

DCS Data Tools

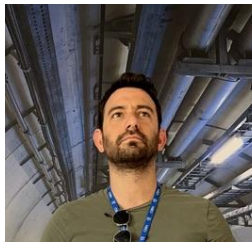
PostgreSQL/TimescaleDB Implementation for
ATLAS DCS Time-Series Data

P. Moschovakos, D. Matakias

Meet the team

P. Moschovakos

*ATLAS DCS Group Leader Member of the C++ Standardization Committee
12 years of experience in software development for CERN industrial control systems (WinCC OA, OPC UA)*



D. Matakias

*Hybrid hardware - software engineer focused on database for ATLAS DCS
6 years of experience in ASIC testing, electronics integration, and test beams*



Our collaborators

CERN DBOD team



ATLAS DBAs

CERN Openstack



BE ICS group



Agenda

- Introduction to ATLAS DCS and the Challenge
- DCS Data Tools Solution - architecture & implementation
- Grafana integration - visualization & monitoring
- Production impact, current usage, results and user adoption
- Conclusions & Takeaways

About ATLAS DCS

ATLAS DCS

- ATLAS - largest of the four major experiments at CERN's LHC
- DCS: Detector control system, manages operations and monitoring & ensures safety

The scale

- 68 TB of time-series data accumulated since 2007
- 18 years of historical data stored in Oracle production database (ATONR)
- Critical for detector monitoring, analysis, and operations



ATLAS control room

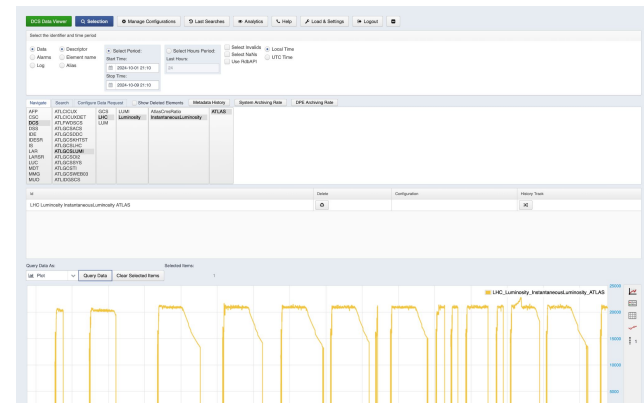
The challenge & requirements

Legacy workflow bottlenecks

- DCS Data Viewer or custom scripts accessing Oracle DB directly
- Complex data access requiring SQL expertise
- Limited accessibility for non-expert users & slow performance when not properly optimized
- Difficult integration with modern visualization tools

Community Survey Results

- 86% preferred Grafana for visualization
- 92% required flexible search functionality
- 90%+ demanded continuous availability with full historical access



DCS Data Viewer - custom web app

The DCS Data Tools solution approach

Based on systematic community survey across ATLAS experiment, focusing on real user needs and workflows

Fully compatible with NextGen Archiver developed by BE-ICS

Core Objectives

Performance & accessibility

- Target: >10× improvement over legacy workflows
- Enable multi-year analysis in seconds, not minutes
- Lower entry barriers through simple APIs

Scalable, Future-Proof Architecture

- Aligned with NextGen Archiver (NGA)
- Handle growing data volumes (9 TB/year)

Foundation: PostgreSQL + TimescaleDB

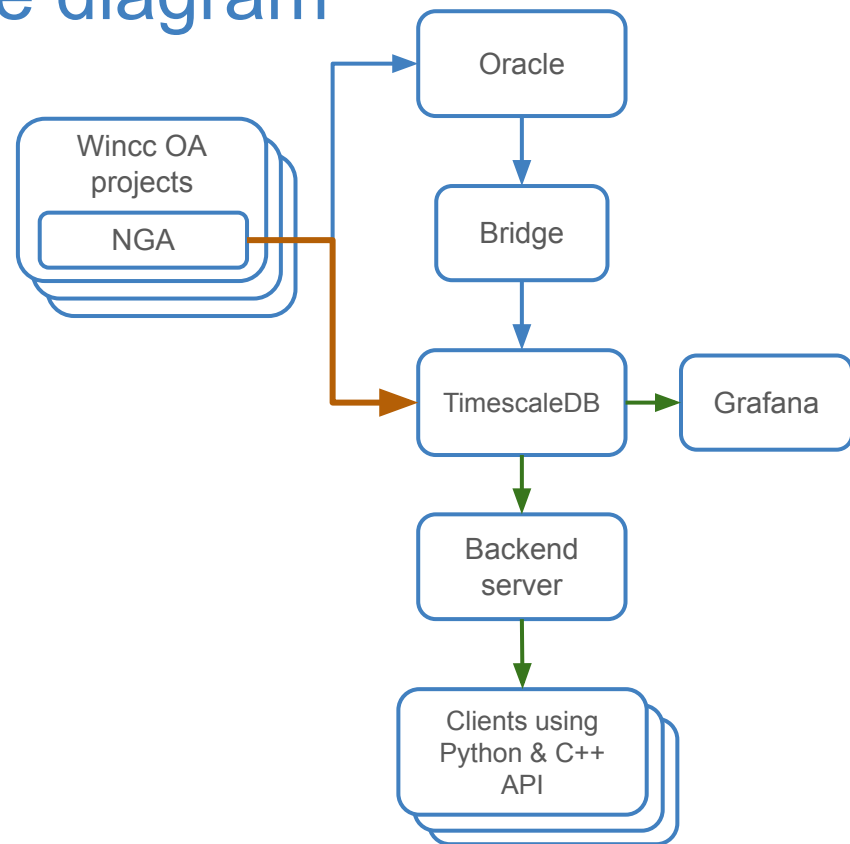
Solution Overview - architecture diagram

Design Principles

- Backward compatibility maintained with Oracle/WinCC OA
- Database-agnostic design enabling multiple data sources
- Modular architecture for independent component upgrades

Key Components

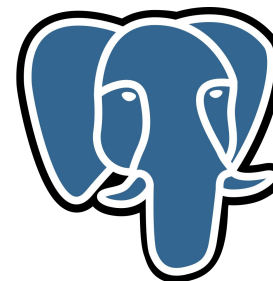
- Oracle Bridge: Transitional data extraction layer
- TimescaleDB: Time-series optimized storage
- Backend Server: Database-agnostic middleware
- Client APIs: Python and C++ interfaces
- Grafana: Visualization and dashboarding platform



Why PostgreSQL + TimescaleDB

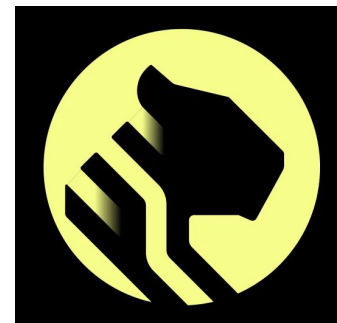
PostgreSQL: Foundation

- Open-source ecosystem with strong community support
- Long-term sustainability and no vendor lock-in
- Enterprise-grade reliability and maturity
- Fallback migration path if needed
- Chosen by Wincc OA as the new default DB for archiving



TimescaleDB: Time-Series Optimization

- Native time-series features and hypertables
- Built on PostgreSQL (inherits all benefits)
- Automatic data partitioning and management
- ~11× consistent compression ratio (68 TB → 6.6 TB)
- No licensing costs for large-scale deployment
- **Significant development and support from JCOP**



Infrastructure - CERN DBOD & Hardware

CERN Database-on-Demand (DBOD) Service

- Enterprise-grade reliability and high availability
- Expert support from CERN IT team
- PostgreSQL & TimescaleDB centrally supported
- Simplified operations and maintenance

Development to Production Pipeline

- Development environment: CERN OpenStack
- Production database: CERN DBOD
- Grafana hosting: CERN OpenShift platform
 - Planning to migrate to Grafana Monit



CERN datacenter

Backend Server - The middleware layer

Database-Agnostic Design

- Supports multiple database types concurrently
- Oracle & TimescaleDB simultaneously
- Flexibility for future technology changes

Core capabilities

- Abstracts SQL complexity from end typical users
- Connection pooling and management
- Detailed logging for debugging and monitoring
- Allows data access without users connecting directly to DB

Lightweight Client APIs

- Minimal dependencies for Python and C++ clients
- Server handles complexity, clients stay simple

TimescaleDB schema & optimization

Simplified schema

- Schema design and key technical decisions derive from BE-ICS studies. Following schema to ensure smooth transition to direct archiving to TimescaleDB
- Cleaner than Oracle implementation
- Taking advantage of TimescaleDB features - data partitioning, hypertables with composite indexing
- Easier maintenance and better performance potential

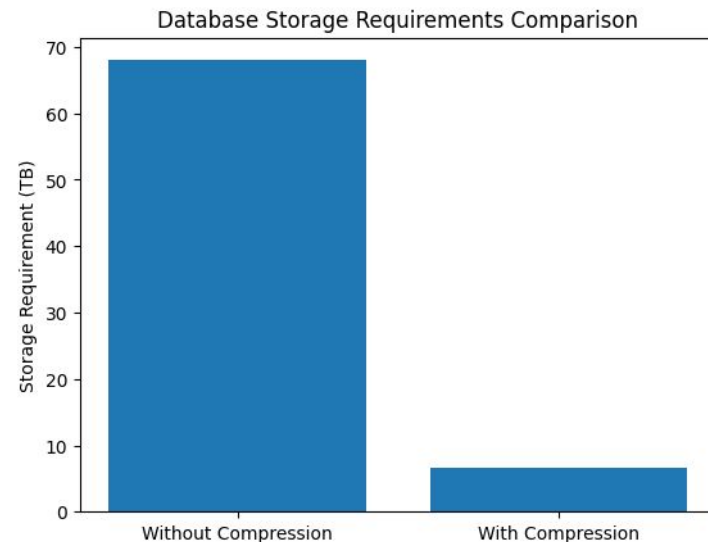
Critical Optimizations

- Currently using 4-day chunk intervals - good tradeoff for multi-year queries & reduced overhead from chunk management
- Tuning & settings chosen in collaboration with CERN DBOD team
- Planning to implement continuous aggregates - working in close collaboration with BE-ICS

TimescaleDB compression in our workloads

Compression metrics

- Consistent $\sim 11\times$ compression ratio on our production data
- Native TimescaleDB columnar compression algorithm
- Compression significantly increases read performance ($3\times$ - $10\times$) in our typical workloads (analytical range queries) and reduces disk usage
- **Read replicas on consumer hardware made possible**
- Simple SQL policies manage compression transparently
- Inserts, updates, and deletes are supported on compressed data



Oracle Bridge - transitional architecture

Risk-Free Development

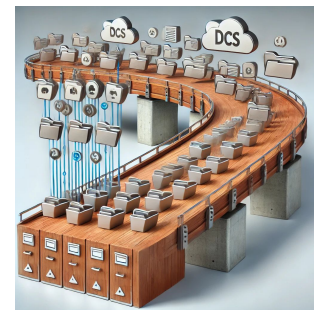
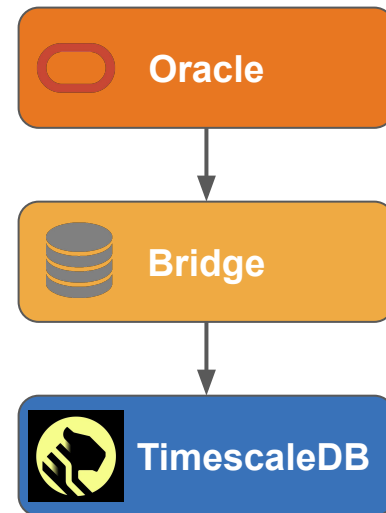
- Enables TimescaleDB development without WinCC OA changes & production data access
- Temporary solution before NGA is production ready

Technical Implementation

- Extracts from Oracle ATONR_ADG (read-only mirror)
- Two-pass real-time data capture: 2-min extraction cycle for recent data
- Hourly late-data collection for delayed arrivals and delays on read-only mirror
- \approx 1-minute average latency maintained

Scalable Performance

- Up to 600.000 rows/s
- Full historical migration completed in \sim 4 weeks



Grafana integration

Infrastructure

- Currently hosted on CERN OpenShift platform.
- Centrally managed by CERN IT
- Planning to migrate to Grafana Monit as long term solution

Features

- SSO authentication
- Available over the internet
- Role-based access control
- Personal, group and public folders configured by CERN e-groups
- Automated backup / restore functionality

Currently over 100 active users

Grafana features & templates

Predefined Query Templates

- Common time-series plotting patterns
- Standardized visualizations for detector monitoring
- Rapid dashboard creation (minutes, not hours)

Critical features from user's perspective

- Real-time data with ~1 min latency
- Full 18 years of historical data accessible
- Dashboard sharing across teams & community
- Seamless access over internet and inside CERN
- Excellent performance and customization compared to previous workflows

Planning to integrate SQL based API in Grafana, designed by BE-ICS group to allow more features, standardization and better performance

Real Use Case - MDT Gas System Monitoring

- Replaced existing workflow based on batch processing with delays & custom scripts
- 80+ panels monitored in real-time
- Multi-parameter correlation views

Impact

- Quick and easy access to data
- Enhanced operator situational awareness
- Template for other subsystem monitoring

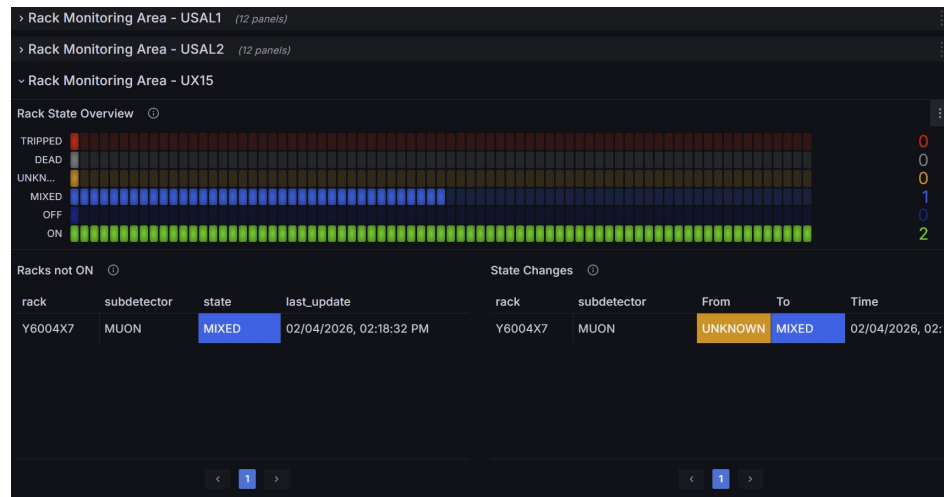


Real Use Case - OPM monitoring

Multiple panels for infrastructure - racks, magnets, cooling & ventilation

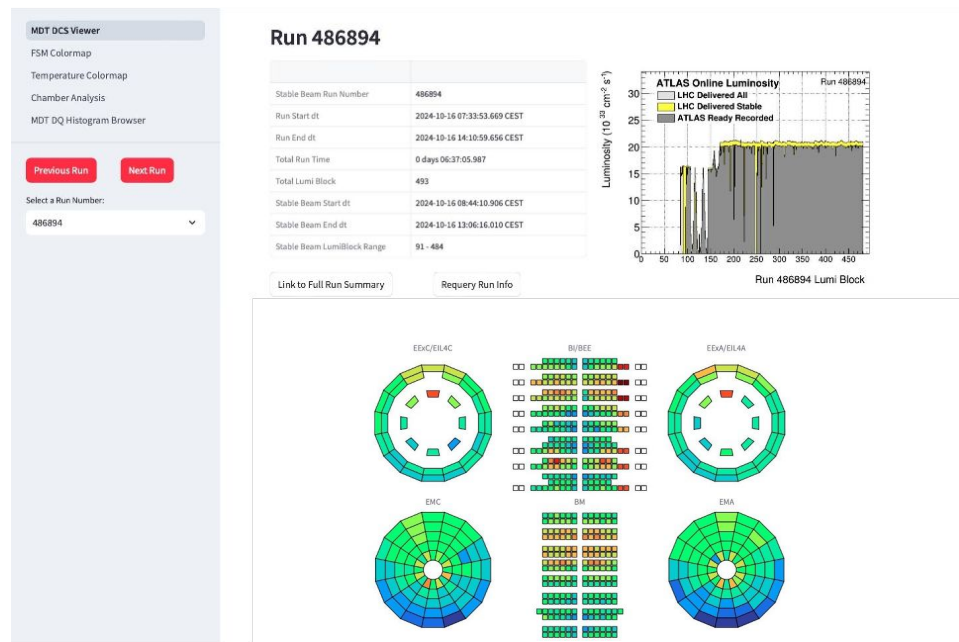
Impact

- Added comprehensive and detailed views of several systems managed by OPM
- Real time access to data
- Previous workflow required accessing the DCS FSM - slow and cumbersome
- Allows easy historical view of the status and makes debugging much easier



Real Use Case - MDT web application

- Dynamic web application built on top of DCS Data Tools API
- Quick and easy access to specific data
- Tool for detector operations - aims to provide critical information to the relevant experts
- <https://mdt-dcs-offline-viewer.app.cern.ch/>



Production readiness & availability

First production version released to CERN community

- Serving active users across ATLAS experiment
- Continuous availability (scheduled maintenance only)
- No disruption to existing integrations
- Stable API interface maintained
- Ongoing database schema & backend development transparent to users

Data availability

- \approx 1-minute average latency from Oracle to TimescaleDB
- Real-time data access for operational monitoring
- Historical data spanning 18 years immediately accessible

Performance Metrics - throughput & query speed

Sustained Throughput

- Meets performance requirements
- Infrastructure can be upgraded with different strategies (e.g. read replicas) to meet future user demands

Migration Capability

- 20 workers: 600,000 rows/s (scalable) with full dynamic checks on data
- Full 68 TB historical migration from Oracle production database: ~4 weeks

Visualization workflow

- Multi-detector system queries in seconds
- 10x - 100x performance improvement depending on the workload, complexity and time range
- Enables interactive analysis of years of data

Performance Metrics - compression & storage

Compression achievements

- Consistent ~11× compression ratio
- Production data: 68 TB → 6.6 TB
- Native TimescaleDB compression (no external tools)

Data Ingestion Rate

- 9 TB/year uncompressed data - expected to rise for Phase II
- ~750 GB/month average ingestion
- ~3,000 rows/s during active data collection

Storage Cost Impact

- **Significant savings on CERN enterprise storage**
- Reduced backup storage requirements
- Critical for long-term data retention

Python & C++ APIs - design philosophy

Python API

- Minimal function set (reduced learning curve)
- Standardized arguments across all functions
- Jupyter notebook environment
- Interactive data exploration

C++ API

- Header-only library (minimal dependencies) - easy integration to existing project
- Identical interface to Python API

Design Philosophy

- Hide SQL complexity completely
- Backend server handles all database logic
- Consistent experience across languages

Python & C++ APIs

```
import dcs_data
```

```
# extract LHC luminosity  
data = get_archived_data_by_name(  
    subdet_schema.DCS,  
    "ATLAS_Lumi",  
    start_time, end_time  
)  
# ≈760k values in ~3s from API
```

```
#include <dcs_data>
```

```
int main() {  
    auto data = getArchivedDataByComment(subdet_schema::DCS,  
                                         "ATLAS_Lumi",  
                                         start, end);  
}  
  
// Lightweight & headers only implementation, easy to integrate to a project
```

NextGen Archiver (NGA) Alignment

NextGen Archiver

- Next-generation archiving infrastructure for CERN control systems
- Modern replacement for legacy WinCC OA archiving - enabling direct TimescaleDB through Wincc OA
- Under active development by CERN BE-ICS group

DCS Data Tools alignment

- Fully aligned with schema defined by NGA
- Closely collaborating with BE-ICS group

Future roadmap & strategic value

- WinCC OA direct archiving to TimescaleDB capability
- Migration path preserving operational continuity
- Prove TimescaleDB production-readiness for ATLAS DCS & Wincc OA archiving

Key Achievements summary

- Production system already serving CERN community
- Significant cost savings on enterprise storage
- Sustainable long-term growth trajectory
- Real-time access with ≈ 1 -minute latency
- Barrier eliminated: no SQL knowledge required
- Replacing legacy workflows & modernizing data visualization
- Strong adoption by several groups

Acknowledgments

CERN DBOD Team

- *Infrastructure support and collaboration*
- *Problem-solving partnership throughout deployment*

CERN BE-ICS group

- *NextGen Archiver (NGA) collaboration*
- *Shared roadmap and architecture alignment*

ATLAS Detector Groups

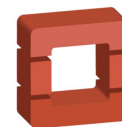
- *MDT group: Early adoption and valuable feedback*
- *LAr group: Use cases and requirements input*
- *All detector experts providing requirements through survey*

CERN IT Services

- *OpenStack development infrastructure*
- *OpenShift platform for Grafana hosting*

ATLAS DBAs

- *Development support, planning & Oracle expertise*



openstack™



DBOD
DATABASE ON DEMAND

Future Directions

Near-Term Roadmap

- Direct WinCC OA archiving to TimescaleDB (NGA integration)
- Expanded detector group adoption

Long-Term Vision

- Advanced analytics enabled by improved performance
- Easier access to DCS data by the community - facilitated by Grafana & the APIs
- Better real time error detection and monitoring
- Scaling infrastructure with user demand growth

Sustainability

- Open-source foundation ensures long-term viability
- Community-driven development reduces vendor dependency
- Model for other CERN experiments and facilities

Thank you for your
attention!

paris.moschovakos@cern.ch
dimitrios.matakias@cern.ch