A DATA SCIENCE AND MACHINE LEARNING PLATFORM SUPPORTING LARGE PARTICLE ACCELERATOR CONTROL AND DIAGNOSTICS APPLICATIONS*

C. K. Allen[†], C. McChesney, M. Davidsaver, B. Dalesio, Osprey DCS, Ocean City, Maryland, USA

Abstract

Osprey DCS is developing the *Machine Learning Data Platform* (MLDP) supporting machine learning and data science applications specific to large particle accelerator facilities and other large experimental physics facilities. It represents a "data-science ready" host platform providing integrated support for advanced data science applications used for diagnosis, modelling, control, and optimization of these facilities. There are 3 primary functions of the platform: 1) high-speed data acquisition, 2) archiving and management of time-correlated, heterogeneous data, and 3) comprehensive access and interaction with archived data. The objective is to provide full-stack support for machine learning and data science, from low-level hardware acquisition to broad data accessibility within a portable, standardized platform offering a data-centric interface for accelerator physicists and data scientists. We present an overview of the MLDP including use cases, architecture, and deployment, along with the current development status. The MLDP is deployable at any facility, however, the low-level acquisition component requires EPICS.

INTRODUCTION

As part of our effort to provide full-stack solutions for machine-learning and data science algorithms, Osprey DCS is currently developing the Machine Learning Data Platform (MLDP). It is a dedicated platform supporting rapid development and deployment of machine-learning and data science algorithms for accelerator systems and large experimental system in general. A conceptual representation of the MLDP and its operation within an accelerator facility is shown in Fig. 1.

Explicitly indicated in Fig. 1 are data acquisition, real-time monitoring and processing capabilities, a heterogeneous data archive with full provenance, and advanced search and query capabilities via a standardized Applications Programming Interface (API). Seen complementing the MLDP are common off-the-shelf tools for building data science, machine learning, and artificial intelligence applications, such as Keras, TensorFlow, PyTorch, and scikit-learn. The MLDP is leveraging off next generation technologies and techniques required for fast, large-scale heterogeneous data acquisition and collection at the front end, through transport, aggregation, and archiving, to the data-science requirements for the client facing back end. It represents the full-stack integration of machine learning capabilities for an accelerator facility into a single platform.

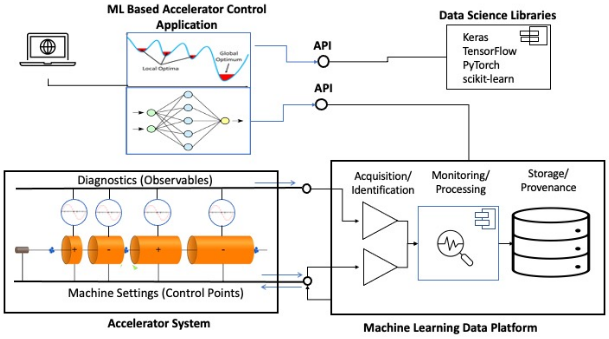
ML Based Accelerator Control Application
Local Optima
Global Optimum
API
API
Data Science Libraries
Keras
TensorFlow
PyTorch
scikit-learn
Diagnostics (Observables)
+
-
+
-
Machine Settings (Control Points)
Accelerator System
Acquisition/ Identification
Monitoring/ Processing
Storage/ Provenance
Machine Learning Data Platform

Figure 1: conceptual diagram of the MLDP.

THE MLDP

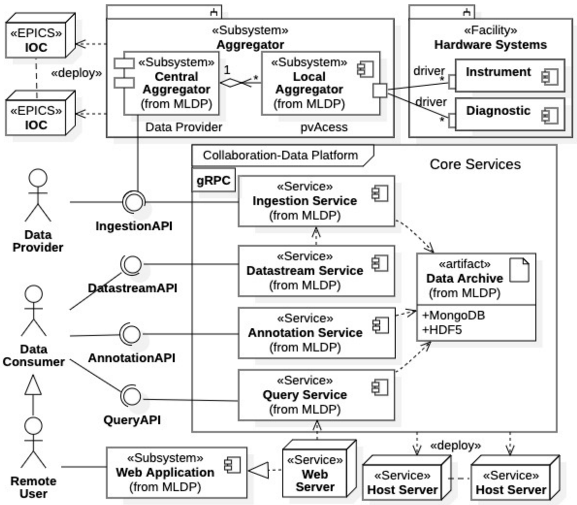
«EPICS» IOC
«deploy»
«EPICS» IOC
«Subsystem» Aggregator
«Subsystem» Central Aggregator (from MLDP)
1
*
«Subsystem» Local Aggregator (from MLDP)
Data Provider
pvAcess
«Facility» Hardware Systems
driver
*
Instrument
driver
*
Diagnostic
Collaboration-Data Platform
Core Services
gRPC
Data Provider
IngestionAPI
«Service» Ingestion Service (from MLDP)
DatastreamAPI
«Service» Datastream Service (from MLDP)
«artifact» Data Archive (from MLDP)
+MongoDB
+HDF5
Data Consumer
AnnotationAPI
«Service» Annotation Service (from MLDP)
QueryAPI
«Service» Query Service (from MLDP)
«deploy»
Remote User
«Subsystem» Web Application (from MLDP)
«Service» Web Server
«Service» Host Server
«Service» Host Server

Figure 2: MLDP subsystems, deployment, and clients.

The MLDP is comprised of 3 primary subsystems: 1) the *Aggregator*, 2) the *Data Platform* or "*Core Services*", and 3) the *Web Application*, all depicted in Figure . The Aggregator is the frontend interacting with facility hardware. The Data Platform is a fully independent system managing all aspects of archive interaction, including that seen by data-science clients and applications. It is a collaboration of independent services each supporting an archive function, thus the alias *Core Services*. A novel feature of the MLDP, archive annotation, is supported by a specialized core service (discussed below). The Web Application provides universal, remote access and interaction with the MLDP

*Work performed under the auspices of the U.S. Department of Energy with funding by the Office of High Energy Physics SBIR Grant DE-SC0022583.
†Corresponding author email address: allenck@ospreydcs.com

data archive using a standard web browser. This tool is also an independent system.

Figure 2 includes the intended clients of the MLDP, Data Provider, Data Consumer, and Remote User. Data Providers are any sources of heterogeneous, correlated, time-series data conforming to the Data Platform ingestion API; the Aggregator is one such client. Data Consumers are any parties interested in the archived data, such as data scientists, facility users, engineers, and control-room applications. A Remote User is a Data Consumer interacting with the MLDP data archive remotely (e.g., outside the control room or completely off site).

Operation and Use Cases

Figure shows the basic architecture of the MLDP and its deployment within an accelerator facility. Note that the Data Platform and Web Application are deployed on independent servers; multiple servers can be used for additional performance. We cover further details and operations of each subsystem through use cases.

Aggregator

The Aggregator has components distributed throughout the EPICS control system (seen in Fig. 2). The function of the Aggregator is shown in Fig. 3. It performs high-speed collection, correlation, and aggregation of heterogeneous, time-series data. For performance the distributed components (i.e., "Local Aggregators" of Fig 2.) are proximal to hardware systems within the EPICS environment. Local Aggregators transmit processed data to the Central Aggregator for coalescing, where it is also staged for transport to the Data Platform as tables. There, the tables are transmitted over the network to the Data Platform Ingestion Service using gRPC.

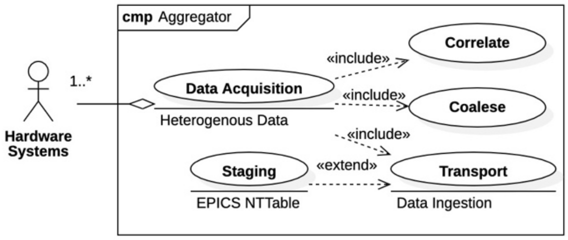

Figure 3: Aggregator use cases.

Data Platform – Core Services

The Data Platform maintains the central data archive of the MLDP containing time-series data, associated metadata, and any user annotations. It manages the archive with a set of collaborating Core Services; shown in Fig. 2 are the Ingestion Service, the Query Service, the Annotation Service, and the advanced Datastream Service. Each service has a well-defined API for external communications. All services can concurrently support multiple clients. The Data Platform uses a gRPC framework for communications, independent of any EPICS protocol. There are also programming language API libraries for external communications.

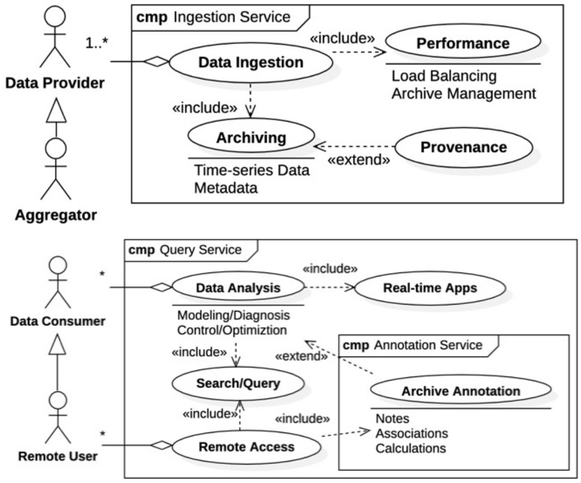

Figure 4: Ingestion Service use cases (top), Query Service and Annotation Service use cases (bottom).

The top of Fig. 4 shows the functions of the Ingestion Service. It is responsible for all time-series data ingestion and archiving, as well as any metadata associated with ingestion (e.g., properties of the data provider, process variables, or data itself). It maintains performance through load balancing ingestion thread tasks and manages all metadata necessary to support full data provenance.

The Query Service use cases are shown in at the bottom of Fig. 4. It is the primary interaction point for all Data Consumers and supports all archive search and query for both data and metadata. Also included in Fig. 4 is the Annotation Service available to Data Consumers. This service allows clients to annotate the data archive post-ingestion with additional notes, associates, and calculations obtained from archive data itself. Thus, the Annotation Service allows clients to perform *value-added* interactions with the data archive which is then available to all other clients. For further details on the Data Platform see [1].

Web Application

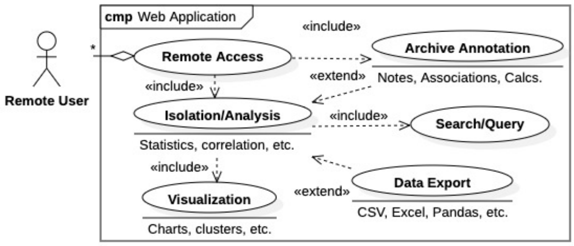

Figure 5: Web Application use cases.

The Web Application is available via a remote login URL on a separate web server (Fig. 2). The web server loads the application into a client web browser then continues to marshal all communications between the browser and core services. An inclusive list of Web Application features is shown in Fig. 5. The intent is to provide remote users a substantial subset of MLDP features, as well as other tools suitable for browser environment, such as visualization and basic data analysis.

PROJECT BACKGROUND

Formal development on the MLDP began in 2022 with a Small Business Innovative Research (SBIR) grant from the Department of Energy (DOE) Office of High-Energy Physics (HEP)*. In Phase I a prototype was developed demonstrating proof of principle. Due to time constraints many off-the-shelf components were used in prototype construction and prototype performance was far below project goals. However, the prototype worked and was comprehensively tested and benchmarked; those evaluations formed the basis of most of our Phase II designs.

The Phase II effort began in 2023 with extensive design studies including performance evaluation and benchmarking of existing technologies and methods. It was determined that a complete redesign and rebuild of the Core Services was necessary to achieve performance goals. Additionally, the Web Application was to be rebuilt with focus on the user interface and interaction.

The new Core Services design eliminated many 3rd party components, streamlined all processing with a concurrent Task/Worker/Manger pattern, and modularized functionality. It relies upon 3 technologies: 1) Java offering reasonable trade-off between development effort and performance, while also simplifying installation and deployment [2]. 2) gRPC provides a fast, language-neutral communication framework allowing future development in other languages [3]. 3) MongoDB database system is used for archiving time-series data, metadata, and annotations [4]; its NoSQL nature is ideal for the diverse types and during evaluations it was found to have exceptional performance.

Status

As of this writing the Aggregator system is mature and has been benchmarked using 3,200 signals at 1 kHz data rates. The Ingestion Service and Query Service are operational for scalar data. Design and development of the Annotation Service began early in 2024 and it is now capable of basic comment annotations. The Web Application is currently being rebuilt with emphasis on "look and feel" rather than the previous feature-oriented design.

A significant project milestone was obtained early in Phase II; performance goals were achieved for ingestion data rates with the new design. Data rates up to 200 Mbps were obtained while ingesting scalar tables such as those produced by the Aggregator. This value is over 6x that of our stated goal and 200x that of the prototype. Early measurements of Query Service data rates are on the order of 100 Mbps, though testing and optimization is still ongoing.

A formal deployment and installation system for the Data Platform component has been developed and is currently online‡ [5]. The system is installed via a downloadable zipped archive. The Core Services are deployed as executable Java archives (i.e., "fat jars") and there are shell scripts for managing the services and utilities that ship with the installation. The installation repository also contains extensive documentation, including instructions, Data Platform documentation (online support, reports, presentations, etc.), release notes, and developer notes. Java 16 and MongoDB 6.0 (Community Edition) installations are minimal requirements for the hosting platform, although we recommend Java 20 and MongoDB 7.0.

A Java language client API library is available for the Core Services. The library currently contains APIs for the Ingestion Service and the Query Service. As Python is popular in data science and machine learning applications, a Python language client library is also planned. The programming language API libraries are targeted towards MLDP data science applications as direct gRPC communications is technical.

A significant feature of the MLDP planned for Year 2 of Phase II is the development of datastream processing capabilities (the Datastream Service of Fig. 2). The intent is to provide fast processing capabilities within the ingestion data stream. The feature is most suitable for online control applications responding to hardware events in real time. We are exploring a "plugin" framework for this feature; for example, if a particular algorithm is deemed "exceptional" it can be incorporated directly within the data stream as a plugin providing fast processing and access to results.

CONCLUSIONS

The Machine Learning Data Platform is in mid-development, but many useful features are now available. In particular, the Data Platform component managing the archive and supporting the MLDP data science capabilities can be deployed locally with a formal installation system available online [5]. Beta releases are to be publicly available.

REFERENCES

[1] C. McChesney, C. Allen, L. Dalesio, and M. Davidsaver, "The Data Platform: an independent system for management of heterogeneous, time-series data to enable data science applications", presented at the IPAC'24, Nashville, TN, USA, May 2024, paper TUPS70, this conference.

[2] Java, Oracle, `https://www.oracle.com/java/` (2024).

[3] gRPC, gRPC Authors, `https://grpc.io/` [Online documentation], `https://github.com/grpc` [code repository] (2024.

[4] MongoDB Community Edition, MongoDB Inc., `https://www.mongodb.com/products/self-managed/community-edition` (2024).

[5] Data Platform, Osprey DCS, `https://github.com/osprey-dcs/data-platform`

‡ Access to the Data Platform installation repository is currently private but scheduled for public beta release once the API is stable. In the interim, interested parties may contact Osprey DCS for access.