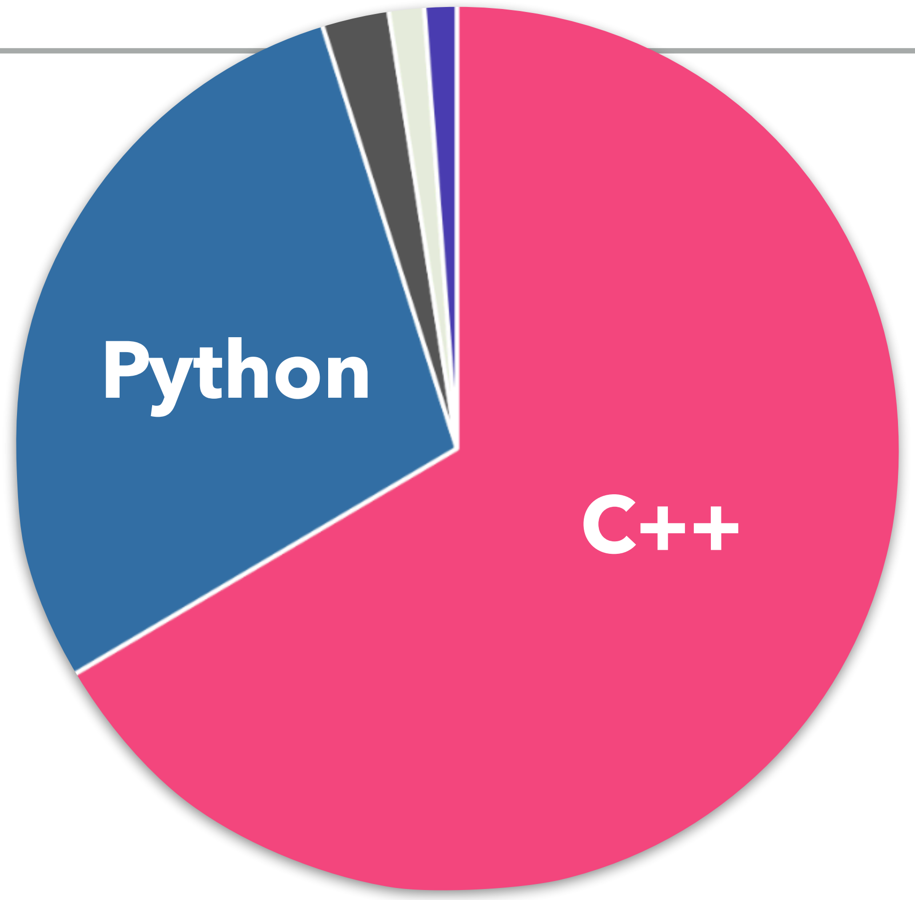
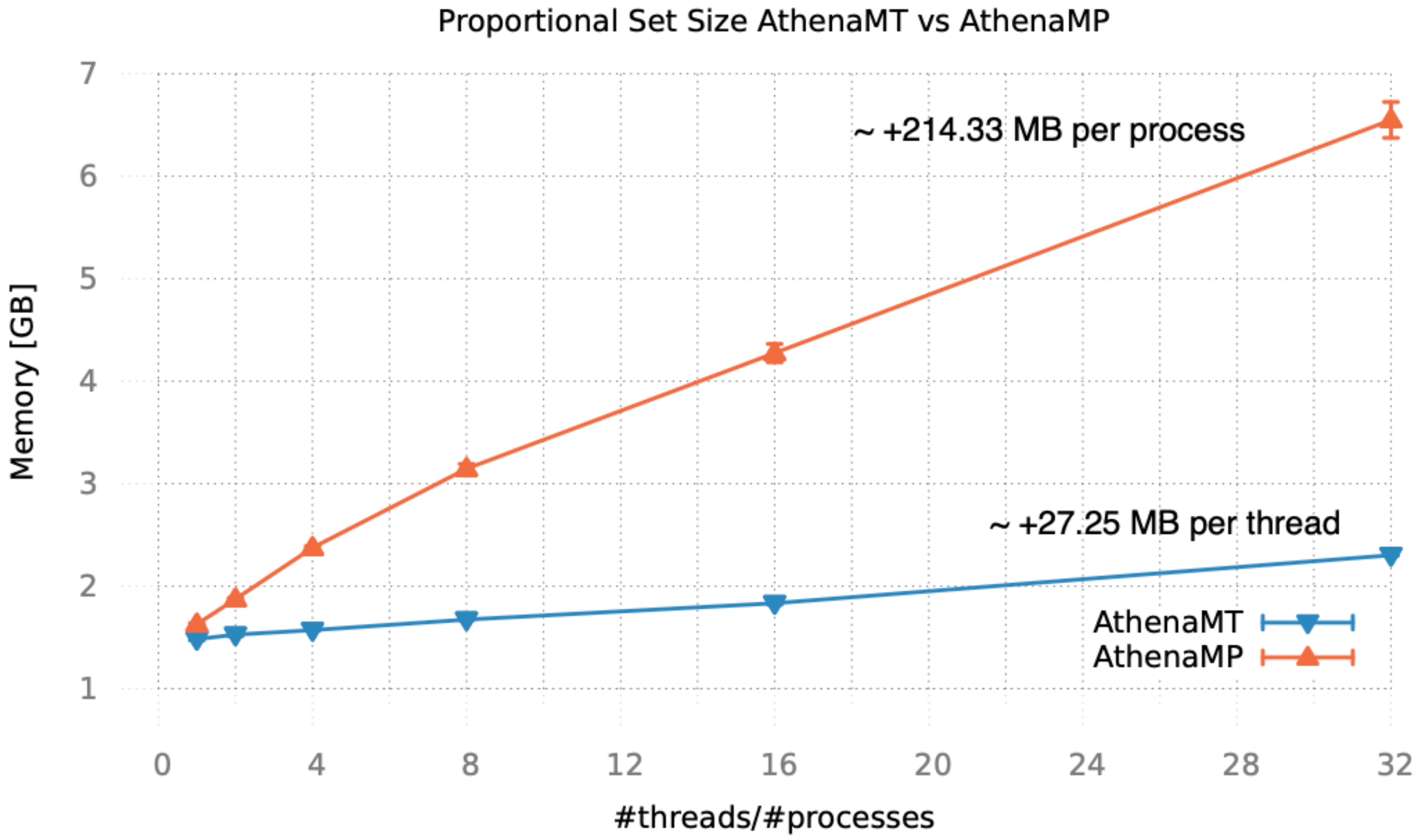
INTRODUCTION

Our main framework is Athena, which we are in the progress of migrating to become multithreaded for run-3 (main motivation: memory)

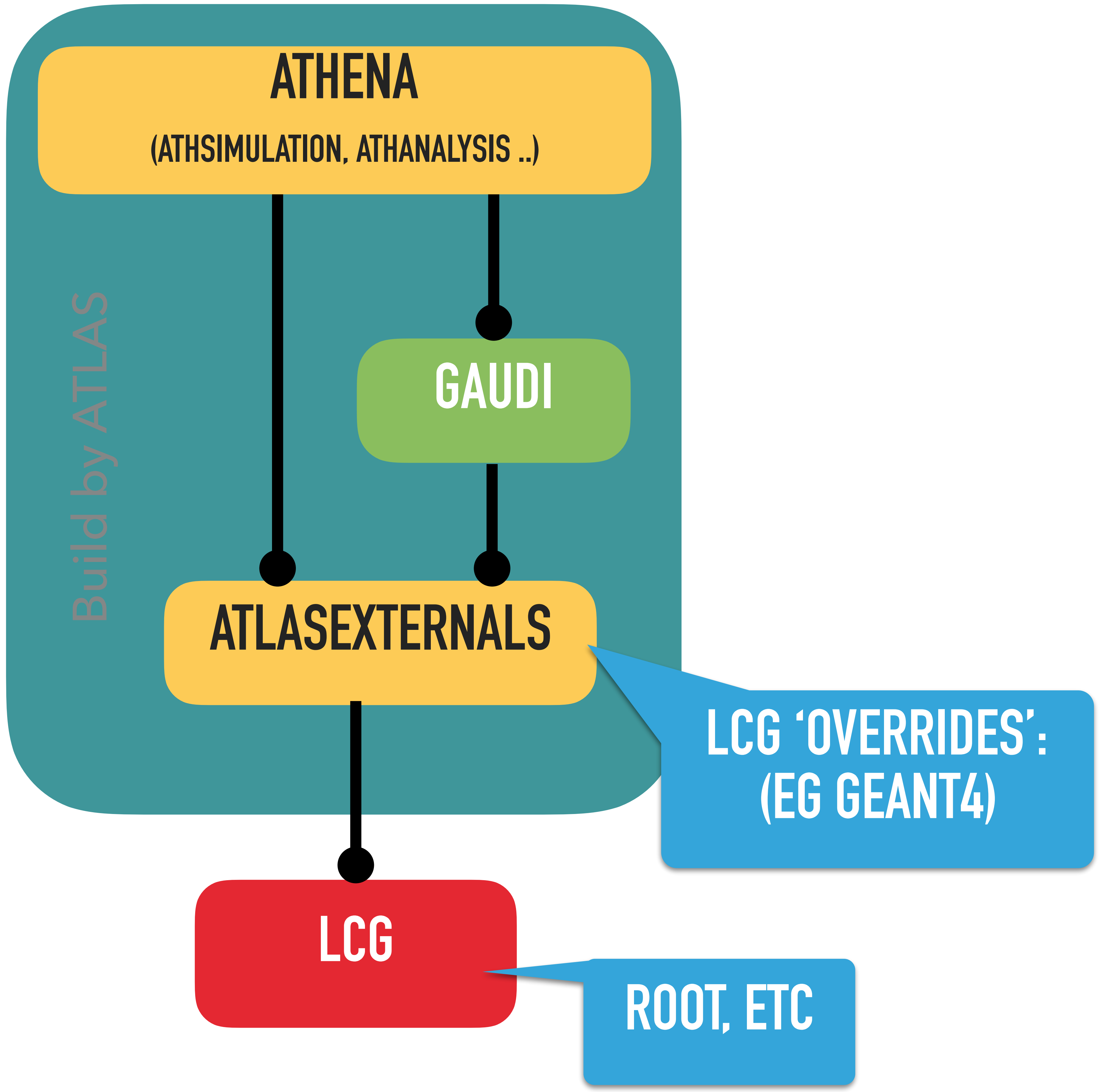
  - Our Athena repository is >1 million lines of python and ~4 million lines of C++
  - ~250 unique committers to master last year, ~30 commits per day

- Of course we have a lot of software in other repositories, but these are much smaller (and less relevant for the current talk)
- There are many ways to look at the ATLAS offline software and we could not come up with a meaningful unique diagram
- Possible points of views:
  - **Build-View**
  - **Components-View**
  - **(Coarse) Analysis Model View**





BUILD-VIEW OF ATHENA



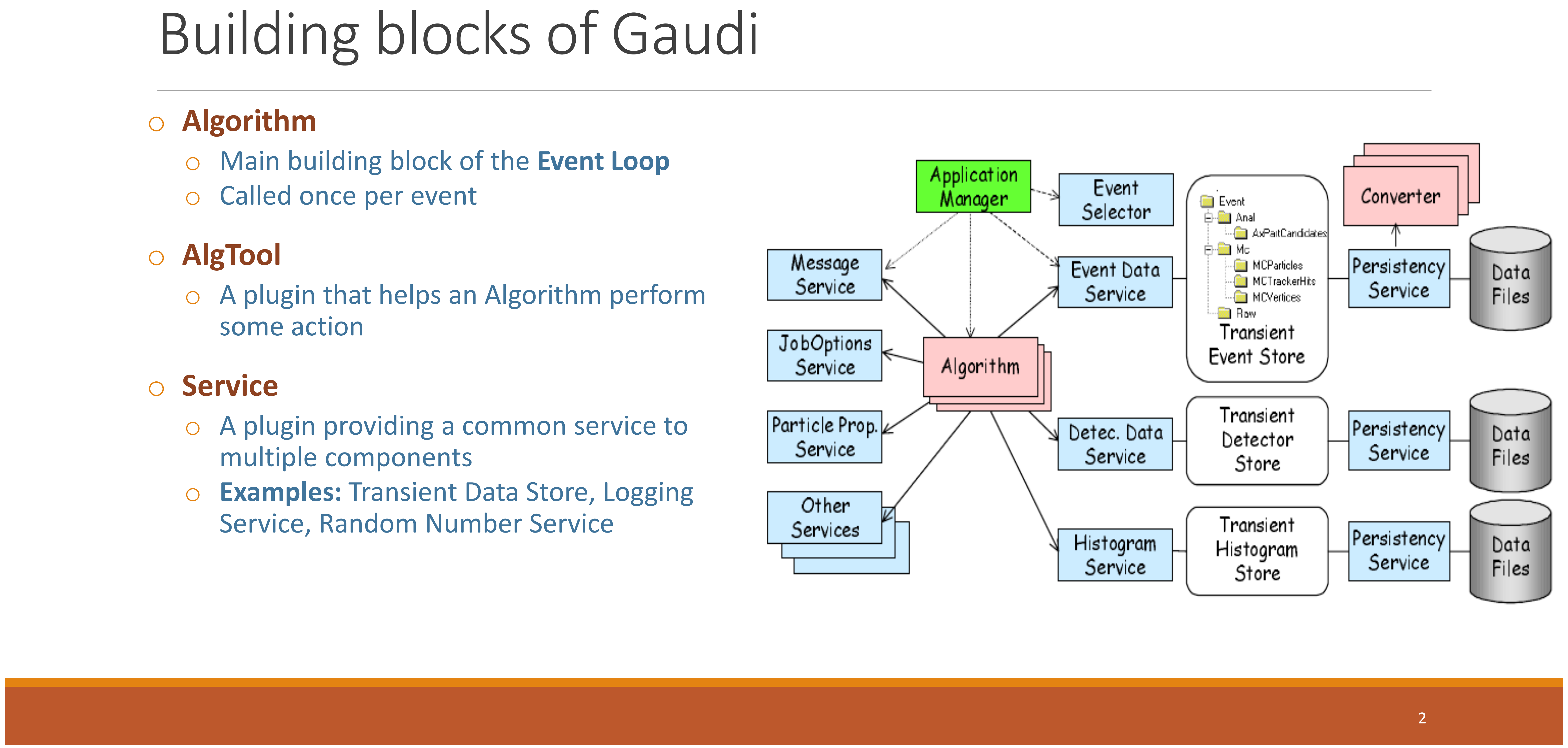
LCG 'OVERRIDES': (EG GEANT4)

ROOT, ETC

- Disambiguation of **Package**:
  - Inside Athena: a directory containing a CMakeLists.txt file and typically a bunch of source file
    - Can recompile one or few packages for development or debugging
    - Allows us to build subsets of the source-tree for dedicated workflows:
      - AthSimulation, AthGeneration, AthDataQuality, AthAnalysis
  - For an external packaging system (like spack), Athena is one package (like ROOT, or geant4)
  - We have detailed instructions on how to build Athena:
  - https://atlassoftwaredocs.web.cern.ch/guides/build_release/

COMPONENT VIEW

- Athena is based on GAUDI, so uses similar component
  - Algorithms, Tools and Services
- Also, handles and a scheduler (see later)



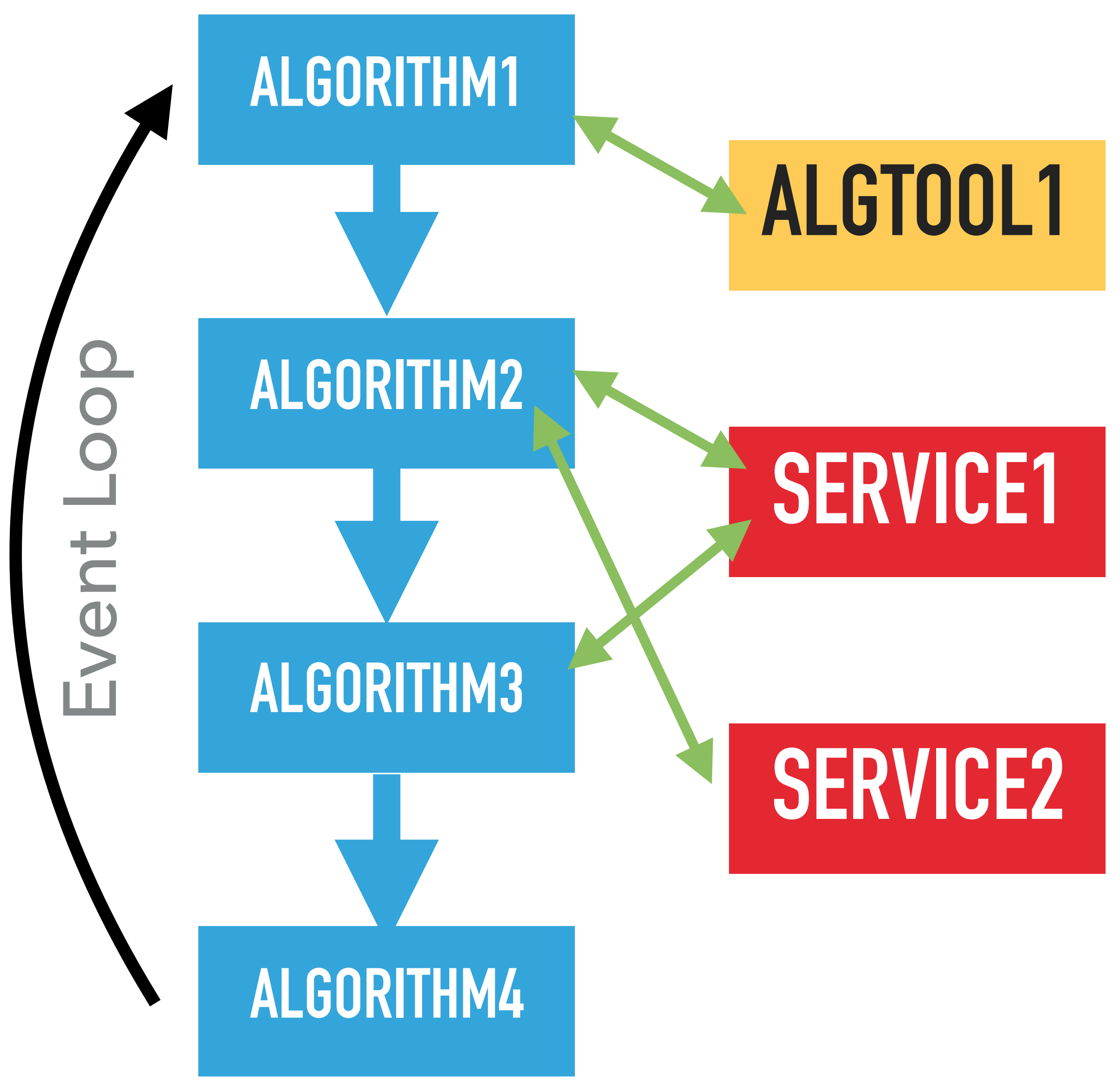
Building blocks of Gaudi

- **Algorithm**
  - Main building block of the **Event Loop**
  - Called once per event
- **AlgTool**
  - A plugin that helps an Algorithm perform some action
- **Service**
  - A plugin providing a common service to multiple components
  - **Examples:** Transient Data Store, Logging Service, Random Number Service

[Intro to Gaudi/Athena]

COMPONENT VIEW

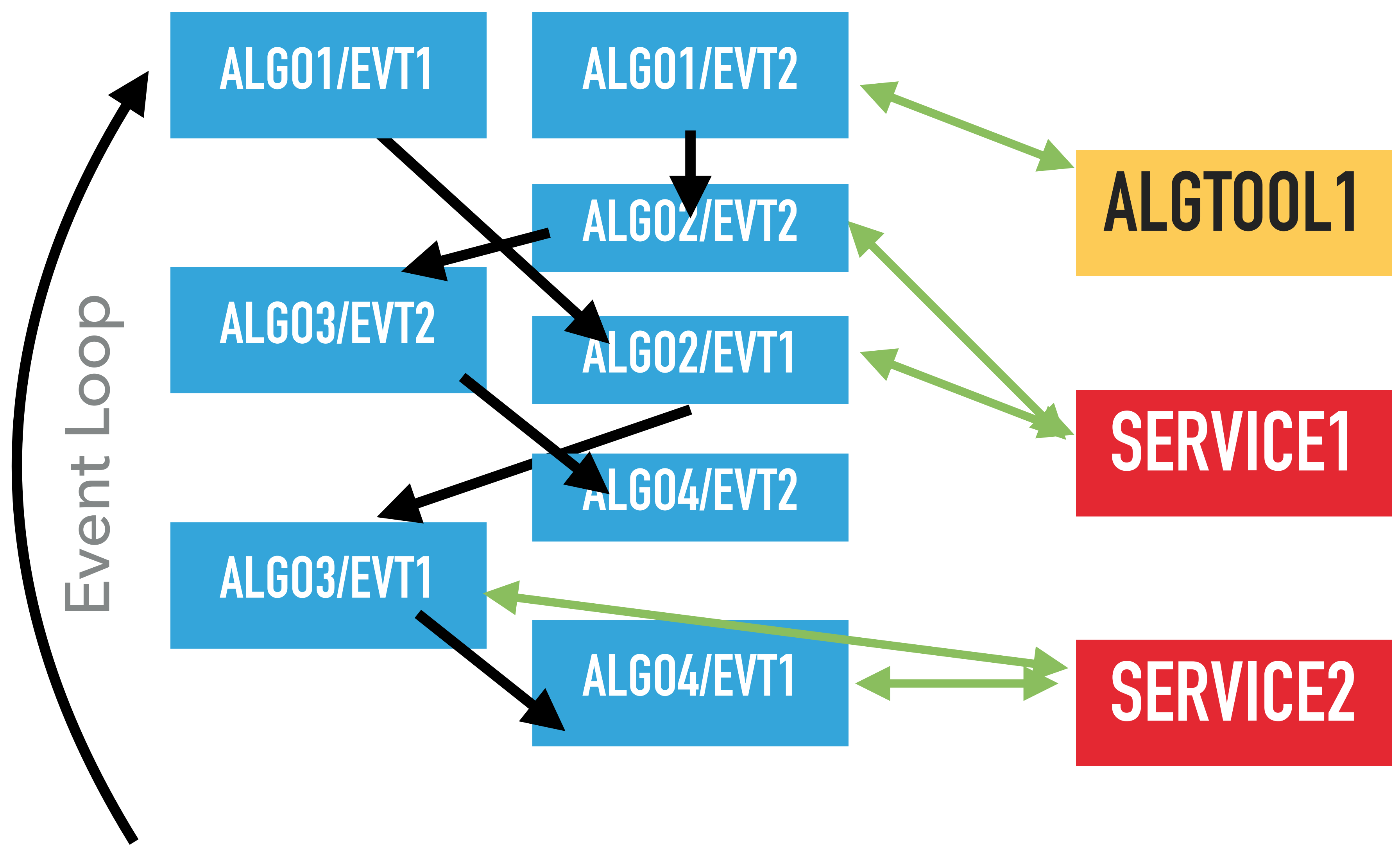
SERIAL ATHENA (RUN1/2):



Dedicated Service to hold event data

Dedicated Algorithms (& services) for I/O

ATHENAMT (RUN 3):

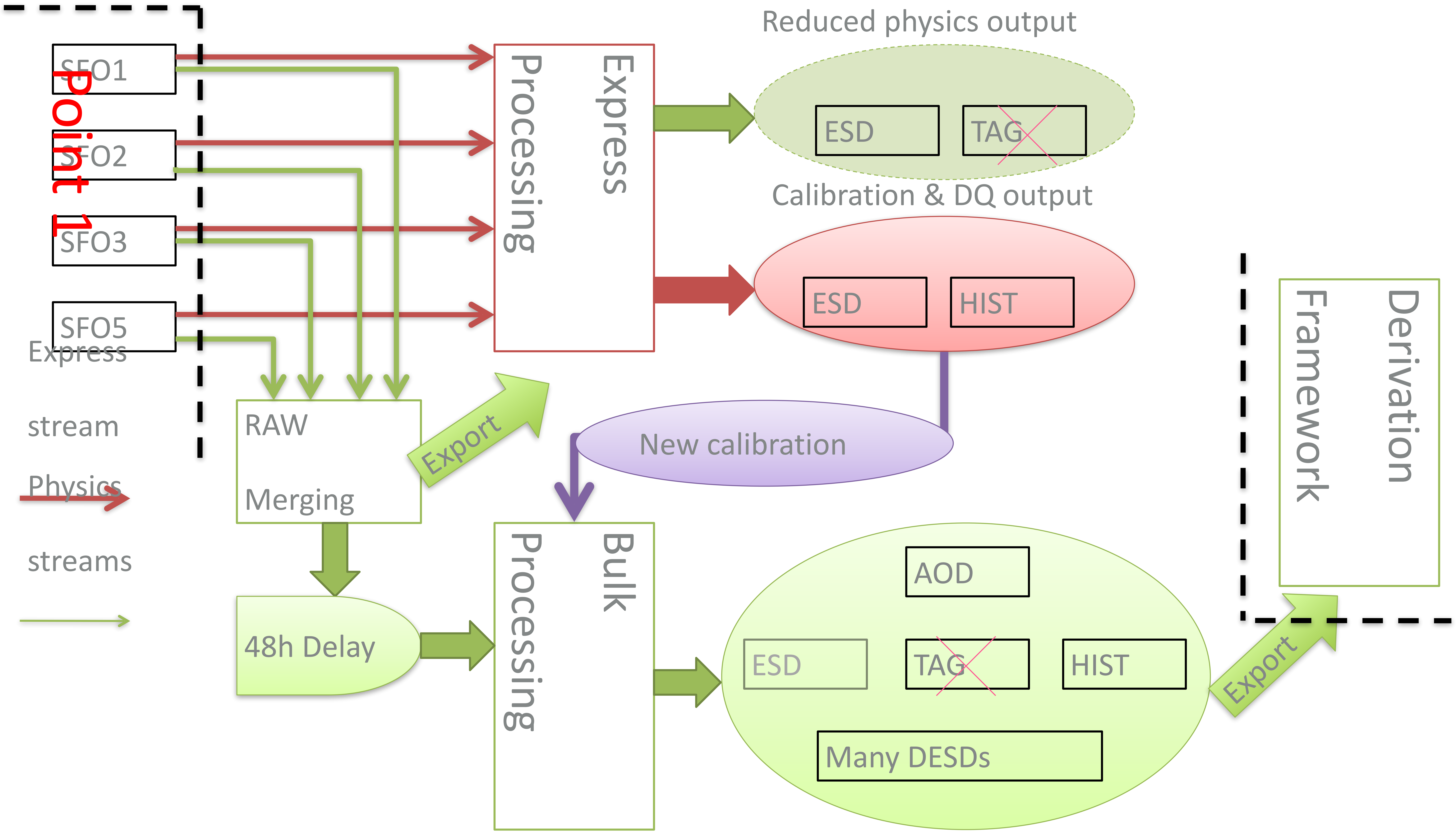


Algorithm execution order determined by a scheduler.

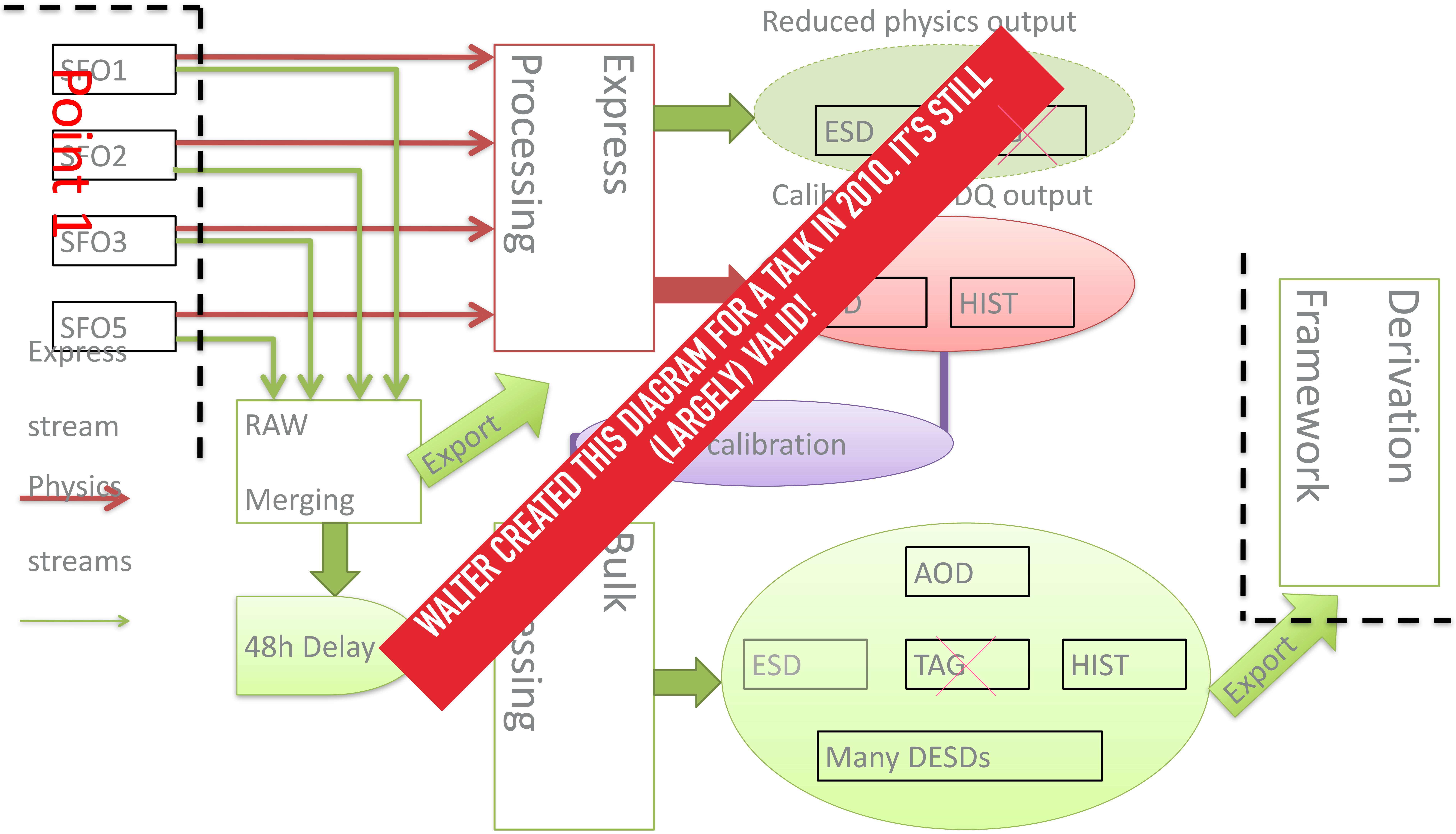
SIDE REMARK:

THE CONCEPTS OF GAUDIHIVE/ATHENAMT HAVE PROVEN VERY USEFUL TO INSULATE JOHN DOE PHYSICIST FROM THE NITTY-GRITTY DETAILS OF THREAD-SAFETY

SIMPLIFIED FLOWCHART WITH CALIBRATION LOOP



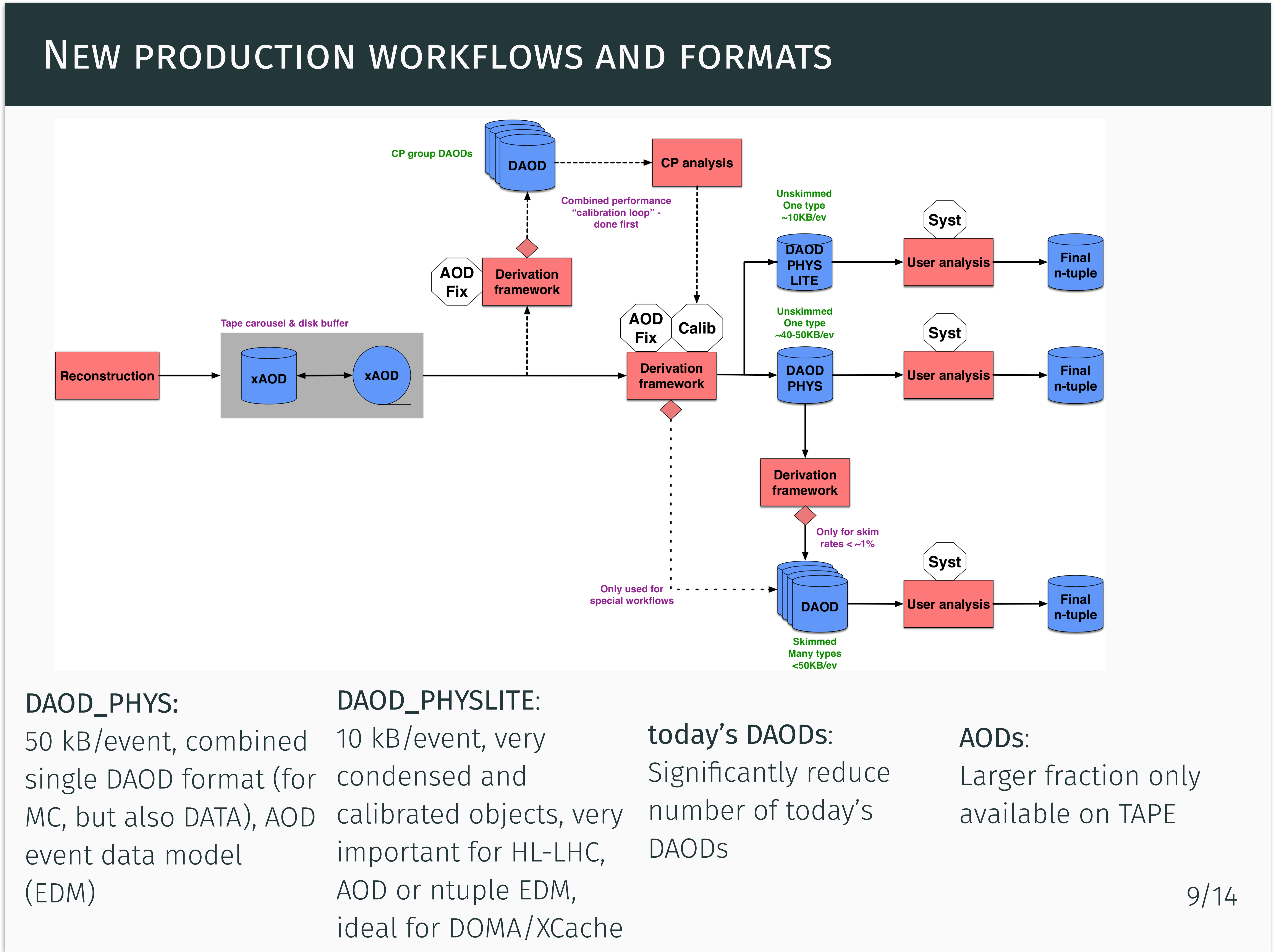
SIMPLIFIED FLOWCHART WITH CALIBRATION LOOP



WALTER CREATED THIS DIAGRAM FOR A TALK IN 2010. IT'S STILL (LARGELY) VALID!

ANALYSIS MODEL

FOR RUN-3, MOVING TO NEW PRODUCTION WORKFLOW AND ANALYSIS DATA FORMATS IN RUN-3



New production workflows and formats

DAOD_PHYS:
50 kB/event, combined single DAOD format (for MC, but also DATA), AOD event data model (EDM)

DAOD_PHYSLITE:
10 kB/event, very condensed and calibrated objects, very important for HL-LHC, AOD or ntuple EDM, ideal for DOMA/XCache

today's DAODs:
Significantly reduce number of today's DAODs

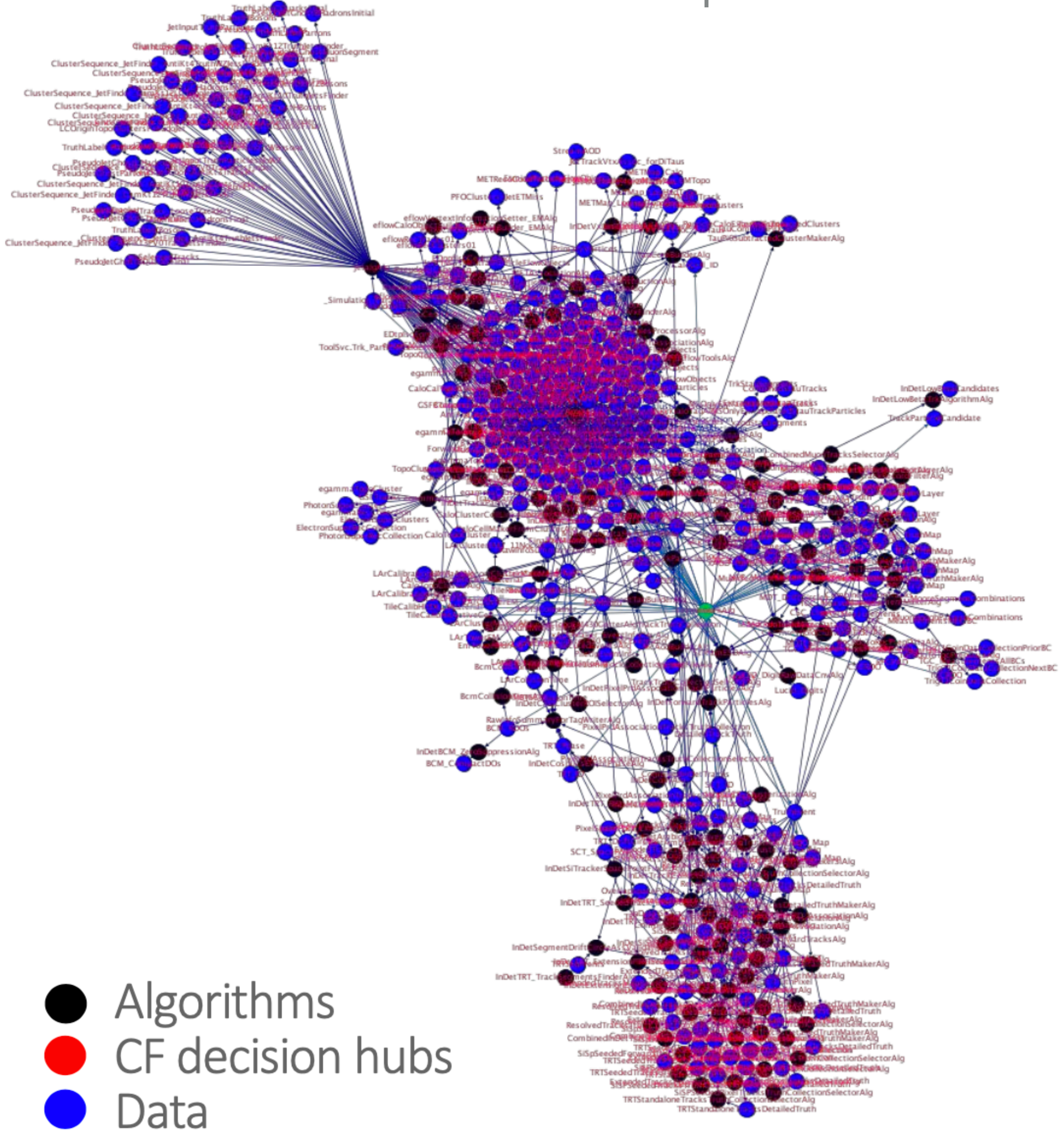
AODs:
Larger fraction only available on TAPE

9/14

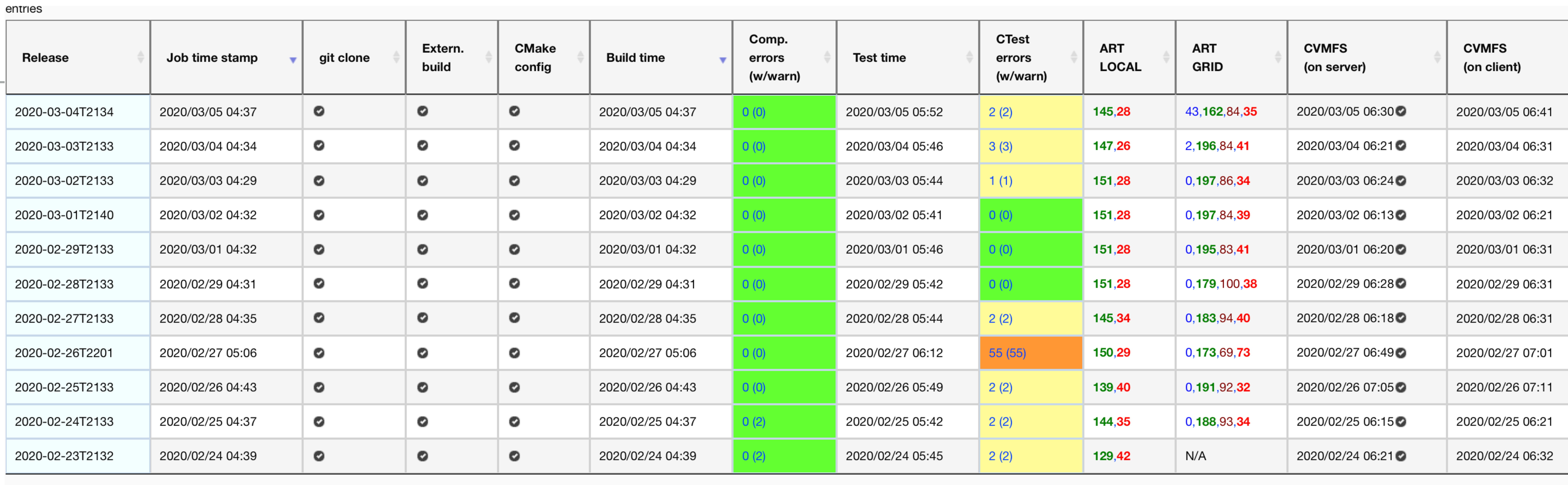
A (LITTLE BIT) MORE ON ATHENAMT

- AthenaMT uses Intel Threaded Building Blocks (TBB) for thread management
  - TBB hidden from developers
- Configuration, Initialization and Finalization are performed serially in the “master” thread
  - Only Algorithm::execute() is concurrent
  - Algorithms are only scheduled when their input data becomes available (rely on DataHandles to express dependencies)
  - Several instances of the same Algorithm can coexist (via cloning)
  - Multiple events can be executed concurrently

Data Flow precedence rule



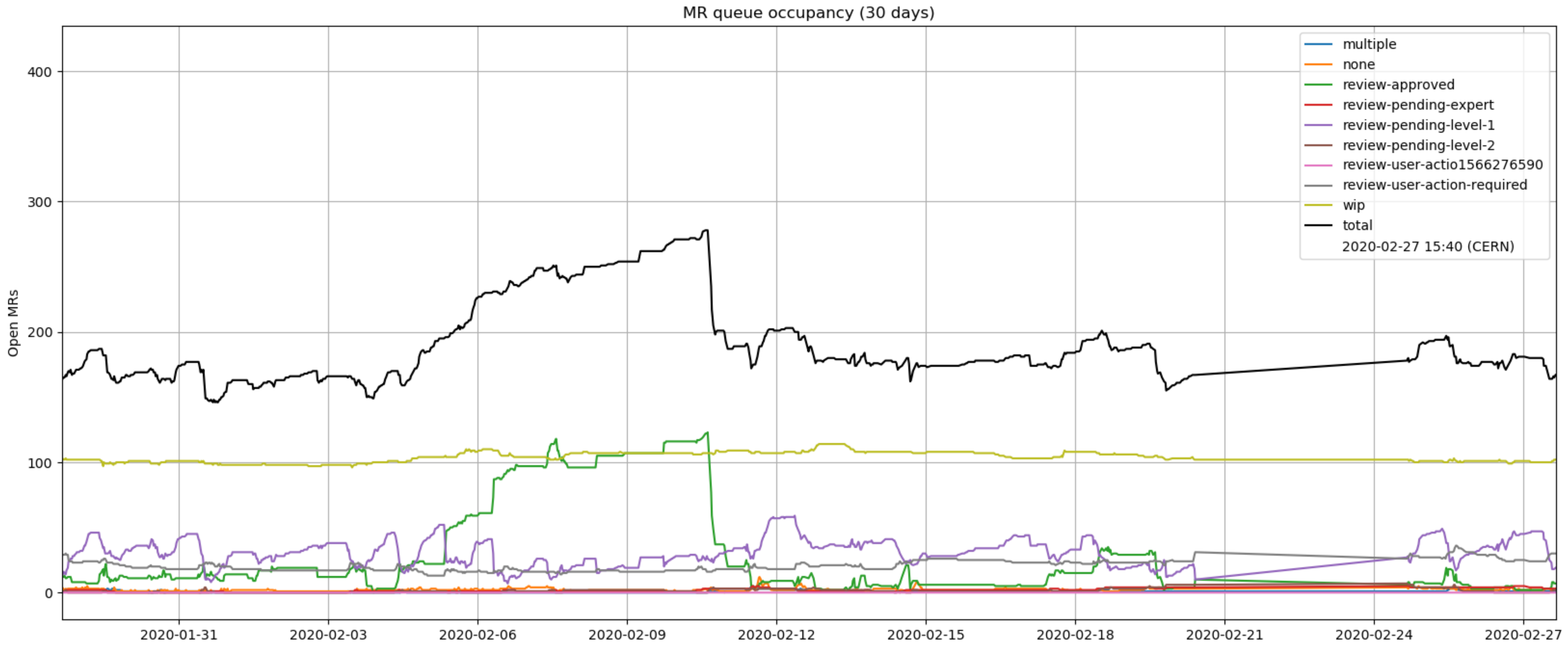
BUILD SYSTEM



entries

Release	Job time stamp	git clone	Extern. build	CMake config	Build time	Comp. errors (w/warn)	Test time	CTest errors (w/warn)	ART LOCAL	ART GRID	CVMFS (on server)	CVMFS (on client)
2020-03-04T2134	2020/03/05 04:37	✓	✓	✓	2020/03/05 04:37	0 (0)	2020/03/05 05:52	2 (2)	145,28	43,162,84,35	2020/03/05 06:30 ✓	2020/03/05 06:41
2020-03-03T2133	2020/03/04 04:34	✓	✓	✓	2020/03/04 04:34	0 (0)	2020/03/04 05:46	3 (3)	147,26	2,196,84,41	2020/03/04 06:21 ✓	2020/03/04 06:31
2020-03-02T2133	2020/03/03 04:29	✓	✓	✓	2020/03/03 04:29	0 (0)	2020/03/03 05:44	1 (1)	151,28	0,197,86,34	2020/03/03 06:24 ✓	2020/03/03 06:32
2020-03-01T2140	2020/03/02 04:32	✓	✓	✓	2020/03/02 04:32	0 (0)	2020/03/02 05:41	0 (0)	151,28	0,197,84,39	2020/03/02 06:13 ✓	2020/03/02 06:21
2020-02-29T2133	2020/03/01 04:32	✓	✓	✓	2020/03/01 04:32	0 (0)	2020/03/01 05:46	0 (0)	151,28	0,195,83,41	2020/03/01 06:20 ✓	2020/03/01 06:31
2020-02-28T2133	2020/02/29 04:31	✓	✓	✓	2020/02/29 04:31	0 (0)	2020/02/29 05:42	0 (0)	151,28	0,179,100,38	2020/02/29 06:28 ✓	2020/02/29 06:31
2020-02-27T2133	2020/02/28 04:35	✓	✓	✓	2020/02/28 04:35	0 (0)	2020/02/28 05:44	2 (2)	145,34	0,183,94,40	2020/02/28 06:18 ✓	2020/02/28 06:31
2020-02-26T2201	2020/02/27 05:06	✓	✓	✓	2020/02/27 05:06	0 (0)	2020/02/27 06:12	55 (55)	150,29	0,173,69,73	2020/02/27 06:49 ✓	2020/02/27 07:01
2020-02-25T2133	2020/02/26 04:43	✓	✓	✓	2020/02/26 04:43	0 (0)	2020/02/26 05:49	2 (2)	139,40	0,191,92,32	2020/02/26 07:05 ✓	2020/02/26 07:11
2020-02-24T2133	2020/02/25 04:37	✓	✓	✓	2020/02/25 04:37	0 (2)	2020/02/25 05:42	2 (2)	144,35	0,188,93,34	2020/02/25 06:15 ✓	2020/02/25 06:21
2020-02-23T2132	2020/02/24 04:39	✓	✓	✓	2020/02/24 04:39	0 (2)	2020/02/24 05:45	2 (2)	129,42	N/A	2020/02/24 06:21 ✓	2020/02/24 06:32

- We current build ~20 branches per night
    - Run unit tests, local longer tests, and grid-based large statistics test
- Run CI on every merge request:
    - currently using Jenkins, but investigating moving to GitlabCI


MR queue occupancy (30 days)
Open MRs
multiple
none
review-approved
review-pending-expert
review-pending-level-1
review-pending-level-2
review-user-actio1566276590
review-user-action-required
wip
total
2020-02-27 15:40 (CERN)