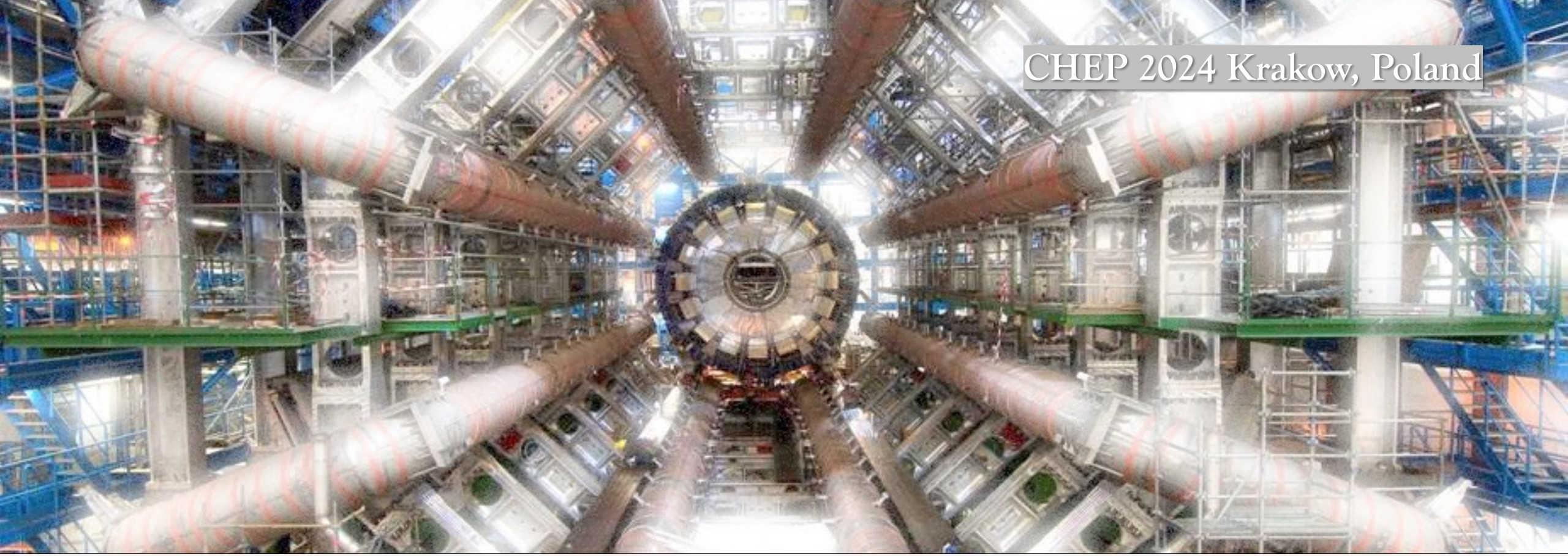


Evolution of the ATLAS TDAQ online software framework towards Phase-II upgrade: use of Kubernetes as an orchestrator of the ATLAS Event Filter computing farm

Alina Corso Radu (University of California Irvine)

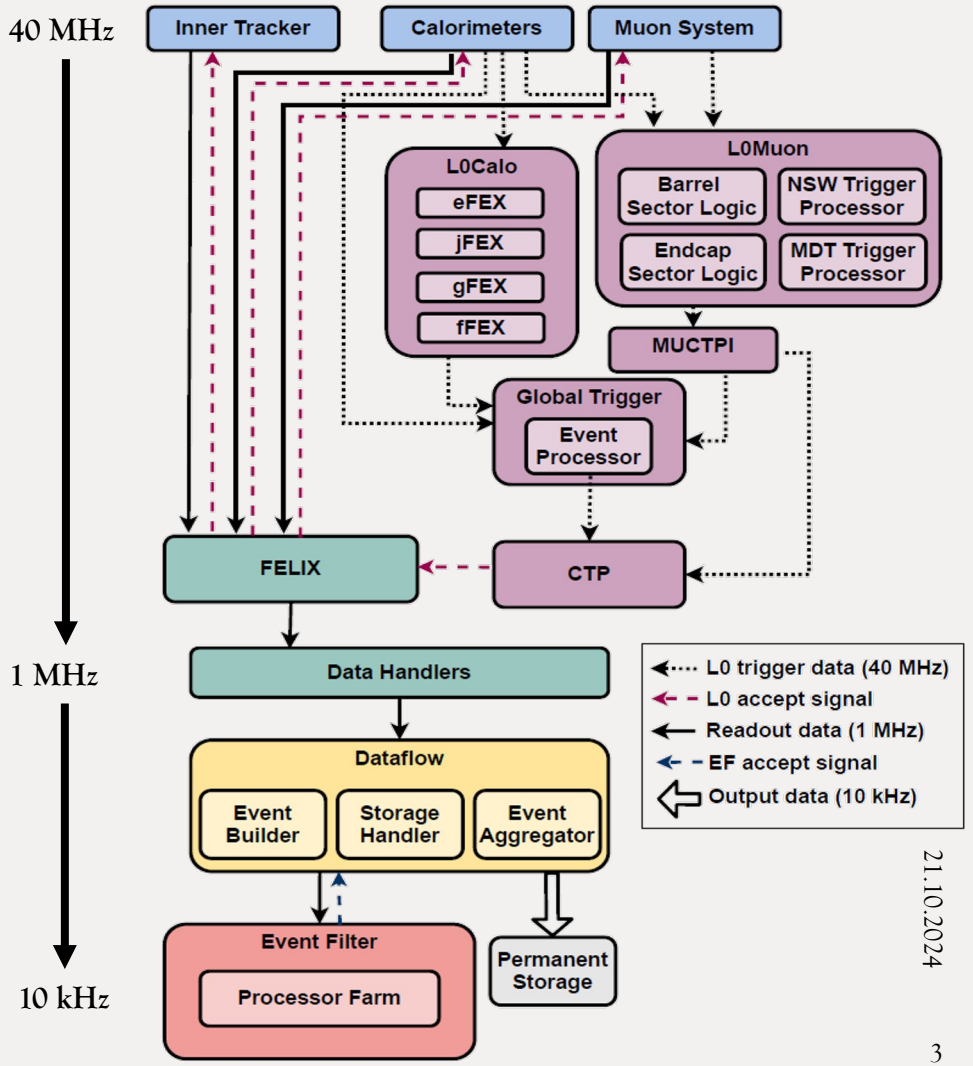
on behalf of ATLAS TDAQ Control & Configuration group

DATA ACQUISITION CHALLENGES FOR PHASE-II UPGRADE

- HL-LHC will provide higher luminosity 7.5 x 10^{34} cm^{-2} s^{-1} with more than 200 interactions per bunch crossing pushing the limits of the current TDAQ system
- More data to deal with:
  - *Increase trigger rates to better exploit full HL-LHC physics*
  - *DAQ throughput increase because of larger event size*
- Aim to increase trigger selectivity using lower thresholds, but with longer processing time, due to event complexity
  - *Longer hardware trigger latency ($\Longrightarrow$ new Readout/DAQ architecture)*
  - *Software trigger more robust against pile-up*
- Need for enhanced scalability, fault tolerance, and more efficient resource utilization

ATLAS TDAQ ARCHITECTURE FOR PHASE-II

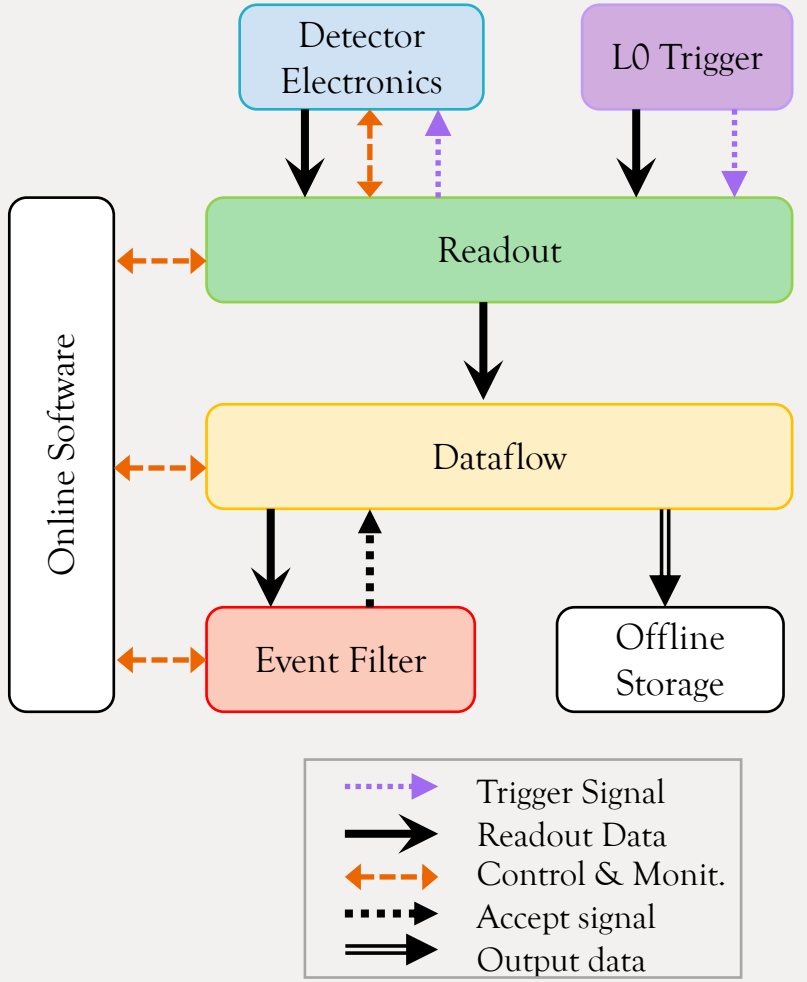
- Hardware based L0 trigger
- Software based Event Filter (EF) trigger
- **New trigger and increased readout capabilities:**
  - *L0 trigger data reduction: 40 MHz →1 MHz (100 kHz in Run 3)*
  - *L0 latency 10 μs (2.5 μs in Run3)*
  - *Full Event Building at 1 MHz L0A rate*
  - *Event size of 4.6MB (1.5MB in Run 3)*
  - *EF data reduction: 1 MHz →10 kHz (3 kHz in Run 3)*



TDAQ Phase-II Upgrade TDR

ONLINE SOFTWARE

- Manages and operates the components of the data acquisition system integrating them with the wider ATLAS data taking environment
- Configures, controls and monitors large distributed system
  - *The EF farm only will consist of about 5000 computing nodes (less, depending on technology - GPU)*
- Current EF farm management is based on in-house process management and is statically configured
  - *Need to reduce maintenance, improve resource allocation, monitoring, scalability and reliability to cope with HL-LHC challenges*



EF FARM ORCHESTRATOR REQUIREMENTS

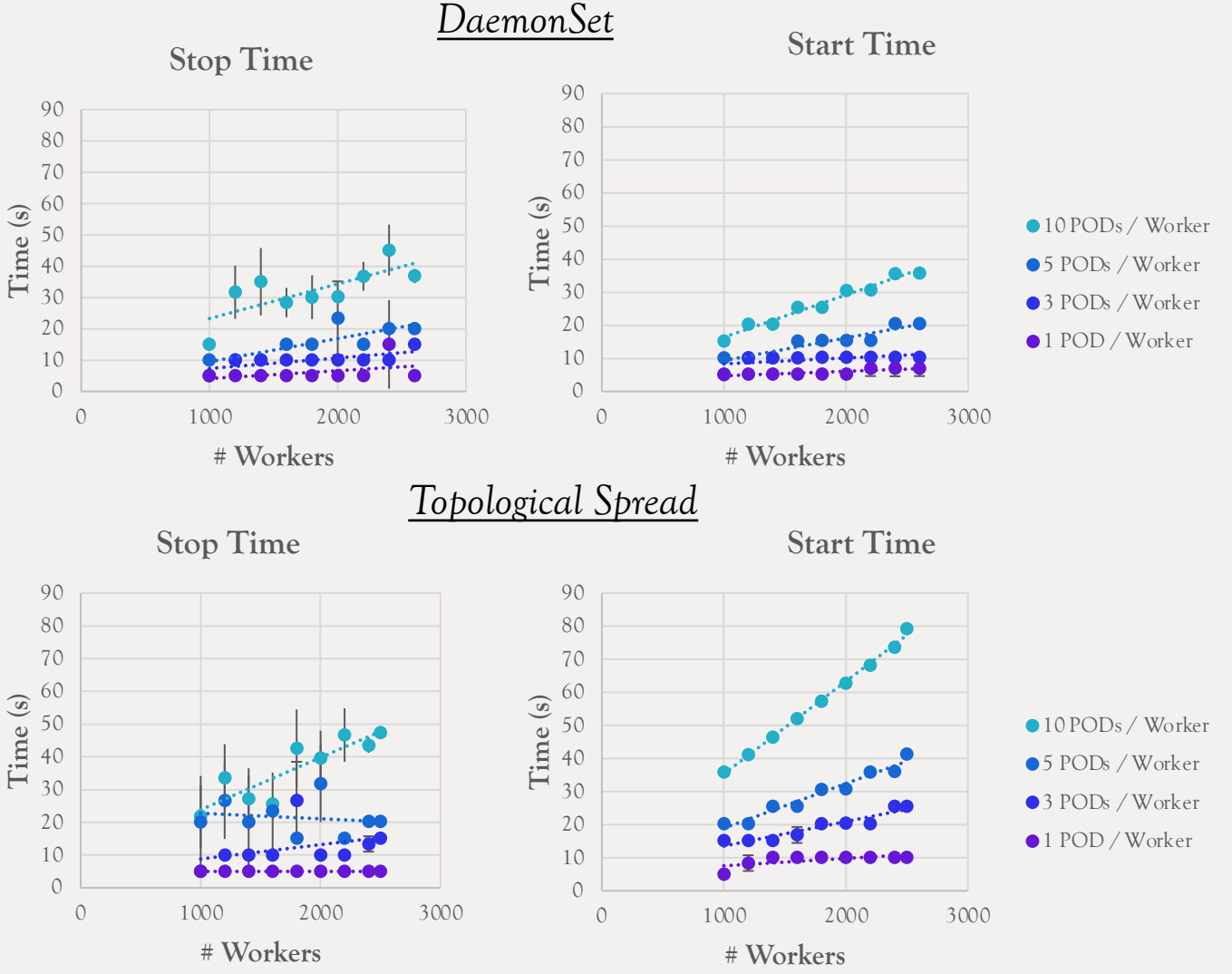
- A robust and reliable mechanism for the management of all processes running in the EF farm to guarantee stable and effective execution of physics events processing
- The mechanism should also facilitate the seamless coexistence of two workloads — physics data taking and simulation production — allowing them to effectively share the available computing resources
- The system should:
  - *support multiple application lifecycle types, including continuous operation, completion-based execution, and scheduled (cron-like) services*
  - *allow both dynamic and static assignment of processes to computing nodes*
  - *dynamically manage cluster resources, allowing for runtime activation or deactivation of computational units and optimizing CPU and memory usage*
  - *scale to accommodate thousands of hosts*
  - *provide capabilities to control (start/stop) and monitor the status of all active processes*
  - *fully support the specification of requirements for each process to be launched, including command-line parameters and environment variables to be passed to the executable*

KUBERNETES AS EF FARM ORCHESTRATOR

- Why Kubernetes?
  - *Dynamically allocates computing resources based on real-time conditions*
  - *Ensures high availability and fault tolerance of data processing services*
  - *Advanced resource utilization and monitoring*
  - *Scalability in response to changing data-taking conditions*
  - *Simplified deployment and management*
  - *Support for containerized applications*
  - *Open-source solution that aligns with industry best practices in container orchestration*
- Kubernetes has been confirmed as the official solution for orchestrating the TDAQ Event Filter farm following the successful development, extensive large-scale testing, and review of a prototype that proved its viability

TESTING KUBERNETES IN TDAQ FARM

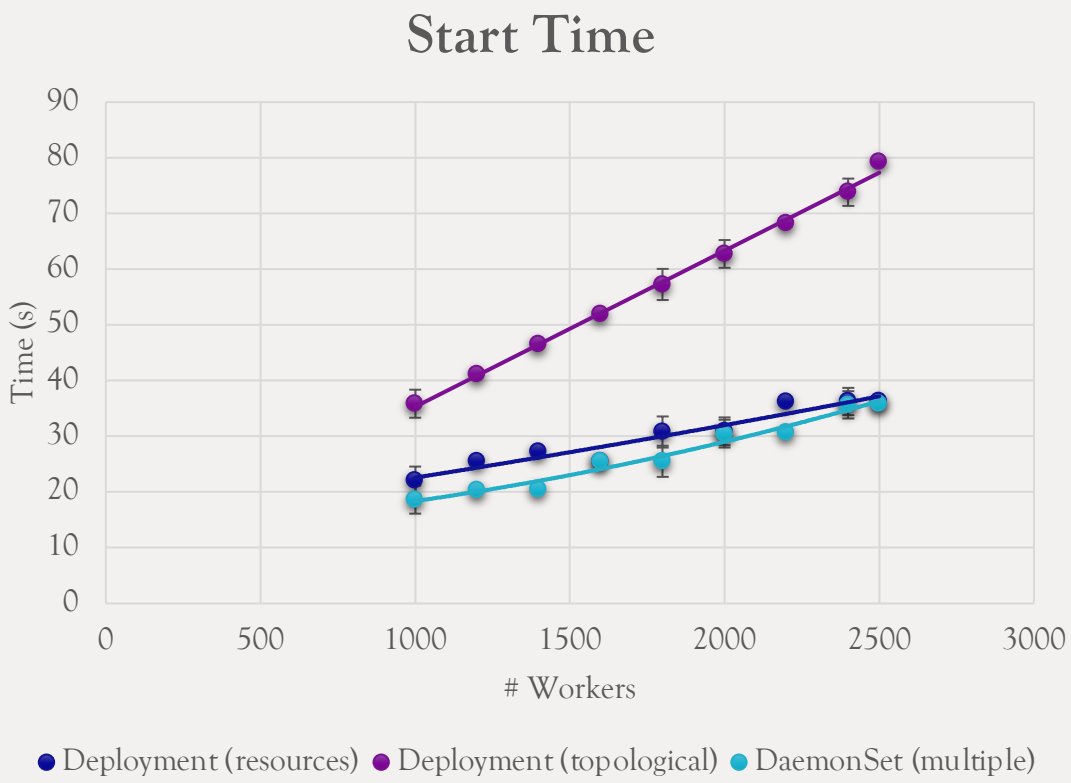
- Kubernetes clusters installed in TDAQ lab and experiment's closed network (P1)
  - *P1 cluster (close to production needs): 4 control planes nodes, 4 ETCD, 2 Prometheus, 2600 workers*
- Several testing sessions were conducted using the Run3 ATLAS EF farm of ~2600 servers
- Tests evaluated scalability, fault tolerance, and resource management with different scheduling strategies (reaching the same goal)
- Focusing on PODs start/stop timing studies using different scheduling strategies – no time constraints for EF farm restart



POD - smallest unit to be deployed by Kubernetes

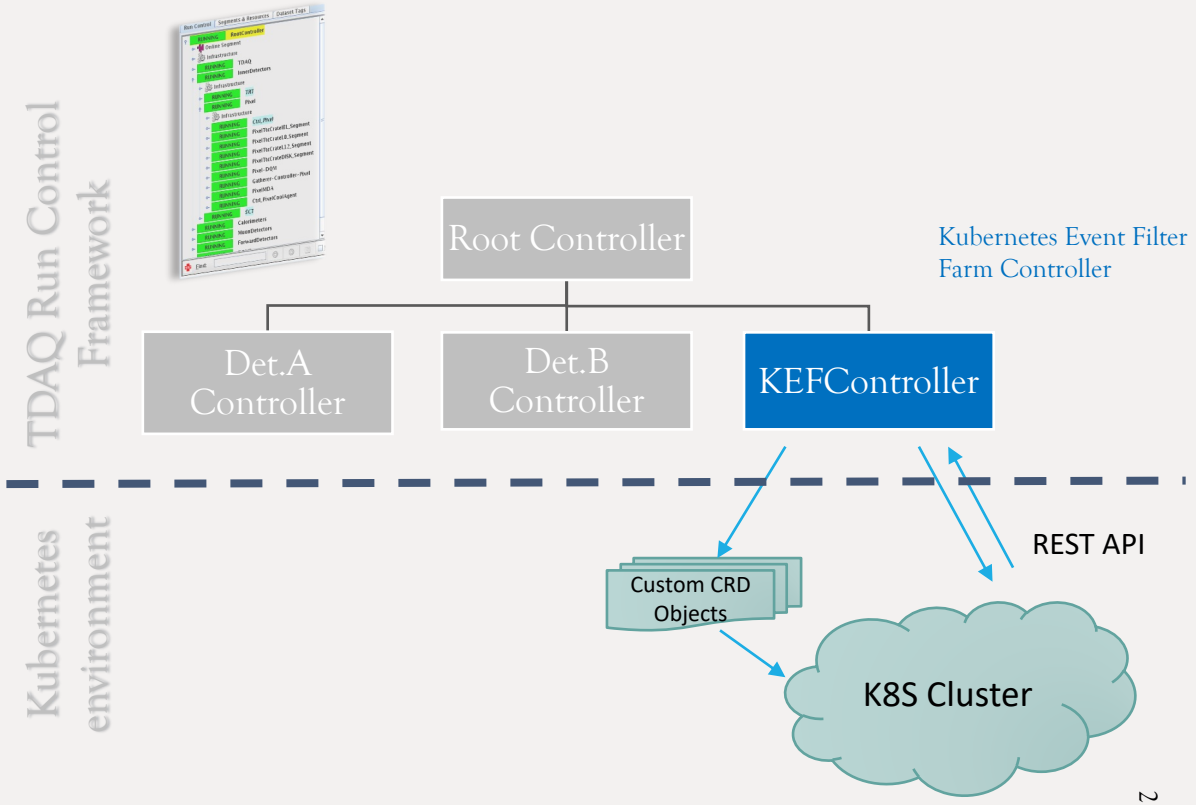
COMPARING SCHEDULING STRATEGIES

- Comparing different scheduling start time for 10 PODs per worker node
- Topological spread: distributes PODs evenly across nodes for better load balancing and fault tolerance
- DaemonSet: ensures a POD runs on every node in the cluster
- Resource-based: focuses only on node resource availability (e.g., CPU, memory)
- Topological spread is better for balanced, scalable, and resource-efficient deployments, but is more costly in terms of scheduling time



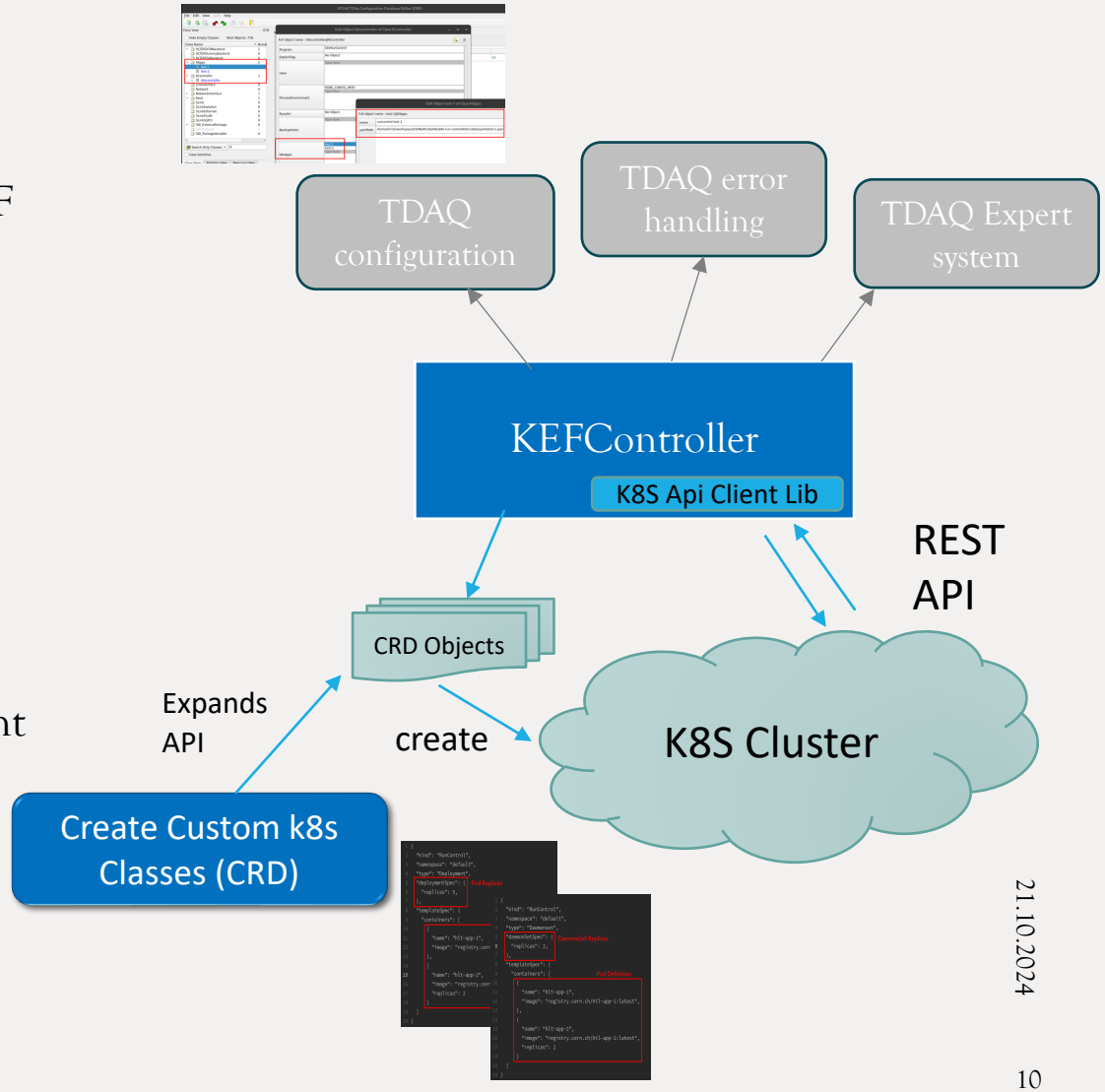
INTEGRATION WITH THE TDAQ RUN CONTROL FRAMEWORK

- Tight synchronization between data acquisition components and computing infrastructure is crucial for the correct operation of the ATLAS detector
- TDAQ Run Control leverages a Finite State Machine (FSM) model to manage the states of data acquisition components across thousands of servers, ensuring synchronized data processing and experiment control
  - *Kubernetes integration respects the FSM allowing for seamless state management across the TDAQ components and Kubernetes-based resources*



KUBERNETES-TDAQ 'BRIDGE' APPLICATION

- Uses Custom Resource Definition (CRD) to manage and deploy EF applications
  - *Different deployment scenarios are possible*
- Implements Kubernetes operator pattern using in-house C++ wrapper on top of Kubernetes-c library to interact with the cluster
- Ensures Kubernetes-related tasks are coordinated throughout the lifecycle of a run
  - *Application implements the DAQ FSM*
- Gathers information about the EF applications needed environment
- Uses standard TDAQ services/tools
  - *Configuration using TDAQ configuration OKS*
  - *Implements error handling using TDAQ ERS*
  - *Monitors the health of the cluster and feeds-back the information to the TDAQ Expert System (CHIP)*



EF DEPLOYMENT SCENARIOS UNDER CONSIDERATION

- Lessons learned

  POD - smallest unit to be deployed by Kubernetes

  - *Balancing maintainability and performance is crucial*
  - *Resource requests must be accurately defined to avoid scheduling issues*
- One container/POD can be costly in terms of scheduling given the estimated large number of EF processing units needed (estimated 60k)
- Single container/POD for Dataflow services (DF) and multiple containers/POD for EF processing
  - *DF POD deployed as DaemonSet, EF POD use Kubernetes custom and fine-tunning resource allocation for efficient scheduling*
- 'All in one' POD approach:
  - *Combining DF and EF containers in a single POD simplifies deployment*
  - *Reduces the number of PODs, lowering scheduling overhead and system complexity*
- Considerations and Constraints:
  - *Assuming DF and EF applications have synchronized lifecycles, limiting independent scaling*
  - *Manual interventions can only be done at the POD level – problematic control in case of multiple containers per POD*
  - *Control at container level possible implementing health checks*
  - *Resource and health management remain critical for optimal performance*

FUTURE DIRECTIONS AND INDUSTRY ALIGNMENT

- The adoption of Kubernetes aligns ATLAS TDAQ with industry best practices in container orchestration and cloud-native computing
- Continuous improvements in automation, monitoring, and scalability
- Potential for further integration with emerging technologies like advanced monitoring and machine learning for data processing optimization

CONCLUSION

- The integration of Kubernetes into the ATLAS TDAQ system marks a significant leap forward in preparation for HL-LHC operation
- Some of the key benefits of using Kubernetes as EF Farm orchestrator:
  - ***Scalability:*** *Ability to handle increasing data volume efficiently*
  - ***Fault Tolerance:*** *Improved system reliability and minimal downtime*
  - ***Resource Optimization:*** *Dynamic resource allocation and better utilization of computing power*
  - ***Simplified Workflow Management:*** *Easier deployment and maintenance of the system*
  - ***Resources Sharing****: Manage physics data taking and simulation production workflows to share same computing resources*
- Ongoing testing under "close to real" data-taking conditions enabled by the Kubernetes cluster installed at ATLAS P1
- Results demonstrated good performance and robustness in handling the complex data-taking environment of ATLAS
- ATLAS is now better equipped to meet the challenges of higher luminosity and increased data volume