DOMA Deep Dive: University of Illinois

Ben Galewsky (bengal1@Illinois.edu)
Mark Neubauer (msn@illinois.edu)

iris hep

ILLINOIS
NCSA | National Center for Supercomputing Applications

ILLINOIS
Physics
COLLEGE OF ENGINEERING

The Illinois DOMA Team

Mark Neubauer

Professor of Physics
University of Illinois at Urbana-Champaign
Affiliate appointments in ECE Dept. & NCSA

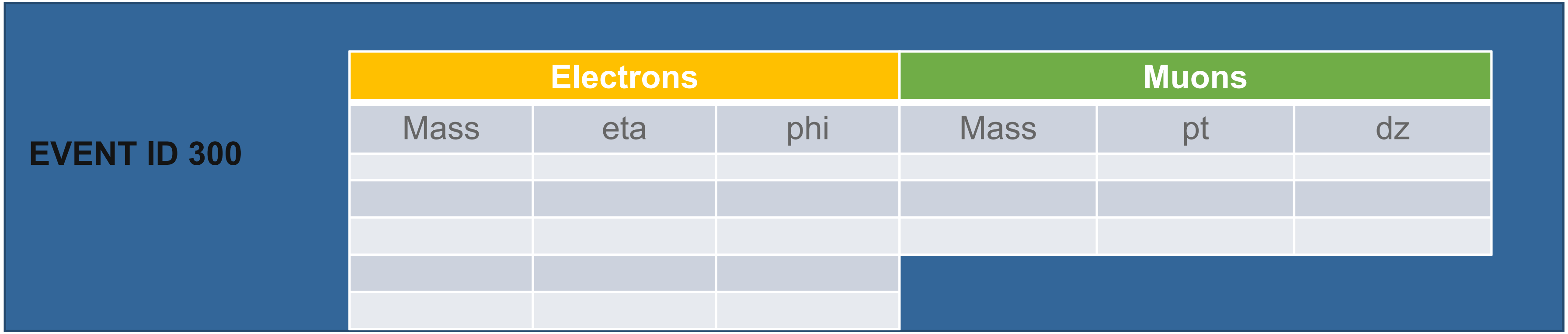
Ben Galewsky

Research Programmer
Innovative Software and Data Analysis Group
National Center for Supercomputing Applications

Current Scope of DOMA Work in IRIS-HEP

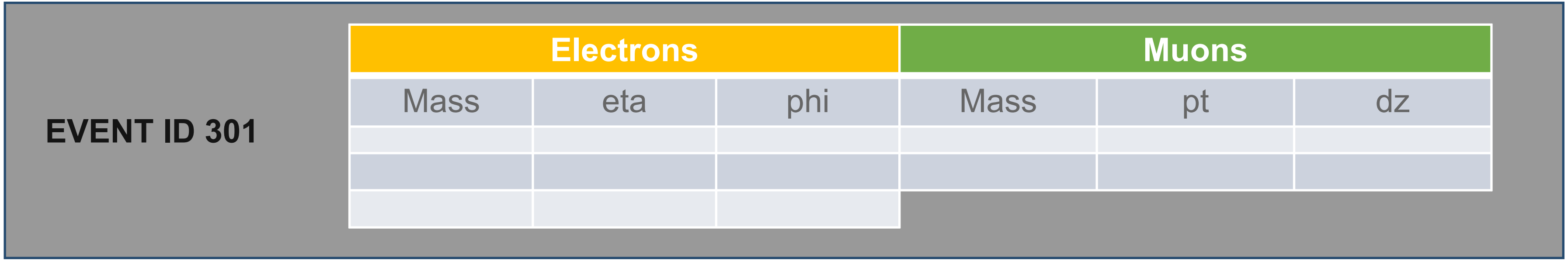
- ➢ Our interest & effort is in an *intelligent data delivery service* for analysis
  - ○ This is the DOMA-side of a coherent R&D effort within IRIS-HEP leading to innovative, multi-experiment data analysis systems and software for HEP
  - ○ Systems have not been optimized for analysis in ATLAS/CMS, only production
- ➢ Our current approach is centered on a *columnar, query-based system*
  - ○ To my knowledge, this was first proposed for HL-LHC analysis (independently) by Neubauer and Pivarski during the round-table discussion at the HSF CWP Kickoff meeting at SDSC in Jan 2017. This was then fleshed-out into the CWPs.
  - ○ To my knowledge, an “intelligent”/accelerated service layer for data delivery between future data lakes and consumers was ServiceX proposed by UChicago
- ➢ The status of Illinois work on a *columnar data delivery service* follows

Events in Root



EVENT ID 300

Electrons			Muons		
Mass	eta	phi	Mass	pt	dz



EVENT ID 301

Electrons			Muons		
Mass	eta	phi	Mass	pt	dz

Event Loop Processing

- Traditional Pattern:
  - Load values from event into local variables
  - Evaluate several expressions
  - Store Derived Values
  - Repeat for each event
- Advantages
  - Familiar to physicists
- Disadvantages
  - Not optimized for CPU vector processing operations
  - Not easily portable to GPUs

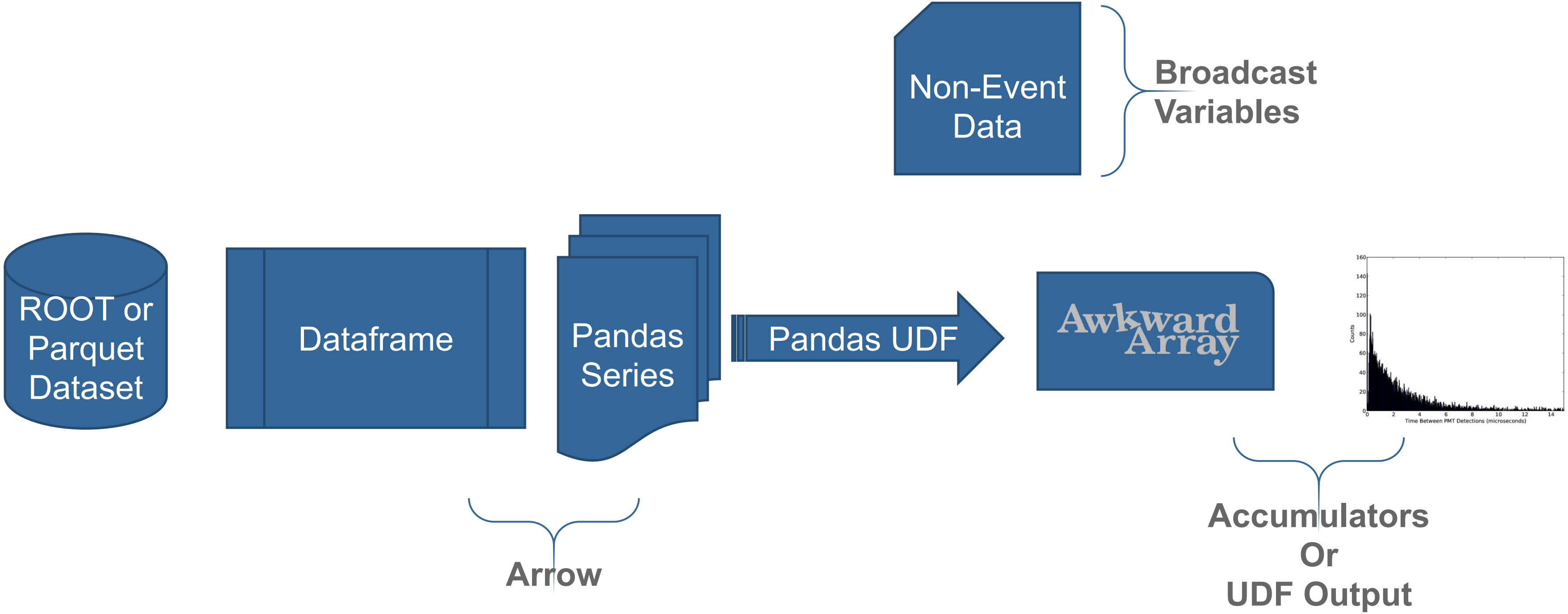
Columnar Analysis

- New Pattern
  - Load values from many events into contiguous arrays
    - Nested content is represented as flat arrays with offsets
  - Evaluate several array operations
  - Store derived values
  - Repeat for next batch of values
- Disadvantages
  - New paradigm for physicists
  - Not inherently supported by Root
- Advantages
  - Takes advantage of CPU vectorized operations
  - Easily ported to GPUs
  - Easy and fun to write

Spark-HEP-Query

- Abstract away the machinery for running columnar analysis
- Physicists write a class that has a calc method that accepts a dictionary of Physics Objects
- Same science code can be run:
  - Locally in Uproot
  - On Spark
  - Parsl on the Grid (in progress)

Spark-Based Analysis



ROOT or Parquet Dataset

Dataframe

Pandas Series

Pandas UDF

Awkward Array

Non-Event Data

Broadcast Variables

Arrow

Accumulators
Or
UDF Output

Issues with Framework

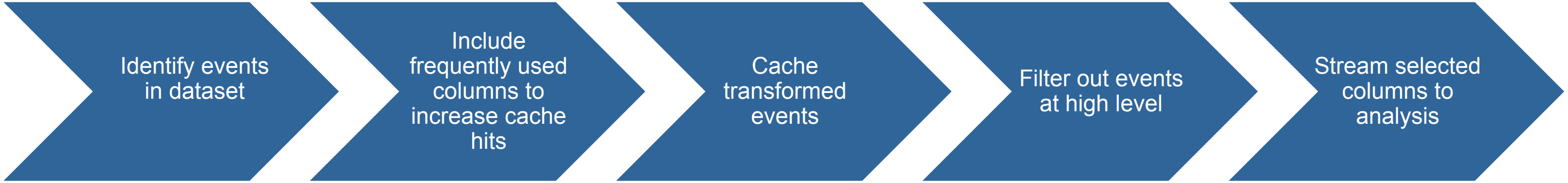
- Expensive to Load ROOT files into Parquet
- Java ROOT Reader can only handle simple ROOT files
  - CMS NanoAOD
- We don’t have existing Spark infrastructure to run jobs on

Looking at the Wider Environment

- ROOT datasets are Large
- Expensive to move datafiles around the world
- Many of the data records require extensive dependencies to read
- The vast majority of file’s properties are not used for analysis
- Many of the properties are common to most analysis

Service X

A distributed, caching columnar data service



Identify events in dataset

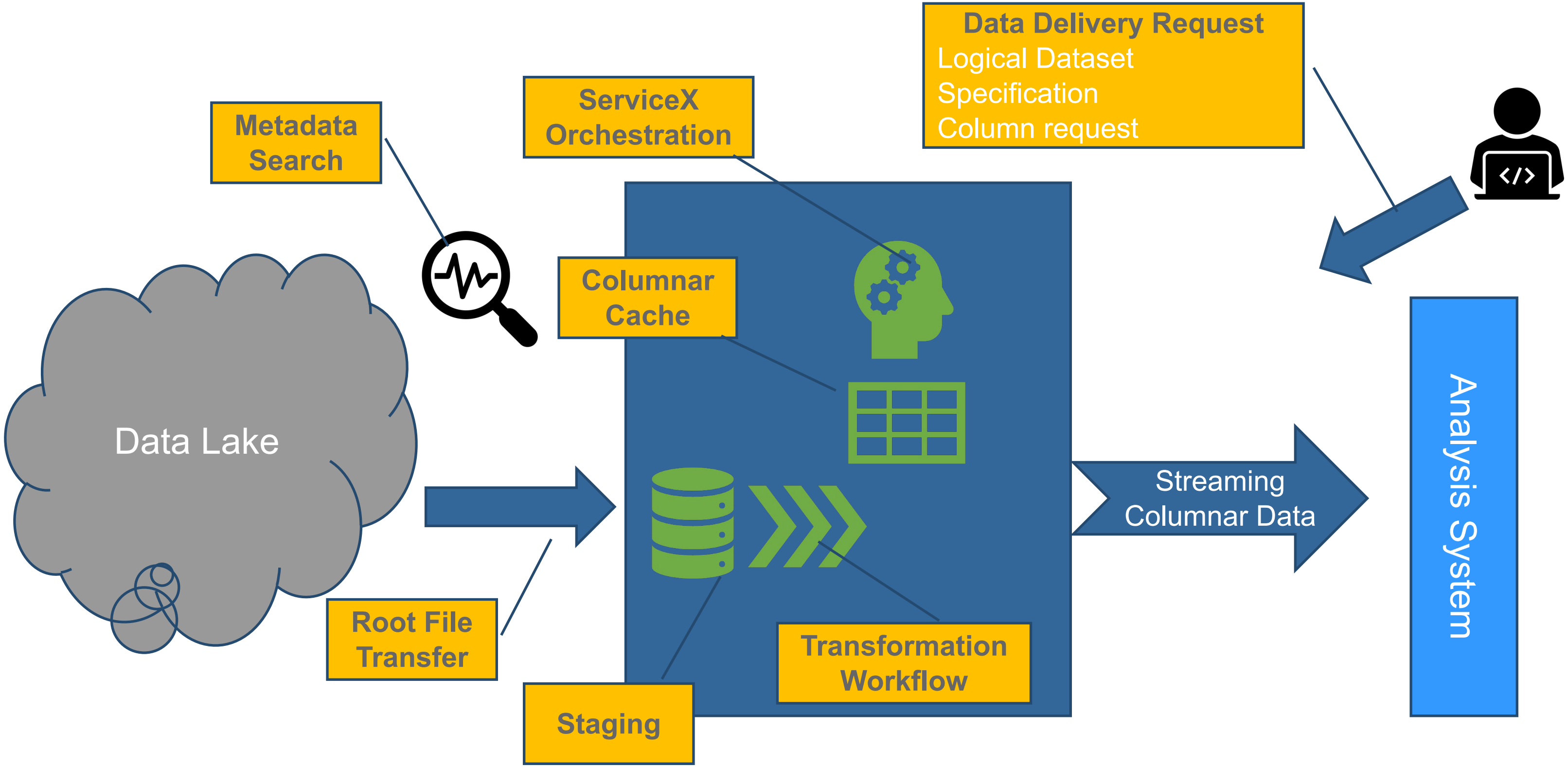
Include frequently used columns to increase cache hits

Cache transformed events

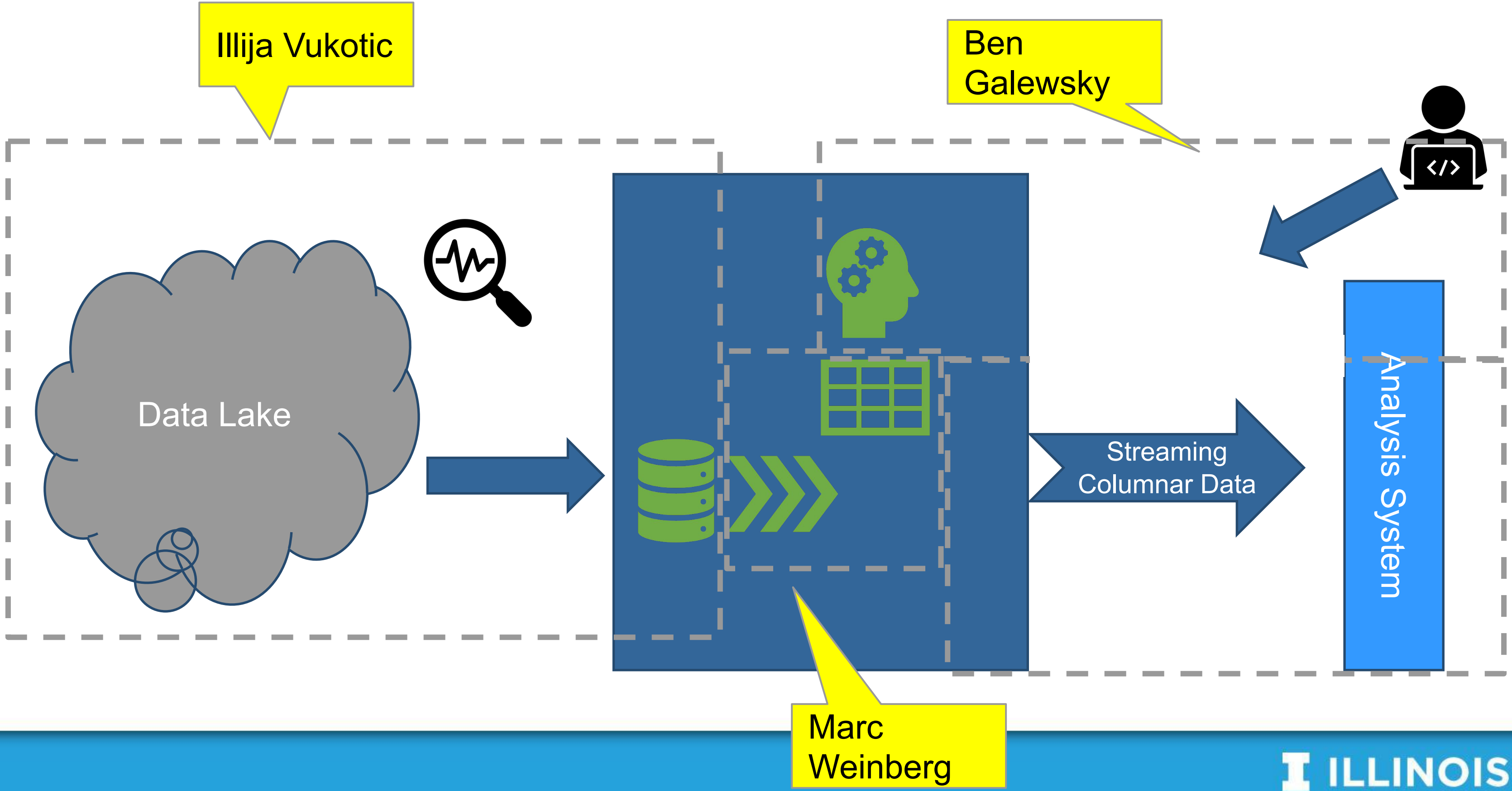
Filter out events at high level

Stream selected columns to analysis

Architecture



Implementation



Component Details

- Data Lake
  - Most likely experiment specific.
  - May be regional replicas
- Metadata Search
  - Find Root file references by logical dataset identification
  - CMS has DBS for this
- Root File Transfer
  - Transfer datafiles from lake to staging area

Component Details

- Staging
  - Root files are transferred from the data lake and staged in local disk prior to transformation
  - Could be staged in XCache
- Transformation Workflow
  - Container based and carefully versioned
  - Code for extracting requested branches no matter how complicated the Root file is

Component Details

- Columnar Cache
  - Cache to hold transformed data
  - Columnar format to efficiently serve up only requested columns
  - Can be indexed to efficiently filter out events
- ServiceX Orchestration
  - Receives data delivery requests
  - Determines if data can be served from cache
  - Upscales requests to include frequently referenced columns to improve cache reusability
  - Orchestrates data download and transformation for cache misses

Output From Service

- Options under consideration
  - Stream Arrow Buffers via Kafka
    - Stream into analytic spark cluster
    - Stream to local parquet file writer
  - Write to local file system and use GridFTP to transfer

Current Status

1. Basic REST service
2. Connection to Rucio
3. Transformer container works with xAOD files. Only single branches
4. Streaming service

Runs in Kubernetes