A NEW EPICS ARCHIVER*

N. Malitsky[#], D. Dohan, BNL, Upton, NY 11973, USA

Abstract

This report presents a large-scale high-performance distributed data storage system for acquiring and processing time series data in modern accelerator facilities. Derived from the original EPICS Channel Archiver, this version consistently extends it through the integration of deliberately selected technologies, such as the HDF5 file format and the RDB-based representation of the DDS X-Types specification. The changes allow scaling the performance of the new version towards data rates of 500 K scalar samples per second and to provide a common platform for managing both EPICS 3 records and EPICS 4 composite data types.

RATIONALE

Efficient data management and processing systems are essential tools in the commissioning and operation of accelerator facilities and large scientific experiments. Analysis of historical data is heavily involved in the troubleshooting process, detection and study of composite behaviour patterns, comparison and design of 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, requiring the acquisition and storage of heterogeneous data streams of variable rates from tens of thousands of distributed devices. Conventional technologies, such as relational database management systems, were not designed for such requirements. As a result, many control teams had to build new proprietary tools or project-specific extensions of existing technologies. For example, the Experimental Physics and Industrial Control System (EPICS) collaboration maintains more than four archiver systems. Constrained by the associated technologies and available resources, each of these solutions cannot address all requirements resulting in trade-offs among different objectives: performance, reliability, extensibility, and others.

The scale, data rate, and complexity of new light source facilities introduce new challenges and demands for new approaches. Particularly, the BNL National Synchrotron Light Source II (NSLS II) shifts 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 the EPICS v4 middleware triggers another request for supporting user-defined composite data types supported by the middle layer services. This change creates a natural path towards the consolidation of the control and experimental data management systems.

To address the data challenges of modern light source facilities and the new EPICS v4 infrastructure, we propose an integrated approach derived from the original EPICS Channel Archiver architecture [1].

INTEGRATED APPROACH

The development and application of the large-scale analytics-oriented data management systems is an emerging topic in academia and industry. Triggered by Google's web technologies, this domain represents an active factory for new products. Most of them are designed after Google's I/O stack: the Google File System, the Bigtable distributed storage system, and the MapReduce processing framework.

Despite success in numerous projects, the web-oriented environment, however, cannot be directly applied to scientific applications. Bigtable is a sparse, distributed 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 web-oriented projects, scientific applications commonly relied on the multi-dimensional array-oriented data model. In particular, the accelerator control and experimental data can be defined in terms of time series of structured data types, device events and detector frames.

Newer generations of databases have begun to address the data models and algorithms of large-scale scientific applications. For example, SciDB extended the traditional approach and explicitly introduced the array data type into the high-level language and all parts of the multi-layer database architecture. In comparison with SciDB and other databases, data from the experimental facilities however are not opaque and are explicitly represented by open repositories of the binary files in one of the scientific formats. As a result, their integration with the database management systems still requires the development of the additional extensions.

Analyses of different approaches confirmed the famous statement that “One size does not fit all” and lead us to a composite solution based on the integration of several open-source technologies. Figure 1 outlines the proposed data management system addressing the different versions of the EPICS control infrastructures end experimental facilities. In general, it is developed after the original EPICS Channel Archiver architecture with the integration of the HDF5-based backend and EPICS v4 distributed data services. The following sections provide a more detailed description of the particular extensions associated with the different subsystems.

*Work supported by DOE contract DE-AC02-98CH10886

#malitsky@bnl.gov

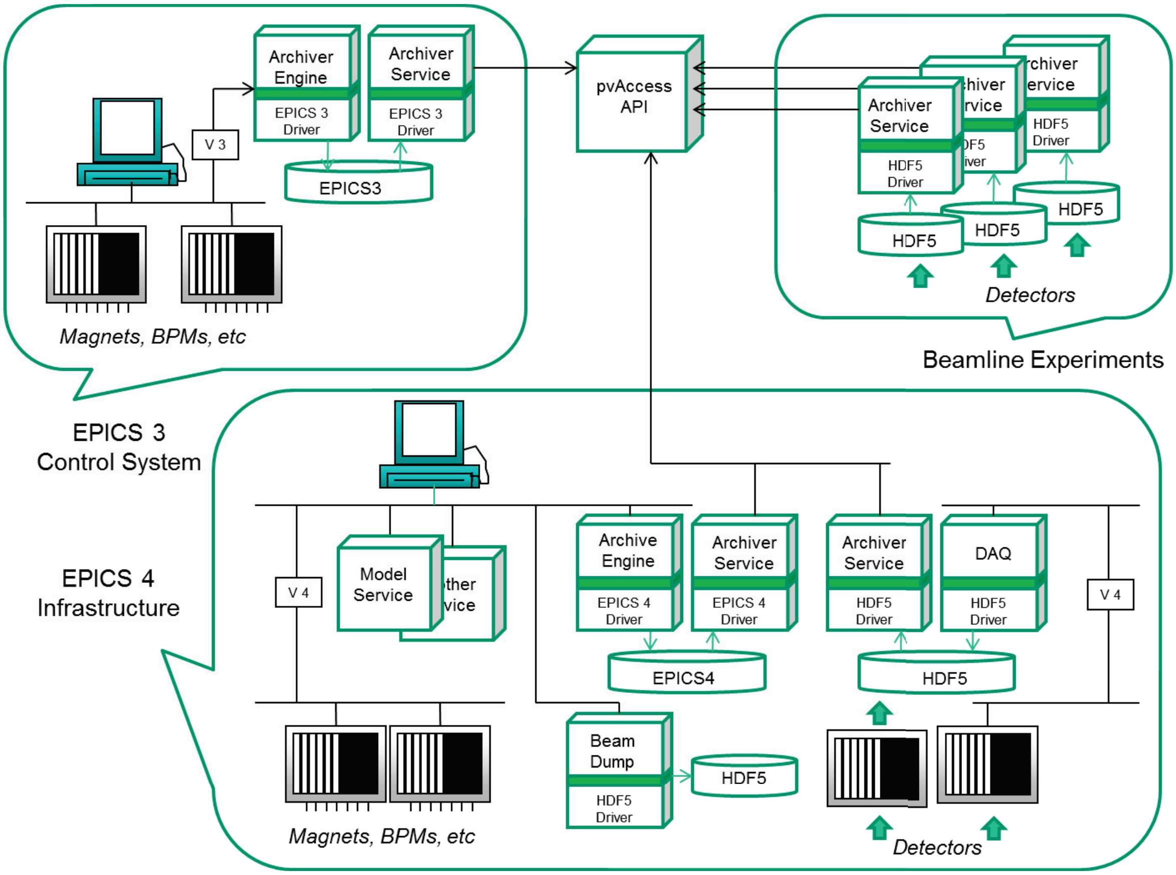

Figure 1: Large-scale integrated data management system

DATA STORE

The data store of the EPICS Channel Archiver is organized as a repository of data files maintaining time series of EPICS DBR data types. I/O access is asymmetric. The writing phase deals with the data sets of multiple channels accumulated during a short period of time. On the other hand, the data processing unit usually works with an extended history of a few channels. As a result, the format of the data file is proprietary and highly optimized to address both requirements. In particular, the data file can support multiple channels, implements chunk-based partitioning of time series and sequential navigation through the channel data. For managing multiple data files, the data store includes an additional index file maintaining a hash table of interval trees associated with each channel.

As such, the EPICS Channel Archiver format has been successfully employed in many EPICS-based control projects. However, it created significant constraints for adapting new user-defined data types introduced by EPICS v4 applications. As a result, it was decided to evaluate a more general solution, employing the latest version of the Hierarchical Data Format (HDF5). This decision was also consistent with other requirements arising from the beamline application domain.

HDF5 is one of the most popular data formats used in many scientific projects for storing experimental data. The core of the HDF5 specification consists of a generic and compact data model based on just 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.

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 datatype object. Users can extend this collection of data types. Moreover, the user-defined datatype can be named, stored in a separate HDF5 file, and shared by other datasets located in external HDF5 files. The HDF5 group is composed of hierarchical collections of datasets and named types.

In the context of the new EPICS Archiver integrated environment, the HDF5 format has introduced a generic approach for describing time series of both the EPICS v3 and v4 use cases with one common data model: variable-length sequence of channel-specific structures. Moreover, the HDF5 software library is highly optimized and implements the same features of the original EPICS Channel Archiver, such as the multi-channel support and

ISBN 978-3-95450-139-7

chunking. Table 1 provides performance results for the HDF5 format. This analysis has been conducted according to the common benchmark procedures writing and reading arrays of dbr_time_double's divided into chunks of different sizes.

Table 1: HDF5-based benchmark (i7/2.8 GHz, HDD)

Channels	Chunks/ Channel	Samples/ Chunk	Size	CPU time, s	Total time, s
Writing data					
1000	1000	1000	24 GB	46	904
1000	10000	100	24 GB	160	974
1000	100	10000	24 GB	42	827
Reading data					
1	1000	1000	24 MB	0.5	2.5
1	10000	100	24 MB	0.5	2.7
1	100	10000	24 MB	0.5	1.2

These studies clearly identified HDF5 as a winner among other alternative variants, including the file format of the SciDB database. The same task and associated studies, however, revealed a serious drawback of the present HDF5 specification, particularly, there was a lack of multi-file interfaces and services with one exception based on HDF5 proprietary external links. This problem has been naturally transformed into an advantage by integrating a repository of the HDF5 files with the original Channel Archiver indexing mechanism. As a result, this hybrid approach resolved the backward compatibility issues and provided a common platform for simultaneous processing of existing and new data files.

The extension of the HDF5 file format with the multi-file indexing service prompted consideration of a more general framework, data store middle layer, providing an efficient interface between the multi-file data repositories and processing engines. As noticed above, the time series of the EPICS v3 and v4 applications can be described using one HDF5-based data model: a one-dimensional array of channel-specific structures. Following this approach, we complemented the original indexing service with a type service maintaining a catalog of the normative and user-defined data types.

The definition of a common type system is a difficult task involving a trade-off between the scope and complexity of numerous approaches. Recently, this issue has been addressed by the Extensible and Dynamic Topic Types (X-Types) specification developed in the context of the OMG Data Distribution Service (DDS). The specification provides the comprehensive type system model that overlaps the scopes of the EPICS v4 and HDF5 data types. As a part of the Archiver project, the X-Types specification has been mapped into the relation database representation and used for the registration of the EPICS DBR types. Figure 2 shows the corresponding schema of the RDB representation.

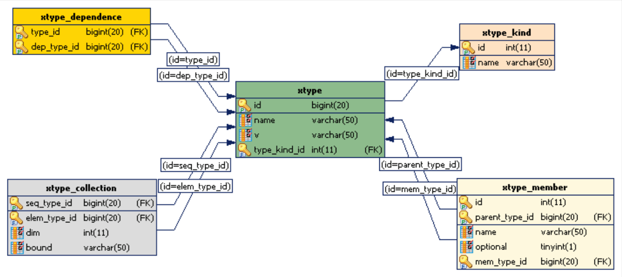

Figure 2: RDB type catalog based on the DDS X-Types type system model

DATA ACCESS SERVICE

The data repository represents the backend of the archiver system that can be accessed with the additional data service. In the original version, this service was based on the XML-RPC protocol using the HTTP transport mechanism for exchanging XML-based messages. The recent EPICS v4 version provides another solution based on the novel concept of PV Data, a generic self-described dynamic data container. The transition to the EPICS-based integrated three-tier infrastructure introduced a significant advantage leading to the reimplementation of the original service in the new framework.

The data service interface consists of four essential commands and associated messages:

- **archiver info**: common information used in other service commands, such as enumerations of alarm statuses and severities, and lists of data pre-processing methods, such as spreadsheet, averaged, and others.
- **archiver names**: a list of archiver names and paths to the index files used by this data service
- **channel names:** a list of channels matched to the input pattern. The result of the corresponding call contains channels names with associated time ranges.
- **channel values**: data of selected channels

The fourth message represents the most complex case, which relied on a special container, a heterogeneous array of parameterized elements. In the PVData framework, this approach was implemented with another generic container, a dynamic structure of self-described members. Other messages were directly mapped into the PVData basic data types, arrays of the homogeneous structures. The one-to-one relationship facilitates the implementation of the new data service. Moreover, the same approach was applied in a straightforward manner to clients such as the new archiver plugin of the Control System Studio.

REFERENCES

[1] K. U. Kasemir and L.R.Dalesio, "Overview of the EPICS Channel Archiver," ICALEPCS, 2001