BRINGING LARGE-SCALE ANALYTICS TO ACCELERATORS*

N. Malitsky#, BNL, Upton, NY 11973, USA

Abstract

The report presents a new approach for storing and processing both the accelerator control data and the experimental results. It is based on the analysis and consolidation of several modern technologies, such as the EPICS control infrastructure, the SciDB array-oriented data management and analytics platform, the HDF5 file format, and others.

INTRODUCTION

The efficient data management and processing systems are essential tools in the commissioning and operation of the accelerator facilities and large scientific experiments. For example, analysis of historical data is heavily involved in the troubleshooting process, detection and study of the composite behaviour patterns, comparison and design of the different operational scenarios, and many other operational tasks. The data acquisition system (DAQ) is responsible for collecting the detector measurements which are the primary results of the dedicated experiments.

Building these systems however represents a serious challenge for control developers that have to acquire and store heterogeneous data streams coming at different rates from tens of thousands of distributed devices. The conventional technologies, such as relational database management systems, were not designed for such requirements. As a result, many of the control teams had to build new proprietary tools from scratch or try to accommodate the existing technologies with some ad hoc extensions. For example, the Experimental Physics and Industrial Control System (EPICS [1]) collaboration maintains more than four archiver systems. Constrained by the associated technologies and available resources, each of these solutions cannot address all requirements and present some trade-off among different objectives: performance, reliability, extensibility, and others.

The scale, data rate, and complexity of the new light source facilities introduce the next challenge and demands for new approaches. Particularly, the BNL National Synchrotron Light Source II (NSLS II) and SLAC Linac Coherent Light Source II (LCLS II) projects shift the frontiers of control systems towards millions of control process variables and streaming rates of up to one million events per second. Similar requirements are introduced by other accelerator projects. Furthermore, recent progress in the development of light source detectors is leading to dramatic changes in the amount and complexity of available experimental data. These quantitative changes in requirements trigger two principal topics: demand for an efficient analytics-oriented database technology and consolidation of the control and experimental data management systems aiming to foster scientific discoveries.

Highly scalable data management and processing are two emerging topics in both industry and academics. Triggered by Google's web technologies, this domain represents an active factory of the new types of the analytics-oriented data storages, such as Casandra, CouchDB, Hbase, and MongoDB. Most of them are designed after Google's I/O stack: Google File System, Bigtable distributed storage system, and MapReduce processing framework.

Despite success in numerous projects, the web-oriented environment however cannot be directly applied to scientific applications. First, a Bigtable is a sparse, distributed multi-dimensional sorted map indexed by row key, column key, and a timestamp. It treats data as un-interpreted strings allowing flexible representations of structured and semi-structured formats. Unlike the web-oriented projects, scientific applications commonly relied on the multi-dimensional array-oriented data model. The map-based approach however is inefficient or incapable of addressing array-oriented queries. Second, the original MapReduce processing framework does not support complex iterative algorithms required by machine learning and scientific applications. This limitation prompted the new wave of hybrid data-intensive techniques merging or extending different data models and parallel paradigms: SciDB, GraphLab, SciHadoop, HadoopDB, MapReduce-MPI, and others.

To address the data challenges of the new modern light source facilities and the variety of technical solutions, we proposed an integrated configurable environment for connecting the different types of accelerator data with the modern large-scale analytics engines. The following sections subsequently overview the EPICS-based accelerator and beamline control system, the HDF5 file format [2] for storing experimental data of the modern light source facilities, two important representatives of the analytics approaches, the SciDB array-oriented parallel database [3] and the GraphLab distributed framework for processing complex algorithms on graphs [4], and finally the proposed large-scale integrated data management and analytics environment for accelerator and beamline experiments.

EPICS ACCLERATOR AND BEAMLINE CONTROL SYSTEM

EPICS is the Experimental Physics and Industrial Control System that has been started in 1989 at Los Alamos National Laboratory and in use today at a significant proportion of the particle physics laboratories

*Work supported by DOE contract DE-AC02-98CH10886
#malitsky@bnl.gov

ISBN 978-3-95450-116-8

in North America, Europe, and Asia. Its base infrastructure consists of two layers: distributed Input/Output Contollers (IOCs) and Operator Interface (OPI) applications connected by the Channel Access communication protocol. IOC provides a uniform interface to heterogeneous physical devices. According to the EPICS generic approach, each physical device is represented by a flat collection of process variables of predefined types. The existing EPICS-based projects usually deal with hundreds of thousands of such process variables. Monitoring and control of the distributed devices on the client level are provided by collection of the EPICS engineering tools and the physics applications. An archiver is one of the essential tools of this collection facilitating the analysis, operation, troubleshooting, and upgrade of the complex multi-system scenarios. Figure 1 illustrates the present EPICS two-tier environment.

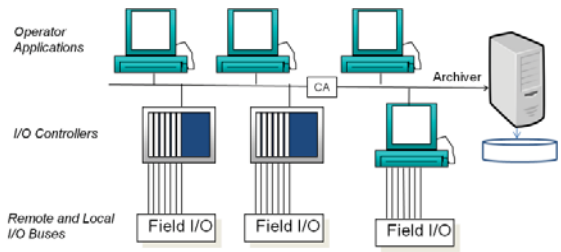

Figure 1: EPICS v3 two-tier environment.

The success of the EPICS infrastructure in the development and operation of numerous projects eventually generated new requests. One of these demanded features was the support of the three-tier model-based control architectures. Until recently, the middle layer servers have been implemented with other technologies like CORBA. In EPICS, this capability was hampered by a limited set of data types supported by the database and the corresponding communication protocol, particularly, by a lack of the structured data required by the high-level applications. To address this issue, the EPICS team started the development of the next version EPICS v4 adding the middle layer servers based on the novel concept of PV Data as show in Figure 2.

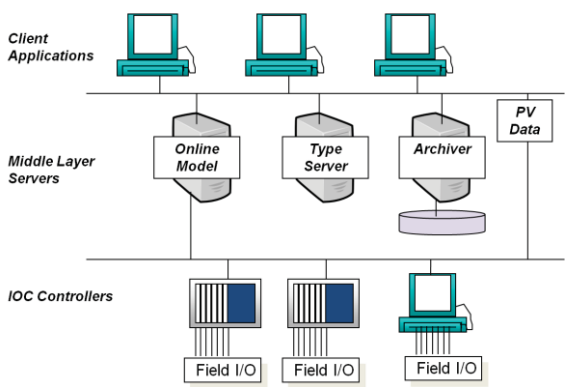

Figure 2: EPICS v4 three-tier high-level application environment.

PV Data is a generic self-described dynamic data container, allowing to generalize the present database records with the new hierarchical structures and to enhance all parts of the EPICS infrastructure: communication protocol, associated interfaces, and services. Moreover, it creates a basis for integrating data acquisition services of the beamline facilities.

HDF5 FILE FORMAT

HDF5 [2] is the latest version of the Hierarchical Data Format developed and distributed by the non-profit HDF group for storing scientific-oriented data of a wide range of application domains. The core of the HDF5 specification consists of a generic and compact data model based only on three primary concepts: dataset, data type, and group.

The HDF5 dataset is a multi-dimensional array of data elements with attributes and metadata including a description of the data elements, dimensions and all other information necessary for processing and storing data. Its structure and associated components are outlined in Figure 3.

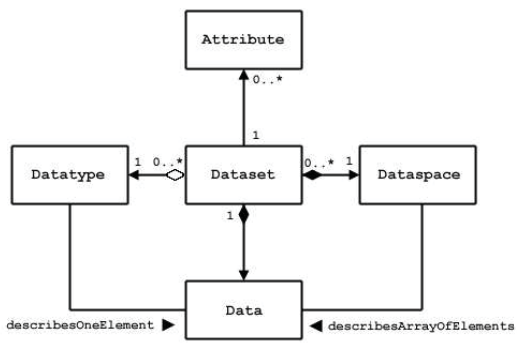

Figure 3: HDF5 dataset [2].

A data element of the HDF5 dataset can be any of several numerical or character types, small arrays, or even composite types similar to C structures. Its description and storage format is defined in the data type object. Data types are categorized into 11 classes. Each class is interpreted according to a set of rules and has a specific set of properties. The collection of data types can be extended by users. Moreover, the user-defined data type can be named, stored in a separate HDF5 file, and shared by other datasets located in the external HDF5 files. The HDF5 group serves for composing a collection of datasets and named types into the hierarchical structures.

The HDF5 software library is highly optimized and benchmarked on various environments ranging from laptops to massively parallel systems. For example, for boosting the data access performance, HDF5 added several important features like chunking, compression, and others. As a result, recently HDF5 is becoming a de facto standard for maintaining the complex experimental results produced in the modern light source facilities.

ISBN 978-3-95450-116-8

SCIDB ARRAY-ORIENTED DATABASE

SciDB [3] is a new open-source parallel database management system that brings the array-based analytics-oriented platform to large-scale scientific and commercial applications, such as astronomy, climate control, risk management and others. The multi-dimensional array is a fundamental data type used in various numerical computing environments (e.g. MATLAB) and is a natural model for describing both the time series of the control historical data and images in the beamline experiments. In recent years, there have been several approaches adding arrays into the relational databases. In contrast with the previous products, SciDB provides and implements a consistent array-oriented formalism including the conceptual model, a set of corresponding operators and a dedicated architecture.

In SciDB, an array can be created with the following command:

CREATE ARRAY Example <a1:integer, a2:float, a3:MyType> [Dim1=0:5 , Dim2=0:4]

The array definition contains three parts: name ("Example"), a cell structure, and a list of dimensions. Figure 4 illustrates what such an array might look like.

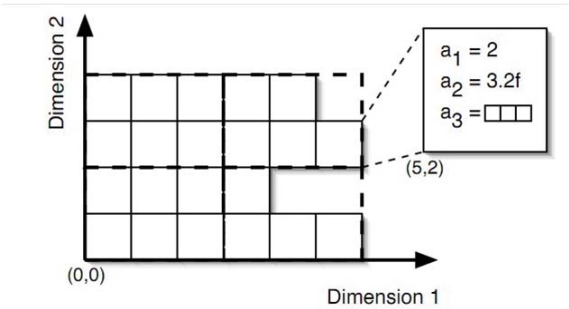

Figure 4: SciDB two-dimensional array [3].

A nominally rectilinear array can be sparse including empty cells and jagged edges. Every cell has the same data types for its values which can be scalars or composite structures defined according to the Postgres user-defined type (UDT) system.

For managing arrays, SciDB provides two flavours of the high-level languages: array query language (AQL) and array functional language (AFL). AQL is modelled after the SQL declarative language including two classes of queries, data definition language (DDL) and data manipulation language (DML). AFL is a proprietary language extending the AQL subset with the lower-level "physical" operators. The operators can generally be characterized based on where or not the operations manipulate an array in terms of its structure – the array's rank, and dimension indices – or by addressing the array's contents – the data values in attributes in cells. Some operators come in both structural and data-dependent forms.

The SciDB database represents the next logical step in the evolution of the modern column-oriented analytics platforms. In the traditional relational database management systems data values are stored and managed by rows. This organization is driven by the online transaction processing (OLTP) applications that generally create or modify one or a few records at a time. Conversely, online analytical processing (OLAP) applications often call upon the database to analyze selected attributes of vast number of rows or records. In such environments, the values for each single column are stored contiguously. The efficiency of the column-oriented approach has been demonstrated in the warehouse marketplace by products like Sybase IQ, Infobright, Vertica, and others.

The modern data warehouse systems are built after the OLAP cube model connecting data into multi-dimensional structures. In the context of the relational paradigm, the cube is usually created with the star or snowflake schemas. SciDB generalizes this approach by explicitly defining the array abstract data type (ADT) and map it into the common application programming interface and all parts of the multi-layer database architecture. Moreover, SciDB further extends the array model with the efficient chunk-based approach boosting the performance of associated data operations.

In recent years, there have been several approaches adding arrays into the relational databases, for example, using an array executor on top of blobs, or object-relational extensions. In order to properly compare and develop the most effective solution, the team of researches derived a benchmark called SS-DB. This benchmark was modelled after several scientific domains and includes a set of common queries and operations. The results clearly demonstrated the advantage of the SciDB explicit array-oriented architecture in comparison with the relation-based extensions.

GRAPHLAB COMPUTATIONAL FRAMEWORK

SciDB brings the array-oriented data model and associated use cases to the large-scale distributed processing environment based on the conventional database technology. Adherence to the existing architecture allowed to reuse and adapt the proven theoretical concepts and corresponding technical solutions, such as ACID, query processing and optimization framework, concurrency control and recovery techniques, high-level interface language and data description. At the same time, the rigid database methodology introduced the principle constraints reducing the scope of the potential applications and supporting algorithms. This limitation, which was consistent with a "one size does not fit all" principle, lead to the development of the alternative highly flexible data processing approach, MapReduce, based on only two functions for transforming and aggregating data.

ISBN 978-3-95450-116-8

Introduced by Google in 2004, the MapReduce programming model has been successfully adopted by many companies for the large-scale web applications and has triggered the development of a new class of so called NoSQL databases. The simple two-function model however was not able to address the full spectrum of scientific algorithms, particularly, requiring multiple asynchronous iterations and complex data structures. Recently, this topic became an active target of many research and commercial projects. One of most systematic approaches for solving the complex analytics tasks was suggested in the GraphLab project and implemented by the team of the computer scientists from the Carnegie Mellon University.

GraphLab is a distributed framework for developing a broad range of large-scale machine-learning applications, such as collaborative filtering, data-mining and computer vision. The framework consists of three main parts: data model, update function mechanism, and execution model.

In GraphLab, data is organized as a graph, the most universal and convenient structure of the scientific and machine learning models. In the distributed setting, the data is partitioned into multiple parts, where each sub-graph can be stored as a different file, representing a vertex in the global meta-graph structure (see Figure 5). This organization compliments the SciDB array-oriented data model and allows extending the hierarchical description of the HDF5 files to the larger scale.

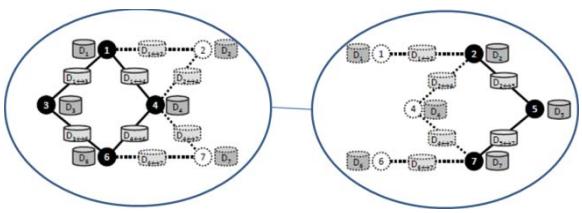

Figure 5: GraphLab meta-graph with two sub-graphs stored in the different files [3].

Similar to the SciDB approach, the algorithm-specific computation on the graph data is implemented via a mechanism of plug-in update functions. The GraphLab update function is a stateless procedure which is able to read and modify the scope of the input vertex including data associated with all adjacent vertices and edges. In addition to data processing, the function can return a set of tasks. Each task is a tuple consisting of an update function and the input vertex. All return tasks are placed in queue and eventually executed according to the scheduling order of the GraphLab execution engine.

This composite approach includes the update function mechanism and the execution queue provides the asynchronous iterative computation model for running parallel algorithms. To facilitate the development and computation of the complex algorithms, GraphLab enforces sequential consistency over update tasks with respect to the appropriate consistency model. The current version offers two alternative execution engines, Chromatic Engine and Locking Engine, which address these requirements using two different techniques, graph coloring and read-write locks.

INTEGRATED APPROACH

Figure 6 outlines the proposed environment designed to bring the modern distributed analytics engines to the analysis of the different types of accelerator large-scale data sets. Particularly, the approach addresses several actual tasks and issues:

- immediate upgrade of the existing EPICS archive system with the modern high-performance database technology;
- extension of the HDF5-based experimental file systems with the distributed quivery services.
- consolidation of the control and beamline data management systems;
- consistent transition to the new EPICS v4 three-tier infrastructure;
- enhancement of the proprietary retrieval engines with the modern analytics platforms for processing the complex array-oriented and data-mining algorithms.

The approach is designed after the three-layer I/O stack architecture consisting of the file system backend, middle-layer distributed data model and associated global services, and distributed computational framework. The data model represents the primary abstraction of the overall system and often determines the implementation of other components. In most data management systems, this architectural integrity is enhanced by the tighter integration of the different layers and encapsulation of the low-level interfaces. In our approach, we suggested to extend these systems with the driver framework enabling the interchange of the different file backends with the appropriate data processing engines.

In the context of the accelerator facilities, this approach resolves two conceptual tasks. First, it provides a consistent mechanism for in-situ integration of the multiple file formats used by the different teams and applications. Second, the driver-based mechanism facilitates the selection of the most optimal computational model (e,g, parallel database or MPI) for solving the particular category of the application tasks. The integrity of the overall system is still preserved by the common data model abstractions which determine requirements to the concrete backend and engine representations.

Many scientific data models can be described via three major components: multi-dimensional arrays, user-defined data types, and directed graphs. Among the various alternative file formats, HDF5 represents the most consistent and complete implementation of this data description. It has been successfully employed in various climate and space information systems. And our analysis and benchmark studies confirmed the advanced conceptual and technical features of this file format.

ISBN 978-3-95450-116-8

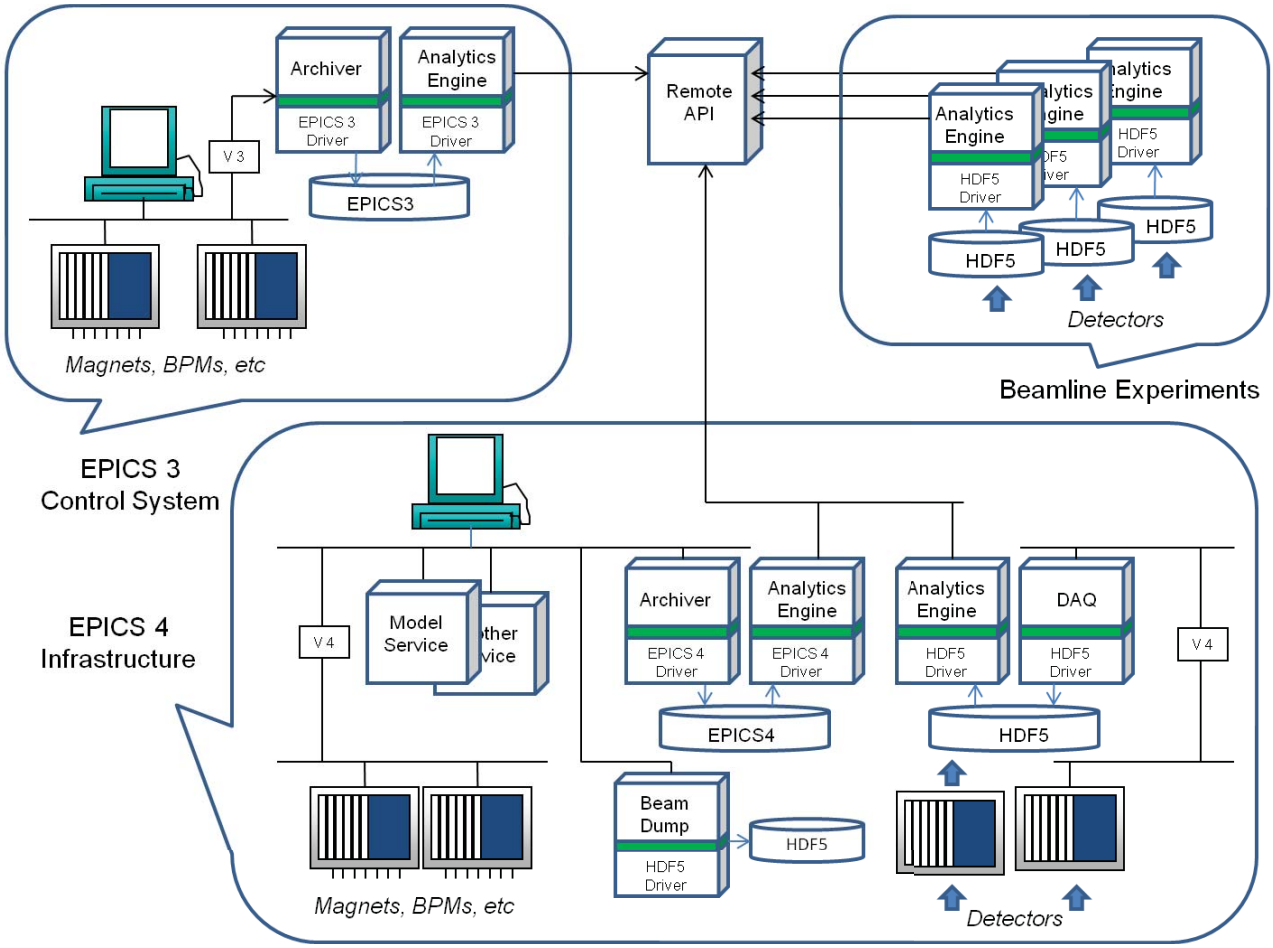

Figure 6: Large-scale integrated data management system.

With the exception of one project developed by Barrodale Computing Services, HDF5 was not generally considered as a backend of the traditional database management systems. The primary reason was associated with the mismatch between the relational and HDF5 scientific-oriented models. The new large-scale data processing tasks fundamentally changed the scope and requirements for the modern data management and analytics systems emphasizing the multi-dimensional arrays and graphs as the first class citizens. Following the incremental approach, the development of this integrated system has been started with more transparent and straightforward array-oriented applications including the large-scale correlation analysis of accelerator historical data based on the SciDB analytics engine and we expect to have a working prototype by the end of 2012 year.

REFERENCES

[1] L.Dalesio et al., "Experimental Physics and Industrial Control System Architecture," ICALEPCS'93. http://www.aps.anl.gov/epics

[2] M. Folk et al., "An Overview of the HDF5 Technology Suite and its Applications," AD'11. http://www.hdfgroup.org

[3] The SciDB Development Team, "Overview of SciDB," SIGMOD'10, http://www.scidb.org

[4] Y. Low et al., "Distributed GraphLab: "A Framework for Machine Learning and Data Mining in the Cloud," VLDB'12, http://graphlab.org

ISBN 978-3-95450-116-8