**Motivation**

- AthenaMT is the ATLAS software framework used in trigger, reconstruction, simulation, and analysis.
- AthenaMT is based on GaudeLite, a core framework shared with LHCb.
- The computing market transitioned towards many-core CPUs while memory price plateaued, leading to less memory/ core.
- AthenaMT was already in use in Run 2, and ATLAS struggled to use the processing resources (WLCG, Tier0) efficiently with AthenaMT.
- A stopgap solution was to force buffering to reduce memory per process (thanks to copy-on-write).
- Ultimate solution: redesign the core framework for native, efficient, and user-friendly multi-threading support → AthenaMT.

**Implementation**

- Based on GaudeLite which uses Intel TBB.
- A both inter-event and intra-event concurrency is achieved.
- Defines an execution order based on data dependencies declared in ReadHandles and WriteHandles.
- Decides which to execute an algorithm based on input/output and the configured number of threads and event slots.
- When input dependencies are met, Scheduler pushes the algorithm into an Intel TBB queue for execution.
- AthenaMT design encompasses the HLT requirements from the beginning, e.g., support for partial event data processing.

**High-Level Trigger in AthenaMT**

- Taking the opportunity of AthenaMT migration to rewrite the HLT framework.
- Run-2 HLT framework used a dedicated top-level algorithm taking care of algorithm scheduling.
- HLT in AthenaMT is closer to the offline reconstruction framework – using the GaudeLite Scheduler and removing the trigger-specific layer allows to use offline algorithms directly in HLT without wrappers.
- Processing of partial event data (regional reconstruction) is integrated in GaudeLite Event Views – algorithms can use partial or full data as input without any modification.
- HLT Central Flow configures an execution graph including Event Views preparation (Input Maker) and early termination of an execution path if trigger not accepted (Filter Step).
- Each HLT chain corresponds to an execution path through the CF graph.

**Data flow**

- Hardware Level-1 Trigger reduces the rate to 100 kHz.
- Software HLT is required to provide further reduction to ~1.5 kHz.
- Run-2 HLT consisted of ~40k physical cores and typically ran one instance of AthenaMT per core.
- Run-3 HLT infrastructure will remain similar to Run-2.

**Operating AthenaMT within TDAQ**

- The online-specific layer implements additional requirements for data-taking operation and integration with the TDAQ system.
- Reading/writing ROD files replaced with an interface to TDAQ applications (DataCollector).
- Extended error handling to prevent application exit where possible – seed erroneous events to a special data stream (debug stream) for later investigation and recovery into physics streams.
- Additional thread to monitor event processing time and interrupt timed-out events.
- Multi-threading brings new crash debugging challenges.
- Cannot determine which concurrently processed event crashed the application – send all to the debug stream and investigate all of them offline.
- More concurrent events = more good events in the debug stream in case of a crash.
- Event order depends on the machine performance – possible irreproducibility of problems.
- Performance measurements will be needed to determine the optimal number of forks, threads, and slots.