metadata caching CoffeaTeam/coffea#662
- □ objects changing in surprising ways in systematic variations CoffeaTeam/coffea#661
- □ allow attaching per-object systematic variations to the full event (to enable running over copies of events)? not great for performance, but convenient for usability
- □ weight-based systematics that use object properties but are attached to events CoffeaTeam/coffea#667

coffea-casa

- □ dask manual scaling settings seem to not be accepted
- □ ServiceX dashboard

`func_adl`

- □ find ways to format queries in a way that helps understand the "layer" at which a given operation acts

processor design

- □ avoid stacking masks of different shapes together (when built after initial filtering), hard to keep track of shapes (perhaps `keepdims=True` , or masking with `None`)
- □ improve systematics loop, potentially streamline everything to use the same pattern, or find a way to automatically track which columns change when, and automatically expand observable with systematics dimensions

Performance

ServiceX+ `coffea`

- □ dask scaling CoffeaTeam/coffea#611

ServiceX

- □ DID finder becomes a bottleneck when running over a large amount of files

`coffea`

- □ consider splitting out pre-processing gist / CoffeaTeam/coffea#675, or merge input files to avoid bottleneck

`servicex-databinder` approach

- □ avoid bottleneck with file conversion / copying (feed data straight to Skyhook?)

coffea-casa

- □ understand issues showing up in dask task stream (file access?)
- □ possibility of guaranteeing fixed number of workers for performance benchmarking

`func_adl`

- □ implement full query with proper b-tagging of jets with pT > 25 GeV

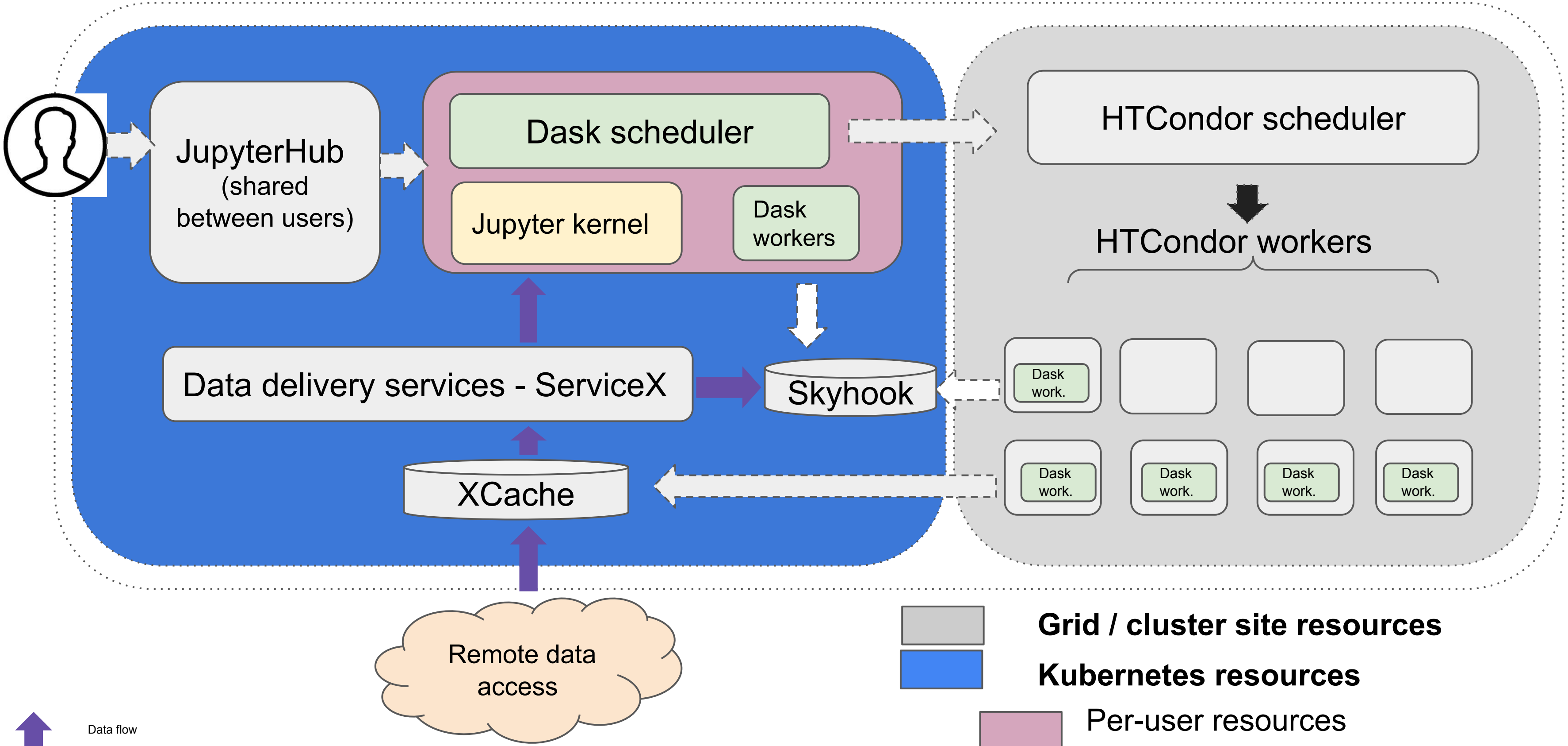
Expanding the AGC, connections with ATLAS & CMS

- Looking to further **expand AGC demonstrator** & **connect more closely to ATLAS and CMS**
  - "ATLAS version": PHYSLITE input + (simplified?) CP tools (related: PHYSLITE schema for coffea)
  - "CMS version": more CMS specific NANOAOD input
  - ROOT **`RDataFrame`-based implementation** of pipeline
  - Would be interesting to talk to analyzers currently using **distributed RDF / `coffea`** in ATLAS/CMS
    - Please get in touch!
  - Actively **looking for more person power** to expand scope

- Interested to hear **feedback**: how to best **align AGC effort with ATLAS**
  - Balance between retaining R&D nature and potential usefulness on Run-3 scale

Running the AGC: coffea-casa AF example

https://iris-hep.org/projects/coffea-casa.html



⬆ Data flow

⇧ X sends requests to Y

▨ Grid / cluster site resources

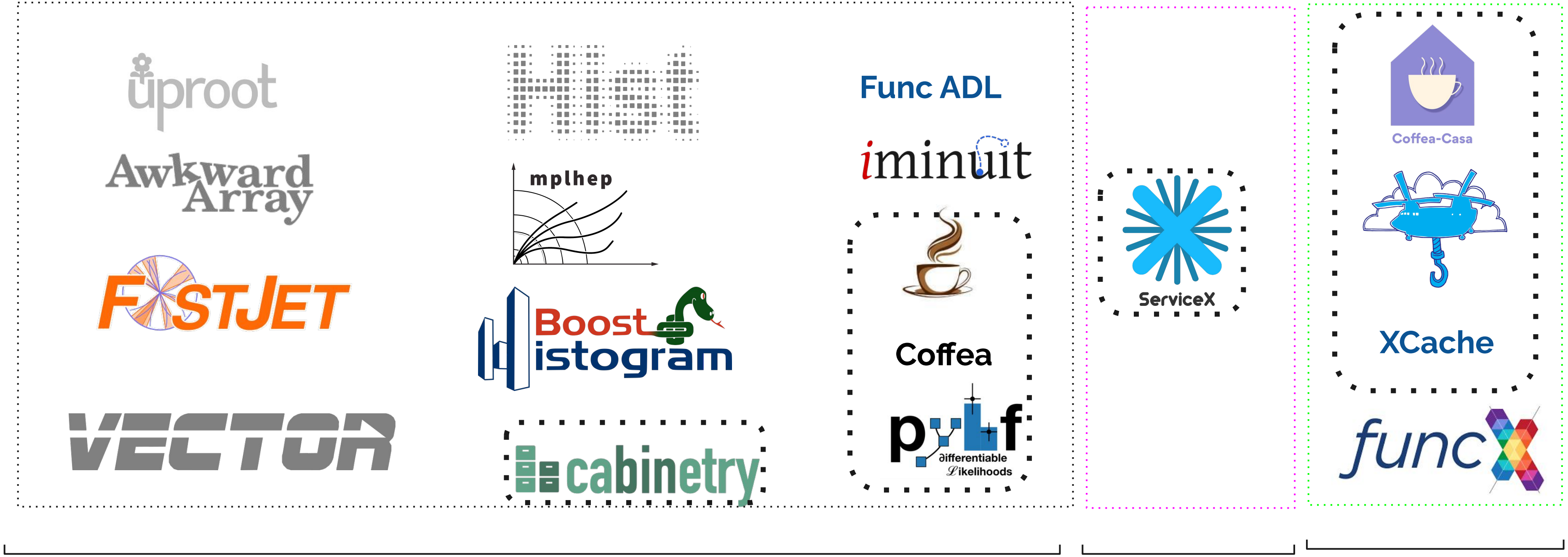
🟦 Kubernetes resources

🟪 Per-user resources

□ Shared resources between users

Matous Adamec, Garhan Attebury, Brian Bockelman, Ken Bloom, Carl Lundstedt, John Thiltges, Oksana Shadura, Andrew Wightman

Tools & services we are focusing on for benchmarking



Analysis specific frameworks and packages (available in Docker container)

Data delivery service (k8s)

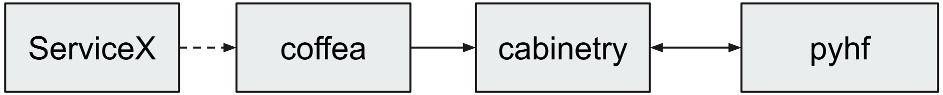
Optional services (k8s)

Requirements for AFs

Modern authentication (AIM/OIDC), tokens, macaroons, scitokens	Ongoing R&D on moving to use scitokens natively for AF (write/read)
Efficient data delivery and data management technologies	Ongoing work on integration ServiceX/Skyhook data delivery services
Columnar analysis and support new pythonic ecosystem	
Modern deployment and integration techniques	
Support for object storage	
Efficient data caching solutions	Integrating XCache in analysis pipeline
Easy integration with existing HPC resources	Looking to add support for other batch systems and task managements frameworks

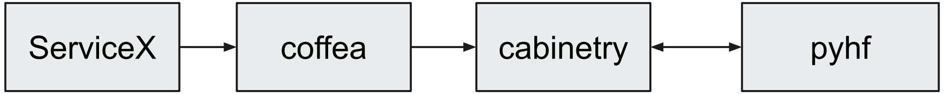
[Coffea-casa team]

Analysis Grand challenge pipelines



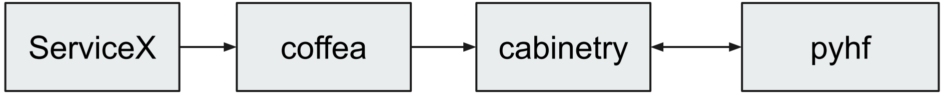
Columns from NanoAOD or flat ntuples from
ServiceX or directly through coffea

Generic analysis pipeline based on *Open Data dataset* allowing to easily port AGC to other analysis frameworks



Columns from NanoAOD
and request column
from MiniAOD (only ServiceX)

CMS specific analysis pipeline based on *Open Data datasets* and *CMS datasets*



Columns from PHYSLITE
and request column
from PHYS (only ServiceX)

ATLAS specific analysis pipeline based on *ATLAS datasets*

Analysis Facilities participating in AGC (for now)

CMSAF @ T2 Nebraska
"Coffea-casa"
https://coffea.casa

OpenData AF @ T2 Nebraska
"Coffea-casa"
https://coffea-opendata.casa

ATLAS AF @ Scalable System Lab
(UChicago)
"Coffea-casa"

New facility with ATLAS IAM, setting this up generated valuable feedback for future coffea-casa developments.

Elastic AF @ Fermilab
(U.S. CMS)

FNAL team is participating in AGC , helping to test coffea-casa AF and re-using its components at the FNAL facility

U.S. ATLAS facilities
BNL / SLAC are participating in AGC
(we are evaluating the possibility to use coffea-casa experience)

Benchmarking & performance

- **How to benchmark AF performance** to demonstrate HL-LHC readiness?
  - **Multi-user environments**, where each user has a different workflow & requirements
    - Collection of "representative" analyses to simulate many users?
  - `Data volume / (time * numCPUs)`
    - Does not inherently account for I/O bound jobs where more CPUs != higher throughput
    - Assumption: `# branches read` matters, while `total # branches` does not, does this generally hold?
  - Fair to assume that **all input data is cached** (`xCache`)?
  - **Latency** is crucial for interactive workflows, how to best account for that?

- **Related questions**
  - How much control does a user have over caching (e.g. when re-calculation from scratch is expensive)?