19th IMEKO TC 4 Symposium and 17th IWADC Workshop
Advances in Instrumentation and Sensors Interoperability
July 18-19, 2013, Barcelona, Spain

Analysis of NetCDF-4 and HDF-5 scientific file formats for the data archiving of an ITER fast plant system controller prototype

R. Castro[1], J. Vega[1], M.Ruiz[2], D.Sanz[2], E.Barrera[2]

[1]*Asociacion EURATOM/CIEMAT, Madrid, Spain*
rodrigo.castro@ciemat.es,jesus.vega@ciemat.es
[2]*Grupo de Investigación en Instrumentación y Acustica Aplicada, Universidad Politecnica de Madrid, Madrid, Spain*
mariano.ruiz@upm.es, dsanz@i2a2.upm.es, eduardo.barrera@upm.es

Abstract- New fast plant system controllers are being designed and developed for ITER (International Thermonuclear Experimental Reactor). They require to manage and to process massive data that is being acquired at high sampling rates. This data has to be archived for being posteriorly analysed in order to rebuild several aspects of the experiment. Our group is currently involved in different works related with ITER, and more concretely related with their fast control technologies. ITER fast plant system controllers will require data archiving, and based on ITER requirements our group developed a data archiving solution based on NetCDF-4 [1] [2] technology. At this time, ITER is supporting HDF-5 [3] scientific file format, as data archiving solution for its control systems. In this sense, we have created a set of tests scenarios in order to analyse and compare both scientific data archiving solutions from ITER fast plant system controller requirements point of view.

I. ITER fast control data archiving requirements

ITER is currently the most important fusion machine project in the world. This machine is being designed to work with long pulses of about 1800 seconds. All involved fast control systems for diagnostics will require implementations that continuously acquire and process data during the pulse. These systems have to be able to manage data that has been acquired at a sampling rate of (at least) 100 KSamples/s per channel. This high data acquisition rate requires that data have to be managed in blocks, and requires that all involved systems are compatible with data blocks structures.
One of the main ITER requirements is that all data managed by a fast plant system controller has to be archived preserving its integrity, for high fidelity future reconstructions of the experiments. In this sense, only lossless codification or compression mechanisms can be applied. Acquired data has to be archived on remote and the transmission protocol has to be reliable and compatible with Ethernet networks. The data archiving process can have some delay, and temporary buffers can be used, but data has to be accessible for reading while is being written. In other words, the solution must guaranty at least one-writer/one-reader concurrency.
From data structure point of view, ITER is going to integrate many different diagnostic systems that are going to generate very different data formats. The implemented data archiving technology should be compatible with all these data formats and should be accessible from different system platforms. Additionally, the nature of the acquired information is based on time evolution data, so this technology should be valid to manage time-indexed data.

II. Fast control platform

The technology supported by ITER for the fast control system is based on RHEL (RedHat Enterprise Linux) operating system, and EPICS (Experimental Physics and Industrial Control System) [4] control framework. EPICS is an open source technology that includes a wide set of tools and applications to implement distributed control systems. Using Client/Server and Publish/Subscribe techniques to communicate between computers, EPICS is not only able to manage local control tasks but also to publish and monitor control parameters. Additionally, asynDriver is an extension for EPICS that simplifies and homogenizes the interfaces of the different devices to be controlled.
In our last projects, fast control systems has been developed using EPICS components with fast control extension

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

that have been developed by our group. This EPICS extension technology has the name of DPD (Data Processing and Distribution) and, on one hand, provides EPICS with massive data movement and processing capacities, and on the other hand, it provides FPSC developer with a modular framework that enables to configure a control system with the necessary functional components and data links. The fig.1 presents a possible internal architecture of a FPSC with DPD high throughput data links. Basically, it integrates a set of data acquisition modules, which distribute acquired data to data processing components, monitoring service, and data streaming modules that transmit continuously the received data to remote archiving servers.

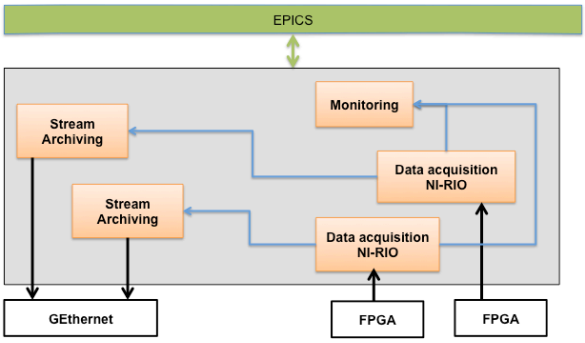

Figure 1: Possible configuration of a Fast Plant System Controller

III. HDF-5 and NetCDF-4

NetCDF and HDF-5 are scientific oriented file formats with a complete set of libraries and management tools (supported by a wide scientific community), and with the main objective of making data platforms independent and portable. In its last version 4, NetCDF storage layer is completely based on the HDF-5 technology, and its libraries work almost as a wrapper for HDF-5 libraries.
There is a list of important advantages that support the use of NetCDF-4 and HDF-5 as a FPSC data archiving format. The following list includes the most important ones:

- Self-description data management: any client can completely manage and know the format of the data stored in a file. This feature also helps to keep versions compatibility.
- Huge file size support: with no size limits apart from the operating system or file system ones.
- Optimized for data segment access: it perfectly fits the data access requirements for scientific data archiving in long pulse experiments.
- Data appending: it is possible to append new data to a file without copies or redefinitions.

As it been previously commented, NetCDF-4 technology it is completely over HDF-5 files but include some additional and interesting features for data archiving. In concrete, NetCDF-4 includes some modifications in order to increase the portability of data and data access concurrency. This last functionality provides read access to archived data meanwhile it is stored, and previously to file closing.

A. Test file formats

The structure of files used for our test is based on blocks of data, and in order to have comparable results, the same data file structure has been implemented for the three file formats (NetCDF-4, HDF5, and RAW). This last format is a binary sequential file, where data blocks are sequentially stored, and data is only accessible in sequential way. In the case of NetCDF and HDF-5 formats, both of them mange their own self-description languages. Thanks to it, information about the structure of data it is included into the archiving file, helping client systems to know how to access and to read the archived data. The Figure 2 presents the internal data structure for the three file formats to test. The first box presents the structure of the NetCDF file in its CDL (Common Description Language), the second box presents the structure of the HDF-5 file in a quite simplified version of its DDL (Data Description Language).

Basically the file is structured in:

- A global header composed by a set of global parameters (attributes), that are common for all the file, and include information such as: the version of the file, the descriptor of the experiment, the descriptor the stored data (for example the name of the acquired signal), or the reference time stamp for all the stored time stamps.
- A consecutive list of data blocks formed by a block header and a block payload. Block header includes

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

information such as: type of block, compression method (if used) and the size of the block payload. The advantage of this type of structure is that is compatible with any type of data to store and it is compatible with data compression mechanisms. The format used for the RAW file follows exactly the same structure.

```
HDF5 "WAV0_0_T000001_d1wave.h5" {
GROUP "/" {
   ATTRIBUTE "Version"
   ATTRIBUTE "pulseID"
   ATTRIBUTE "sourceID"
   ATTRIBUTE "timeRef"
   ATTRIBUTE "time_stamp_start_nanosecs"
   ATTRIBUTE "time_stamp_start_secs"
   ATTRIBUTE "title"
   DATASET "acqtimes" {
      DATATYPE  H5T_STD_U64LE
      DATASPACE  SIMPLE { ( 115 ) /
                  ( H5S_UNLIMITED ) }
   }
   DATASET "blockHeaders" {
      DATATYPE  H5T_COMPOUND {
         H5T_STD_I32LE "block_type";
         H5T_STD_I32LE "compress";
         H5T_STD_U64LE "size";
      }
      DATASPACE  SIMPLE { ( 115 ) /
                  ( H5S_UNLIMITED ) }
   }
   DATASET "payloads" {
      DATATYPE  H5T_VLEN { H5T_STD_I8LE}
      DATASPACE  SIMPLE { ( 115 ) /
                  ( H5S_UNLIMITED ) }
   }
 }
}
```

```
netcdf WAV0_0_T000001_d1wave {

  types:
    byte(*) vlen_t ;

  dimensions:
    acqtime = UNLIMITED ;

  variables:
    uint64 acqtime(acqtime) ;
    int block_type(acqtime) ;
    int compress(acqtime) ;
    uint64 size(acqtime) ;
    vlen_t payload(acqtime) ;

  attributes:
    :sourceID = "WAV0_0" ;
    :pulseID = "T000001" ;
    :title = "Acquisition channel data" ;
    :version = 1. ;
    :time_stamp_start_secs = 1000UL ;
    :time_stamp_start_nanosecs = 10000000000UL ;
    :timeRef = "UTC" ;
    :license = "Freely available" ;
}
```

Figure 2. HDF-5 and NetCDF-4 data structures used in the test. Both file structures are presented using their self description languages

IV. Test bench

In order to test the performance of the different archiving file formats, a common test bench has been implemented. As it is shown in the Figure 3, the test system is formed by: a FPSC, a 1 GEthernet network link, a remote archiving server, and an archived data-monitoring client. In an initial stage, the archiving server is waiting for incoming archiving data, and as soon as it is received, try to store it in its corresponding file. If this file doesn't exist, the archiving service is responsible for creating it in the specified format (RAW, NETCDF-4 or HDF-5). Data between FPSC and the archiving server is transmitted over TCP connections in order to warranty the reliability of received data.

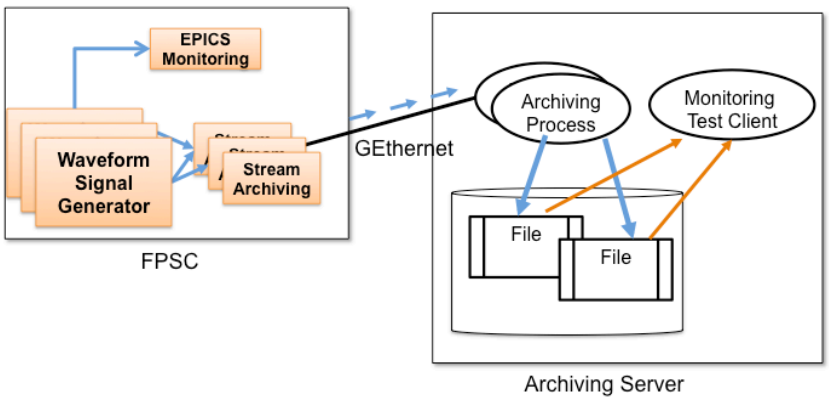

Figure 3. Architecture diagram of the test bench system

A. Fast Plant System Controller

FPSC is the part of the system that is going to be responsible of generating and streaming data to the remote storing archiving server. From the hardware point of view, this system integrates a PC Core i7 3770K with 16 GB RAM DDR3 1600 Mhz and PCIe 3.0 x16 data bus. And from the software point of view, the FPSC

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

integrates a Linux RedHat Enterprise 6 with real-time kernel, an EPICS base version 3.14.12.2, the asynDriver version 4-19, the DPD EPICS extension version 9.

The FPSC configuration for the test is formed by eight software waveform signal generators, which distribute generated data to the EPICS monitoring module and to the four data streaming modules.

B. Archiving server

The remote archiving server is the component of the system responsible of creating and managing the different file formats (RAW, NETCDF-4 and HDF-5), and it is also responsible of managing the incoming network connections and store all received data in the corresponding data file. Based on estimated incoming data throughput and duration of an experiment pulse (1800 seconds), it has been decided to assign a data file per pulse and per signal. This means that incoming data of different signals or different pulses are stored in different files. The hardware configuration of the archiving server corresponds to a PC Intel Core I5-660 - 3.33 GHz, 4.0 GB RAM at MHz DDR3, and 1TB SATA hard disk. And its software configuration corresponds to a RedHat Enterprise 6 with real-time kernel.

Additionally to the archiving process, a monitoring client has been included into the archiving server. This element can be activated in order to read and check online the validity of the file stored data. In the case of HDF-5 it is not possible because the format doesn't support concurrent write and read, and this test has to be done offline once the archiving process is finished. Of course, for test scenarios with intensive data store load this element is disabled and stored data is checked offline.

V. Test cases and results

There are two main objectives for the analysis. The first one is to evaluate the maximum generated data throughput that can be achieved for the different file formats, warranting a steady state archiving process during pulse period. The second objective is to analyse the storage overhead of the different file formats.

A. Maximum data throughput analysis

For this analysis, data throughput is being increased until a limit where the system gets unstable and generated data saturates intermediate buffers that corresponds to the data link between waveform data generators and data streamers. Buffer occupancy of data links is continuously measured and saturations can be detected. Apart of this measure, when the system saturates, important oscillations in the network throughput graph of the system-monitoring tool of Linux are observed. It is very important to remark that throughput limit of a 1 GEthernet link for a TCP connection is about 98 Mbytes/second.
A very important factor in the results is the size of the data block to be streamed and stored. A data block corresponds to a pack of generated data that is distributed in block. The presented results show measures for different data block sizes: 4KBytes, 16KBytes and 32Kbytes

Table 1. Maximum data throughput for different data block size

File Format	4KB/block	16KB/block	32KB/block
RAW	94,4 MB/s	98 MB/s	98 MB/s
NetCDF	33 MB/s	51,2 MB/s	92,8 MB/s
HDF-5	38,5 MB/s	72,8 MB/s	92,8 MB/s

As it is shown in Table 1, RAW format saturates perfectly the network link at different block sizes. On the contrary, NetCDF and HDF-5 present better performance at bigger block sizes. As it is showed in the Figure 4, these results seams to have a direct relation with CPU usage in the archiving server, and this last parameter is related with the frequency of data block reception and storage. For small size data blocks, NetCDF-4 and HDF-5 formats present quick CPU saturation on the archiving server and support low data throughput. On the contrary, for bigger data blocks, archiving service performance has an important increment. In this sense, with 32 Kbytes data locks, the archiving server is able to achieve the maximum bandwidth of the network. These measures support the results of other previous analysis presented by this team [5].

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

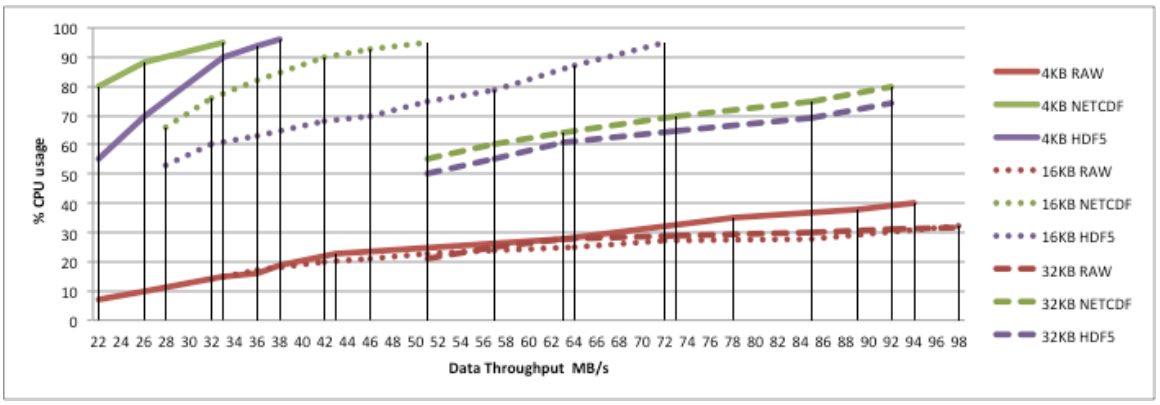

Figure 4. Graphic that represents the evolution of the percentage of use of CPU in the archiving server versus archived data throughput and for different data block sizes

B. Data overhead

For this analysis the same signal was stored in the three different format files during the same period of time. The resulting files were checked in order to verify that all of them had stored the same number of data blocks, with the same payload. As it is showed in the Table 2, the NetCDF-4 and HDF-5 have a close to 10% of overhead respect the original size of payload data to store, and NetCDF-4 add about 4% more overhead than HDF-5.

Table 2. % of size overhead for different file formats respect original payload size and respect RAW file size

File Format	RAW	NetCDF-4	HDF-5
Overhead respect payload	0,95%	14,06%	9,9%
Overhead respect RAW	0%	12,98%	8,8%

VI. Conclusions

In this work, NetCDF-4 and HDF-5 file formats have been evaluated in order to be used as data archiving file formats for fast data acquisition and control systems, on remote data archiving servers. NetCDF-4 and HDF-5 are two widely used formats in the scientific community based on their features, such as: managing huge amount of data, multiplatform support, auto content data description language, read access to portions of data, and the wide range of data access libraries and programming language support.
Focussing on the results of the analysis, NetCDF-4 and HDF-5 have intensive use of CPU, related with write operation rate. This effect is more evident in the case of NetCDF-4. From the overhead point of view, HDF-5 increases about 10% the original data size, and in the case of NetCDF-4, the overhead gets about 14%.
Focussing on ITER requirements and the results of the analysis, HDF-5 presents a crucial disadvantage for not being compatible with concurrent 1-writer / 1-reader schema, this means that, weather archived data couldn't be accessed until the end of the pulse, weather new files are generated during the pulse at some intervals. Apart from that, intensive use of CPU for archiving, forces the aggregation of data in fast controllers before being streamed for archiving. The presented measures support the results of other previous analysis presented by this team.

Acknowledgements

This work was partially funded by the Spanish Ministry of Economy and Competitiveness under the Project No. ENE2012-38970-C04-01

References

[1] M. Ruiz, J. Vega, R. Castro, D. Sanz, et al., "ITER Fast Plant System Controller prototype based on PXIe platform, Fusion Engineering and Design, Volume 87, Issue 12, December 2012, Pages 2030-2035
[2] R. Rew and G. Davis, "NetCDF: an interface for scientific data access", IEEE Computer Graphics and Applications 10 (4) pp. 76-82 (1990)

[3] "HDF-5 home web page", http://www.hdfgroup.org/HDF5/

[4] "EPICS home web page", http://www.aps.anl.gov/epics

[5] R. Castro, J. Vega, M. Ruiz, G. De Arcas, et al. "NetCDF based data archiving system applied to ITER Fast Plant System Control prototype", Fusion Engineering and Design, Volume 87, Issue 12, December 2012, Pages 2223-2228