I. PROJECT IDENTIFICATION

a. Award ID Number:	DE-SC0022583
b. Agency:	US Department of Energy, Office of High Energy Physics
c. Award Program:	SBIR Initial Phase II, DE-FOA-0002991
Topic:	29 (d): Data Infrastructure for the Next Generation of Adaptive Real-Time Controls for Large-Scale Facilities
d. Project Title:	A Data Science and Machine Learning Platform Supporting Large Particle Accelerator Control and Diagnostics Applications
e. Principal Investigator:	Leo R. Dalesio bdalesio@ospreydcs.com (443) 834-3775
i. Company Information:	Osprey Distributed Control Systems 304 Blue Heron Court Ocean City, Maryland 21842-2452
j. Grant Period:	Aug 2023 to July 2025
k. Reporting Period:	Aug 21, 2023, to Nov 19, 2023
l. Report Term:	Quarter 1
Proprietary Data:	There is no proprietary data. All work performed in this project is open source. It will be developed in a GIT repository and provided to the EPICS collaboration for use throughout that community.

Table of Contents

Research Performance Progress Report: Quarter 1

DE-SC0022583 SBIR Initial Phase II

II. ACCOMPLISHMENTS

A major milestone was accomplished in the first quarter, the performance goals were achieved for a redesigned and newly implemented Datastore Ingestion Service. Additionally, a substantial body of knowledge was obtained for future development efforts through a coordinated and exhaustive set of evaluations.

To appreciate the development approach, it is instructive to reiterate the project tasks and milestones.

1. Project Goals

As stated in the original narrative, the following tables provide itemized task lists and milestones for the current SBIR Phase II project. The tables include estimated schedules, levels of effort, and staffing.

Year 1

Task	Items	Start	End	Effort	Yearly	Milestones
Datastore Upgrade	Upgrade/Redesign	Q1	Q3	6 MM		100 Mbyte/sec Full Annotations
	Performance	Q1	Q3	3 MM		
	Annotations	Q2	Q3	1 MM		
	Senior Staff			10 MM	0.83 FTE	
APIs	Administration API	Q2	Q3	2 MM		
	Python Query API	Q3	Q3	1 MM		
	Junior/Senior Staff			3 MM	0.25 FTE	
Adv Data Science	Applications	Q2	Cont.	1 MM		
	Archive Support	Q4	Q4	2 MM		
	Senior Staff			3 MM	0.25 FTE	
Deployment	MLDP Testing	Q4	Q4	1 MM		Data Throughput Aggregator Install.
	Aggregator Install.	Q4	Cont.	1 MM		
	Senior Staff			2 MM	0.16 FTE	
Web Application	Upgrades	Q1	Cont.	2 MM		
	Visualization	Q1	Cont.	1 MM		
	Data Science	Q1	Cont.	1 MM		
	Junior/Senior Staff			4 MM	0.33 FTE	

Table 1: Phase II Year 1 Performance Schedule

Year 2

Task	Items	Start	End	Effort	Yearly	Milestones
Adv Data Science	Applications	Q1	Q4.	2 MM		MLDP Appl. Plugin Operation
	Datastream Process.	Q1	Q3	4 MM		
	Plugin Framework	Q1	Q3	4 MM		
	Integrat./Performance	Q2	Q4	4 MM		
	Senior Staff			14 MM	1.16 FTE	
APIs	ADS Support	Q2	Q3	1 MM		Facility Install.
	Documentation	Q3	Q3	1 MM		
	Senior Staff			2 MM	0.16 FTE	
Deployment	Deployment Systems	Q1	Q4	1 MM		Cloud Deploy. Facility Install.
	Cloud Deployment	Q1	Q4	1 MM		
	Facility Installation	Q1	Q4	1 MM		
	Senior Staff			3 MM	0.25 FTE	
Web Application	Commercialization	Q1	Q4	2 MM		Remote Demon.
	Data Science	Q1	Q4	1 MM		
	Junior/Senior Staff			3 MM	0.25 FTE	

Table 2: Phase II Year 2 Performance Schedule

2. Quarter Accomplishments

The primary focus of first quarter activities was to address the performance issues with the Datastore component of the Machine Learning Data Platform. Stated performance goals are data rates of at least 32 Mbytes/second (Mbps) for data ingestion and transport whereas Phase I evaluations found these rates less than 1 Mbps[1] for the Datastore prototype. Meeting performance objectives requires at least a 100-fold increase in data rates.

i. Datastore Redesign

It was determined that a complete redesign of the ingestion and archiving system was necessary to achieve performance goals. The design studies included extensive evaluations and benchmarking of existing technologies and methods. The focus of these efforts was based upon the following considerations for the Datastore design (in order of importance):

1. Performance
2. Modularity/dependency
3. Development effort
4. Ease of deployment

A summary of the technology benchmarking results is shown in Table 3. The benchmarking contains 3 categories distributed within the table rows:

1) gRPC Network Transmission – Overall gRPC network rates without archiving using Java gRPC.
2) Data Archiving – Time series data storage rates for specific archiving systems.
3) Metadata Update –Archive metadata update rates.

[1] The original proposal stated rates of 32 Mbps for native 8-byte wide double values and 100 Mbps for Java 24-byte wide Double object. For the sake of clarity Osprey DCS has formalized all performance criteria to consider only native formats, which is consistent with all low-level transport and storage mechanisms.

Benchmark Description	Data Rates	
	Double Values (vals/sec)	Bytes (bytes/sec)
gRPC network transmission (Java)	22M – 33M	176M – 264M
Archiving, structured - HDF5 large	68M – 77M	544M – 616M
Archiving, structured - JSON files	38M – 47M	304M – 376M
Archiving, buckets - MongoDB	7M – 11M	56M – 88M
Archiving, buckets - MariaDB	4.5M – 5.5M	36M – 44M
Archiving, structured - HDF5 small	1.3M – 2.4M	10.4M – 19.2M
Archiving, points - InfluxDB	750K – 940K	6M – 7.52M
Archiving, points - MongoDB	360K – 410K	2.88M – 3.28M
Archiving, points - MariaDB	140K – 162K	1.12M – 1.3M
Metadata updates - MongoDB	11K to 36K updates/sec	-

Table 3: Datastore component benchmarking summary

Table 3 is informative in that is isolates data archiving performance from data transmission performance. In particular, the archiving performance is reported for multiple data storage systems and techniques. Four disparate data archiving systems and techniques are considered:

1) System File Archiving – Direct to disk archiving using JSON and HDF5 file formats.
2) MongoDB – A formal NoSQL, document-based database system.
3) MariaDB – A formal relational database.
4) InfluxDB – A formal database system tailored for time-series, measurement data.

From the table we see that file archiving with HDF5 format performs best overall when storing structured data directly to disk.

In addition to the technology benchmarking a parallel effort was performed investigating the feasibility of a Datastore implementation in the C++ language. Such an approach yields executables in the host platform's native binary format. Presumably this condition yields the fastest performance available, although at the cost of added development effort and deployment overhead. The original Datastore prototype was implemented in the Java language.

For these evaluations a basic Ingestion Service was built in C++ utilizing gRPC as the communications framework (as before). Figure 1 depicts a component-level schematic of the C++ testbed. The testbed simulated the creation, transmission, and ingestion of large heterogeneous data tables over a local network. The objective was to study a basic system focused solely on performance. A simple archiver scheme was employed utilizing direct-to-disk binary storage, no formal database systems were employed. Several configurations and transmission strategies were investigated.

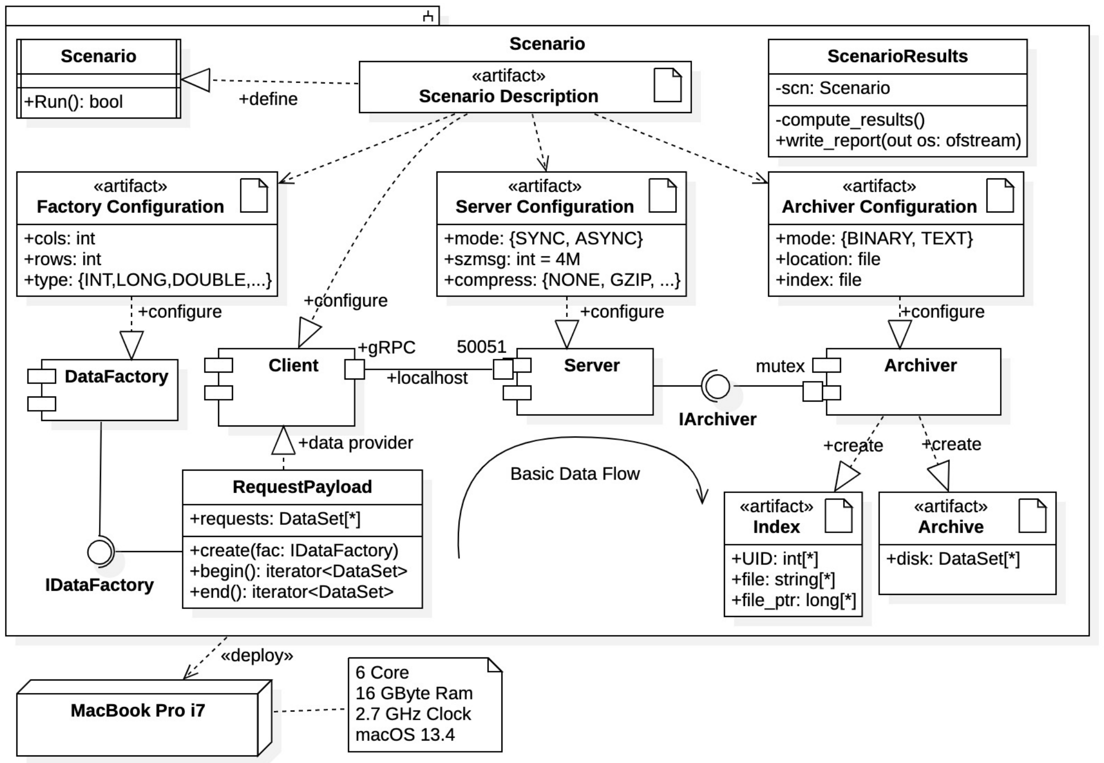


Figure 1: C++ Ingestion Service testbed

A summary of the C++ results is shown in Table 4. The summary table contains results for varying sizes of data tables being ingested and for various transmission methods (e.g., synchronous, asynchronous, unary, and streaming). Results are presented for both archiving and when archiving is turned off. The table dimensions stress the carrying capacity of the data transport while the transmission methods are properties of the gRPC system. When archiving is disabled the gRPC network transmission rates are isolated.

Table Dimension	**Unary Calls**				**Streaming Calls**			
	Synchronous Rate		**Asynchronous Rate**		**Synchronous Rate**		**Asynchronous Rate**	
	Archived (Mbps)	None (Mbps)	Archived (Mbps)	None (Mbps)	Archived (Mbps)	None (Mbps)	Archived (Mbps)	None (Mbps)
100x100	61.072	74.605	97.076	108.22	148.49	321.56	168.11	345.46
500x500	64.150	87.867	145.70	170.95	146.84	317.38	186.28	434.45
1000x500	62.884	88.455	145.93	168.50	154.183	316.10	207.53	454.57
1000x1000	66.765	91.027	145.64	168.92	143.08	320.28	200.18	450.92
2000x250	65.976	89.526	146.37	168.54	150.12	319.86	200.51	435.93
2000x500	66.616	91.863	146.73	163.93	145.48	323.86	206.67	458.57
4000x100	63.113	87.123	129.10	155.09	140.56	320.55	208.41	425.00
4000x250	59.520	89.463	145.11	160.97	146.42	319.15	196.41	455.24
4000x500	49.864	62.585	121.35	135.48	94.436	118.88	154.59	213.15
4000x1000	39.352	45.955	106.57	110.75	69.272	74.581	131.31	158.84

Table 4: peak transmission and archiving rates in C++ implementation

Upon conclusion of the design studies a new course of action was decided. The Datastore system was to be completely redesigned and rebuilt. Although operational, development of the previous prototype was to be suspended. The new design is based upon the following systems:

- Java – The Java programming language offers a reasonable tradeoff between performance, development effort, and ease of deployment. C++ provided the best performance but at a significant development increase, particularly so for gRPC development and installation. Java also provides a platform independent deployment solution.
- gRPC – It was decided to maintain the Datastore as a standalone system independent of EPICS. The gRPC system provides a fast, platform-independent, and language neutral communication framework that is offers modularity and extensibility. The latter feature being particularly important for future Advanced Data Science efforts.
- MongoDB – The MongoDB database system is now used for both time-series data storage and metadata storage. Previously it was utilized solely for metadata archiving due to its document-based storage structure. With redesign of the time-series data archive, the MongoDB system was found to have exceptional performance. Additionally, it is freeware and installs on most platforms.
- HDF5 – The HDF5 library provided the fastest overall storage performance. Its "self-describing data file" format is compatible with the archiving of heterogenous, time-series data. However, it requires development overhead required for file management and organization.

Note that many third-party systems and components are eliminated from the Datastore prototype, including the InfluxDB time-series database system and the Spring and SpringBoot application frameworks.

Although C++ is not to be used for general Datastore development it was discovered that C++ gRPC provides exceptional performance when utilizing asynchronous, streaming communications. Unfortunately, this gRPC protocol is the most difficult to implement and deploy. However, due to the flexibility of a gRPC communications framework it is possible to create specialized Datastore components built in C++. C++ and C++ gRPC are particularly attractive for the datastream processing and plugin components to be addressed in Year 2; these are real-time systems requiring exceptional performance characteristics. Thus, the implementation of asynchronous, streaming gRPC communications in C++ was carefully documented after the evaluations.

We are not currently utilizing the HDF5 library but are reserving it as an option for specialized data storage. Specifically, we are considering HDF5 as a companion storage system for legacy data. We anticipate that the MongoDB database archive may become overwhelmed when storing data over extended periods. Thus, historical data can be moved to HDF5 disk storage when MongoDB nears capacity (such size is yet unknown). If legacy data is queried by a data consumer it can then be "hot-swapped" back to main storage.

The basic architecture of the Machine Learning Data Platform (MLDP), including the new Datastore design, is shown in Figure 2. The diagram also includes MLDP subsystem and client relationships.

It is important to note that a new naming convention has been adopted; *Datastore* has been replaced by *Data Platform* (DP). Since the Datastore prototype has been completely redesigned and is being rebuilt it was prudent to identify it as such. The new convention helps to clarify the new system from the previous prototype, as well as isolation components within code repositories. Note further that, due to the redesign and current reimplementation effort, component naming and API definitions are still fluid.

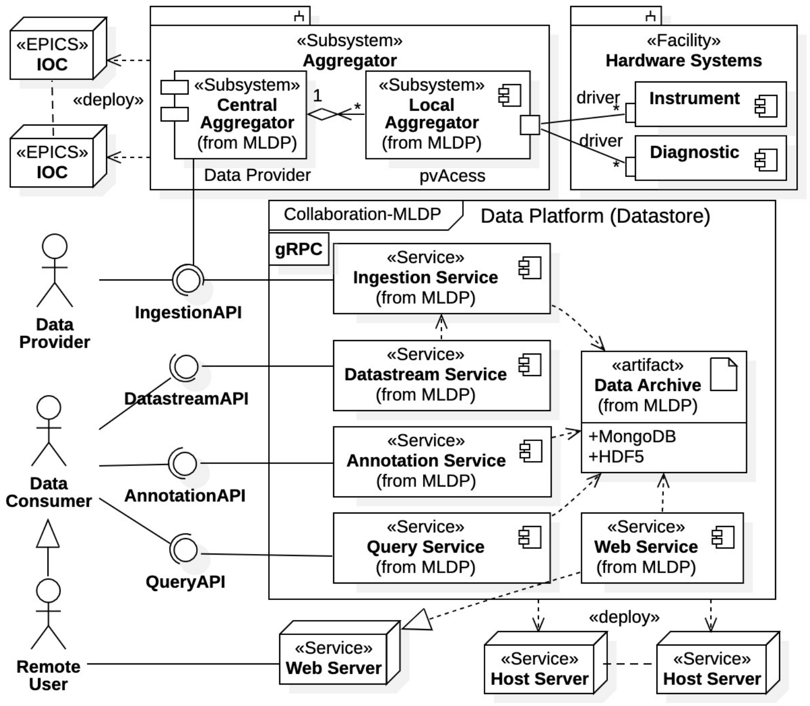


Figure 2: MLDP composite diagram with client relationships

As seen in the diagram the new *Data Platform* is composed of a set of collaborating services, all communicating through gRPC. The new design is simplified, performance based, and distributed. Additionally, the gRPC communications framework was narrowed and simplified to accommodate the distributed nature.

Due to the poor performance of the Datastore prototype Query Service component, it was decided to separate the annotations capabilities into a separate Annotations Service. Additionally, the query processing is now distributed between Query Service and client API library (no shown). Specifically, for expedited performance the Query Service within the Data Platform provides minimal functionality whereas significant data processing and data reassembly is to occur locally on the client platform.

ii. Performance – New Ingestion Service

A new Ingestion Service has been implemented in the first quarter based upon the above design considerations and the new technology adaptation. Figure 3 provides a breakdown of the internal use case requirements addressed by the new Ingestion Service. Note that the Ingestion Service can concurrently service multiple Data Provider clients.

For Ingestion Service testing a benchmarking service was also developed (included with the deployment). The benchmarking service is similar to that for the C++ testbed shown in Figure 1, simulating data creation and transmission. Data tables of various dimensions and sizes are continuously transmitted to the Ingestion Service using its gRPC communications API while various performance and operational characteristics are measured. Hosting on a MacMini platform with M2 CPU containing 10 cores and 16 Gbytes memory the new Ingestion Service achieved *an overall transmission and archiving rate of 200 Mbps*. This value is over 200 times the peak performance of the Datastore prototype, and over 6 times the stated SBIR Phase II project performance goal.

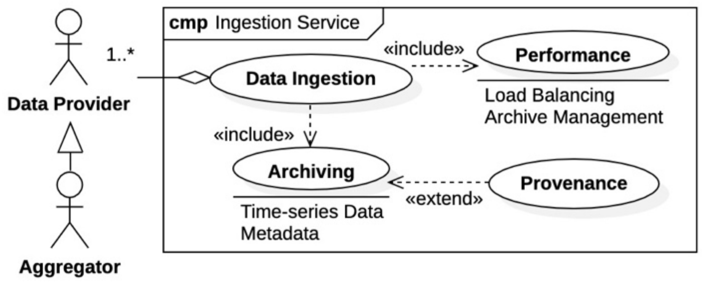


Figure 3: Ingestion Service internal use cases

The above performance number represents a significant, and early, accomplishment within the Phase II effort. Osprey DCS has thus achieved a stated project milestone for one system component. This accomplishment implies that our overall performance expectations for data accessibility are reasonable and achievable. We are currently pursuing the same goals while developing a second component, the Query Service, where performance characteristics are expected to be on par with the Ingestion Service.

The performance achievement also reflects on the necessity and subsequent success of the Datastore redesign efforts. During the Datastore prototype evaluations of Phase I it was found that data rates depended strongly upon the shape and structure of the data ingested or queried. Rates for data transported in the form of wide scalar tables (e.g., tables with thousands of data columns) performed substantial worse than all other forms (arrays, structures, imagines, etc.), with peak data rates as low as 0.25 Mbps in the old benchmark convention, which is 0.03 Mbps in the current convention. Additionally, the performance drop was proportional to the number of table columns. The current benchmarking utility specifically targets the worst-case scenario, and in that case, the performance improvement is *over 5,000 times the original Datastore prototype*. More importantly, in the new design there is no longer any correlation between table width and performance.

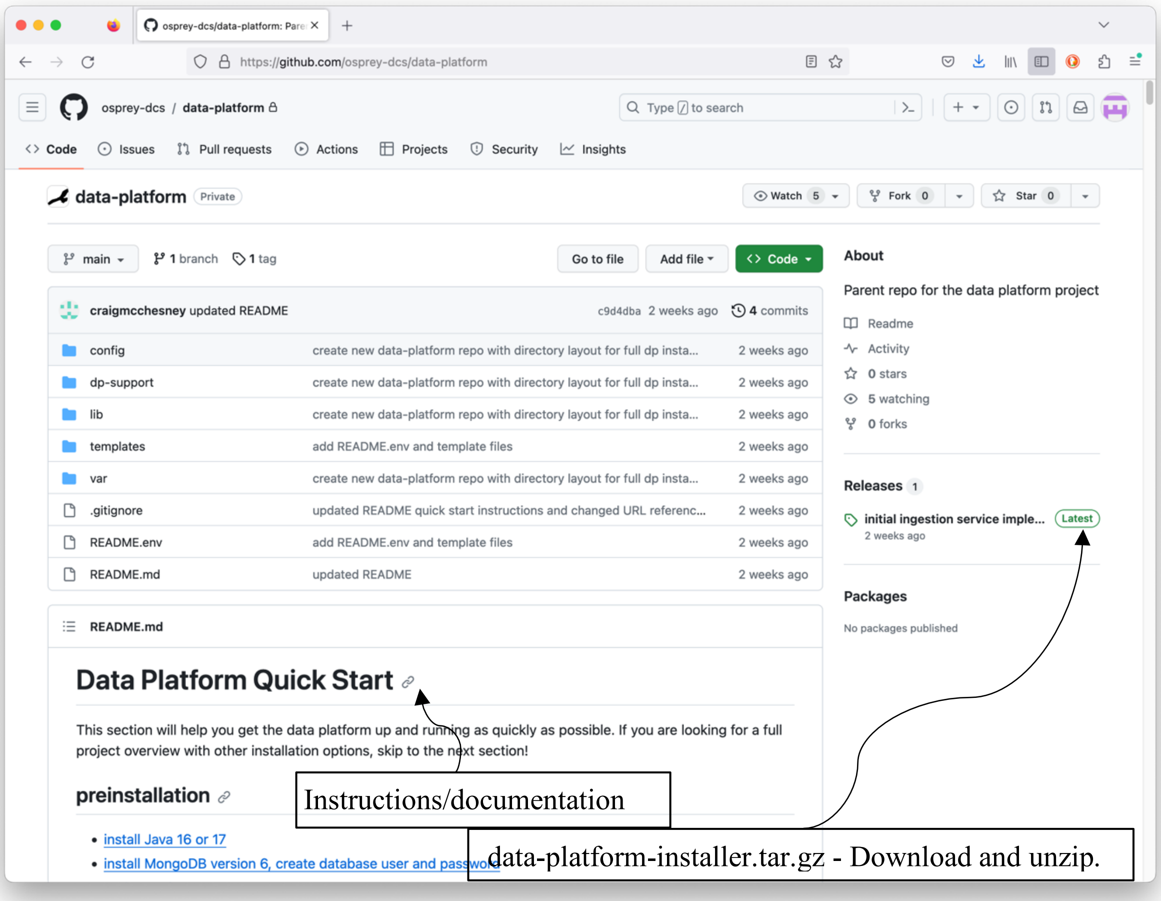
iii. Deployment Systems

A formal deployment system for the new Data Platform system has been initiated and is currently operational with basic functionality. The previous Datastore prototype had no formal deployment system. The prototype has multiple components distributed across 6 code repositories plus an additional repository containing a data simulator used for testing. Datastore prototype installation was a complex process where each code repository required cloning, building, and local installation. In addition, multiple third-party system required installation and configuration, such as InfluxDB, MongoDB, and several code libraries and frameworks.

The Data Platform system is now fully deployable as a zipped archive. This archive is available for download, hosted as a Github repository located at https://github.com/osprey-dcs/data-platform. A screen capture of the web page is provided in Figure 4. All available (pre-)releases of the Data Platform installer are available for download by selecting the desired version within the "Releases" section on the page, as indicated in the figure.

The installation and deployment process for the Data Platform now consists of the following steps:

1. Install Java (version 16 or greater) locally.
2. Install MongoDB (free Community Edition) and create administrative account.
3. Download and unzip the Data Platform installation archive.
4. Set the DP_HOME environment variable to the local installation directory.



craigmcchesney updated README		c9d4dba 2 weeks ago 4 commits
config	create new data-platform repo with directory layout for full dp insta...	2 weeks ago
dp-support	create new data-platform repo with directory layout for full dp insta...	2 weeks ago
lib	create new data-platform repo with directory layout for full dp insta...	2 weeks ago
templates	add README.env and template files	2 weeks ago
var	create new data-platform repo with directory layout for full dp insta...	2 weeks ago
.gitignore	updated README quick start instructions and changed URL referenc...	2 weeks ago
README.env	add README.env and template files	2 weeks ago
README.md	updated README	2 weeks ago

About

Parent repo for the data platform project

Readme
Activity
0 stars
5 watching
0 forks

Releases 1

initial ingestion service imple... Latest
2 weeks ago

Packages

No packages published

README.md

Data Platform Quick Start

This section will help you get the data platform up and running as quickly as possible. If you are looking for a full project overview with other installation options, skip to the next section!

preinstallation

- install Java 16 or 17
- install MongoDB version 6, create database user and password

Instructions/documentation

data-platform-installer.tar.gz - Download and unzip.

Figure 4: screen capture of the Data Platform installation repository

The process is well documented. There are detailed instructions for each step included within the installation home page, as well as extended documentation for the Data Platform and release notes for the current installation (as shown in Figure 4).

The component services of the Data Platform are deployed as independent, fully executable Java archive files (i.e., Java “fat jars”). For convenience the installation contains a package of executable shell scripts used to start and stop the various services, include the MongoDB database system. The current release contains an operational Ingestion Service with basic functionality. It also includes a data simulator that is used to benchmark the Ingestion Service performance on the current platform. An operational Query Service is intended for the next release.

Installation repository public access is not yet provided since the Data Platform gRPC communication interfaces are still under development (i.e., the Query Service interface). However, parties interested in inspection and operation of the Ingestion Service component may obtain access by contacting Osprey DCS. Note that any external development based on early releases is discouraged since the service interfaces are subject to change.

iv. Advanced Data Science

Advanced Data Science support was addressed in the new Data Platform archive design. Preliminary design efforts for the datastream systems were also addressed in the C++ evaluations; specifically, a fast gRPC communications framework was prototyped, characterized, and documented [1].

v. Web Application

Osprey DCS is currently evaluating candidates for a junior developer position. The new hire is to be assigned to Web Service and Web Application development.

vi. Documentation

The code repository hosting the Data Platform deployment system is now also hosting a central MLDP documentation library. Activities for creating a formalized MLDP online documentation system are currently ongoing.

3. Training Opportunities and Professional Development

Notable professional development was required for project participants C. McChesney and C.K. Allen in the first quarter. Meeting project performance and operational goals necessitated design studies for the new Data Platform to identify the most promising courses of action. The activity involved significant research and investigation into state-of-the-art technologies and methods in the areas of data transport and data storage.

4. Dissemination of Project Results

Several MLDP developers plan to attend the International Particle Accelerator Conference in Nashville, Tennessee (IPAC24), having submitted abstracts concerning the current project results and status. Additionally, Osprey DCS has provided notification and announcements of Data Platform development and the larger MLDP efforts to its current customer base and within the control system community.

5. Activity Schedule for Next Quarter

The focus areas of the second quarter are the Query Service and Annotations Service components of the Data Platform. The development of client Application Programming Interface (API) libraries also anticipated sometime in this quarter, once the gRPC communications interfaces for the Data Platform are sufficiently established. Thus, there are 3 focus activities scheduled for the second quarter: 1) Query Service development, 2) Annotations Service development, and 3) client API library development.

1) Query Service – In the new design the Query Service has been decomposed into two services: the formal Query Service, and the Annotations Service. (That is, the archive annotations capabilities of the Data Platform have been isolated and formalized.) In the new design, the Query Service has been streamlined to provide only basic search and query operations; archive annotations are delegated to the Annotation Service, and all high-level data processing has been delegated to external client API libraries. The new approach focuses on performance within the Data Platform, which must service multiple concurrent clients. From Datastore prototype evaluations it was found that advanced data processing within the Query Service creates a significant performance cost for all clients. Within the new design data processing is distributed between multiple services and clients. A preliminary implementation of the Query Service based upon the new design has been initiated but is not yet complete.
2) Annotations Service – The post-ingestion annotation of the data archive is crucial feature of the Data Platform. It allows clients to interact and contribute to the data archive in the form of notes, data associations, and calculations based upon the current archive condition. Thus, data scientists and applications can create "value-added" data to the archive which is then available to all other users. A significant and ongoing development effort is anticipated for this service, requiring feedback from users. Creating a separate service responsible for this use case is a significant design modification that provides enhanced modularity within the Data Platform. Development of this service requires an archive schema supporting annotations, a gRPC interface for user interaction, and the service itself. We plan to develop a basic Annotation Service component during the second

quarter providing basic functionality with limited annotations. As mentioned, full functionality will require ongoing development.

3) Client API Libraries – Client libraries provide a programming language interface to the Data Platform. Direct communications with the Data Platform require familiarity with the gRPC framework, which is efficient but can be involved. Programming language support for Data Platform communications greatly simplifies client interaction and is particularly desirable for backend users of the Query Service and Annotations Service, specifically for data scientists. Client API libraries in both the Java and Python languages are planned for these services, the Python API being based upon the Pandas library familiar to most data scientists. We also intend to provide a Java language API library for the Ingestion Service facilitating simplified communications for Data Providers (enhancing portability of the Data Platform). Development of API libraries can be leveraged off the client libraries previously developed for the Datastore prototype (although functionality and gRPC interfaces have changed). Since the new Ingestion Service is currently operational the Java API library for this service is the first to be address in the second quarter and completion is anticipated. Initial development of the Query Service Java and Python libraries is also planned once the gRPC interface is sufficiently established. Development of a client API library for the Annotations Service will not proceed until the service is mature, likely in the third quarter.

II. PRODUCTS

A main component of the Data Platform has been developed, the Ingestion Service. The Ingestion Service is responsible for populating and maintaining the heterogeneous, time-series data archive, a primary component of the MLDP. The Ingestion Service demonstrated basic functionality and performed beyond stated project requirements. Successful demonstration of the Ingestion Service also indicates success of the new archive design and implementation.

1. Publications

No formal publications have yet been produced. Publicly available publications concerning current development efforts, products, and status of the MLDP are planned as proceedings of the IPAC25 conference. Journal publication(s) are anticipated as a final product of the project once the technologies required for fast heterogeneous data management are mature and/or the MLDP has been used successfully in real-world application.

Documentation on the status and operation of the Data Platform is available on the installation repository web page. Additionally, an internal document detailing C++ gRPC implementations has been produced [1]. The latter document contains necessary information for future development of the Advanced Data Science systems to commence in Year 2.

2. Intellectual Property

Osprey DCS claims no intellectual property rights over the current SBIR project results. All products are to be open source and freely available to the community.

3. Technologies and Techniques

Osprey DCS is developing technologies and techniques for the fast storage and retrieval of heterogenous, time-series data and metadata necessary for data science applications within a large particle accelerator facility. The capability to annotate archived data with user notes, relationships, and calculations is also of particular importance as it provides a "value-added" archive nature obtained from user interaction. These technologies and techniques are to be shared with the accelerator and experimental physics community through open-source availability of public code repositories.

4. Other Products

A formal deployment system for the Data Platform component of the MLDP is now available. It has basic functionality and can install current pre-releases of the Data Platform for testing and external inspection. The Ingestion Service and a companion benchmarking utility is included in the initial installation release.

III. PARTICIPANTS AND COLLABORATING ORGANIZATIONS

Due to the preliminary state of the Phase II SBIR project, all design and development activities within the first quarter have been internal to Osprey DCS, although MLDP developers have sought guidance and advice from the larger developer community. Collaboration is premature until communications interfaces and APIs are well established. This situation will likely remain throughout the first year.

External organizations have expressed interest in utilizing MLDP components as they become available. As a particular instance, Sierra Peaks company contracted with Osprey DCS for control systems design and development support concerning their efforts on a current DOE project (SCORPIUS); the contract included specific investigations of the Datastore as a system component.

1. Participants

1) Name: Bob Dalesio
2) Project Role: Principal Investigator
3) Effort (months): 1
4) Contributions: Project management, high-level MLDP architecture.
5) Foreign Collaboration: No.
6) Foreign Travel: No

1) Name: Craig McChesney
2) Project Role: Senior Developer
3) Effort (months): 3
4) Contributions: Data Platform redesign, Ingestion Service development, deployment systems.
5) Foreign Collaboration: No
6) Foreign Travel: No

1) Name: Christopher K. Allen
2) Project Role: Senior Developer
3) Effort (months): 3
4) Contributions: Data Platform design studies, datastream processing investigations.
5) Foreign Collaboration: No
6) Foreign Travel: No

1) Name: Michael Davidsaver
2) Project Role: Senior Developer
3) Effort (months): 1
4) Contributions: Data Platform design.
5) Foreign Collaboration: No

6) Foreign Travel: No

2. Partners

Sierra Peaks was briefly involved with investigations of the Datastore prototype before the Phase II project was initiated. Sierra Peaks company contracted with Osprey DCS for control systems design and development support concerning their efforts on a current DOE project (SCORPIUS). Their staff discussed development and system integration.

1) Organization Name: Sierra Peaks
2) Location: Albuquerque, New Mexico
3) Contribution: Consultation
4) Financial Support: $30,000 - contracted for control system support (included use of Datastore)
5) In-kind Support: None
6) Facilities: None
7) Collaborative Research: None
8) Personnel Exchanges: None

3. Other Collaborators

No external organizations have yet participated in any formal development of MLDP systems.

IV. IMPACT

As the current SBIR Phase II project is in its initial stage, direct impact on particle accelerator systems has yet to be determined. However, there is a growing interest in the project as we disseminate our current activities within the community.

It is anticipated that the technologies developed for rapid storage and retrieval of heterogeneous data will have a significant influence in the areas of data systems and data transport. The fact that we achieved a performance milestone for the Ingestion Service is significant.

1. Impact on Principle Discipline

2. Impact on Other Disciplines

3. Impact on Development of Human Resources

4. Impact on Physical, Institutional, and Information Resources

[Describe ways, if any, in which the project made an impact, or is likely to make an impact, on physical, institutional, and information resources that form infrastructure, including: physical resources such as facilities, laboratories, or instruments; institutional resources (such as establishment or sustenance of societies or organizations); or information resources, electronic means for accessing such resources or for scientific communication, or the like.]

5. Impact on Technology Transfer

6. Impact on Society Beyond Science and Technology

7. Foreign Spending

No project expenditures have been made in or to foreign countries. None are expected in the future.

V. CHANGES AND PROBLEMS

No significant modifications to the project approach or schedule have been made. The Datastore system of the original MLDP prototype has been redesigned and is current being rebuilt. However, this was a possibility that was anticipated and addressed in the SBIR Phase II Project Narrative. The details of the final design differ from preliminary designs offered in the original narrative, as they are the result of exhaustive design studies conducted at project initiation.

1. Changes in Approach

Crucial to project success is the efficient redesign of the Data Platform, from which almost all other MLDP activities rely. Some Advanced Data Science investigations were performed early, as they could be conducted in parallel with the design studies.

2. Actual and/or Anticipated Problems

As described above, the Datastore subsystem of the Machine Learning Data Platform (MLDP) required a complete redesign and rebuild to meet project performance requirements. The likelihood of such an action was recognized in the Phase I Final Report and included in the original Phase II Project Narrative. The Phase II schedule included estimates of time and effort if this action was necessary.

3. Changes Requiring Significant Impact on Expenditure

No changes in project expenditures have been incurred.

4. Significant Changes in Use of Human Subject, Animals, and/or Biohazards

The project uses no animals or hazardous materials. Other than feedback and critiques from MLDP users, there are no human subjects or experimentation.

5. Changes in Primary Performance Site Location

There are currently no changes in site locations.

6. Carryover Amount

There are no project carryover amounts from this quarter.

VI. DEMOGRAPHIC INFORMATION

[Provide email addresses for each participant listed in the participant section of this report. Once you submit this report, PAMS will send the participants not registered in PAMS an email inviting them to register and complete their PAMS person profiles so that any demographic information provided can be collected. Entering demographic information is optional for participants. Demographics are collected for reporting purposes.]

Bob Dalesio: bdalesio@ospreydcs.com

Craig McChesney: cmcchesney@ospreydcs.com

Christopher K. Allen: allenck@ospreydcs.com

Michael Davidsaver: mdavidsaver@ospreydcs.com

VII. SPECIAL REPORTING REQUIREMENTS

There are currently no special reporting requirements for this project.

References

[1] C. K. Allen, "Evaluation of C++ gRPC," Osprey DCS, TM-2023-003, Ocean City, Maryland, 2023.