Data model implementation in ITER data archiving system

R. Castro[1], Y. Makushok[2], L. Abadie[3], J. Vega[1], J. Faig[2]

1. CIEMAT Fusion program. Avda. Complutense 40. Madrid. Spain
2. INDRA Sistemas, S.A. Unid. de Sistemas de Control. Dirección de Tecnología Energética, Madrid, Spain[b]
3. ITER Organization, Route de Vinon-sur-Verdon, 13115 St. Paul-lez-Durance, France

ITER's CODAC archiving system currently manages three different sets of data: DAN, SDN and PON, that correspond with the three networks: Data Archiving Network, Synchronous Data Network, and Plan Operational Network. In this sense, ITER's CODAC data archiving system has been implemented to manage a wide variety of complex types of data and to support multidimensionality, dynamic resolution, metadata embedded types with header and footer sections, or user defined composition of types. This work describes ITER archiving data model, how it is able to fulfill ITER data requirements and keys of its implementation.

Keywords: Data archiving, CODAC, HDF5, UDA

1. Introduction

ITER data archiving technology has to archive different types of data coming from their systems. ITER data can be grouped in three main groups: DAN, SDN and PON[1][2], according to the network used for their distribution. DAN (Data Archiving Network) data come from acquisition systems, fast control and diagnostics. SDN (Synchronous Data Network) data are used for real-time communication. PON (Plant Operational Network) data come from EPICS process variables and are used for distributed and no real-time control communications.

For correct archiving, data models that fulfill ITER requirements have been created. This article describes these models and some keys of their implementation.

1.1 Requirements

There is a set of main requirements [2] which are common to the three types of data:

- Metadata support: There is a big set of information around sampled values that must be correctly archived. In some cases, metadata will be unique along the data acquisition process (static) but in other cases they can vary along time (dynamic).
- Continuous data acquisition: ITER is a long pulse device and data acquisition systems will be implemented in this sense. Data models must be ready to support continuous data acquisition independently of pulse length. In this sense, archived data can be read while they are being written.
- Complex data structure: Archiving system must support complex data types that include multidimensionality and data structures.
- Variable sampling rate: Along the data acquisition process it may be necessary to vary the sampling rate depending on some circumstances. For example, when a point of interest or when a specific phase of the pulse occurs.
- Control of versions: Data properties can change along ITERs life and the model must support versions

1.2 HDF5

All data models which are described in this article are currently implemented over HDF5[3] files. This file format is widely used in the research community and provides very important features such as: self-description information, huge file size support, optimized performance for data segment access, data appending and concurrent reading while writing. In the point 3.1, there is a brief example of HDF5 file information that implements SDN data model.

2. DAN model

DAN data are thought for data acquisition systems which require high sampling rate and bandwidth. This type of data is acquired or produced in fast control systems and is distributed through the Data Archiving Network; this is the reason of its name.

2.1 DAN specific requirements

rodrigo.castro @ciemat.es

There are some requirements which are specific of DAN data:

- Fast sampling rate: DAN data must support fast sampling rate acquisitions with MHz range frequencies.
- Multichannel: DAN data must support multichannel data acquisition devices where data are synchronously sampled in different channels.
- Variable dimensionality: During the data acquisition process, in some specific cases, changes on sample dimensionality are required. For example, a camera can change resolution when a point of interest is reached.

2.2 DAN stream

As it has been explained, DAN data come from fast sampling rate acquisitions systems where samples are not managed individually but they are rather structured in data blocks. The DAN archiving model follows the same paradigm and consecutive samples are grouped in data blocks with a unique timestamp and set of properties that are valid for all samples in the block.

Apart from the data block there is another important concept in the DAN model which is the stream. A stream is a time-ordered sequence of data blocks (of one or multiple channels), and a set of properties that affect them.

Stream and data block structuration has two main objectives. The first one is to provide metadata with different scope levels (from complete stream to data block) and, as a consequence, to allow different data properties per stream and per data block. For example, it is possible to set different data resolution per data block. The second is to avoid replication of metadata information meaning a significant reduction of the workload on data processing.

The figure 1 presents a diagram that shows the current data model of a DAN stream implementation in ITER data archiving system. The diagram represents how a stream is structured along the data acquisition time. It contains the following elements:

Stream metadata: It includes attributes that affect all the data in the stream. Some information that is typically archived at this level is: CODAC version, compression type, source host, DAQ operational mode, etc. This element can contain not only ITER required attributes, but also other user-defined set of attributes that can be different per stream.

Datatype: The sample is the main element in the DAN model, However, in order to support all the different experiment data types, the sample has been implemented as a multidimensional array of cells named 'items' (see figure 2). Items have a user-defined structure that can be: a primitive type, a multidimensional array, a multifield structure or a combination of all of them. The only limitation is that the item type must be equal for all channels along all the stream. In case of different data type per channel, multiples streams should be managed. The datatype section contains the definition of the item for the stream.

Channels metadata: The ITER data model must support data archiving from typical multichannel DAQ devices. In this sense, every stream can have multiple channels and each of them has perfectly defined characteristics, whether standard attributes, or user-defined attributes. Standard information for every channel includes: variable name, channel number, description, unit and calibration type.

Data block: A data block groups a set of complete samples from the different channels of the stream. A data block comprises a data block header that contains metadata affecting all its samples, and samples payload. However, at implementation level, data block metadata and their associated payload are independently managed to allow buffering and increase performance. A data block header includes fields such as: sampling rate, timestamp of the first sample, number of samples and multidimensional size (resolution). Apart from this set of standard fields, users can define their own attributes. These ones must be constant along the stream. As it is showed in the figure, samples from

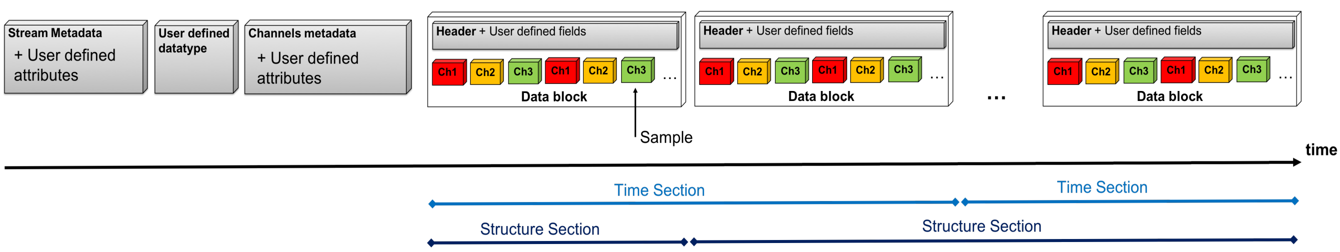

Figure 1 DAN data model diagram that presents a complete structure of archived stream elements along the time

different channels have interleaved organization into the data block.

Time and structure sections: A time section defines a sequence of data blocks with the same sampling rate and no data loss. In a similar way, a structure section defines a sequence of data blocks where dimensionality is constant. Both pieces of information are fundamental to improve data read performance.

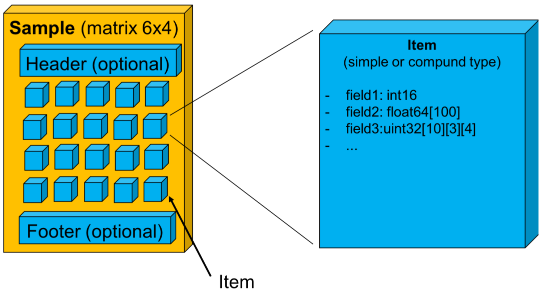

Figure 2 DAN sample structure

3. SDN model

3.1 SDN specific requirements

The main requirements of SDN data are:

- Up to 2KHz sampling rete
- Multi-topic: Data model must support multiple topics
- Timestamp precision at sample level: every sample will have its own timestamp with nanosecond precision.

3.1 SDN Data model

The basic element in SDN data is the topic. A topic is a variable where systems connected to SDN network can be subscribed and receive all messages distributed for this topic. The SDN data archiving system is ready to archive one or several topics in one HDF5 file.

Another important property of the SDN data model is that it must be sample oriented (not block oriented as DAN is) because SDN data require sample timestamp precision and every sample has to have its exact timestamp with nanosecond precision. With 2KHz sampling rate, it is a reasonable requirement from the performance point of view.

The figure 3 presents an example with a resumed text dump of an archived SDN topic "TEST_CPS_PF2_QVAL" in its HDF5 file. It shows the following sections:

Topic metadata: Includes information about the archived topic that affects all the samples in the stream. It includes information such as the name of the topic ("TEST_CPS_PF2_QVAL"), the multicast group where it is distributed through the SDN network ("239.0.154.177") or the version of the SDN topic ("9").

Topic data type: Defines the data type of the SDN topic along the stream. The type of a topic can change for different versions. The SDN data archiving system and HDF5 provide mechanisms to define the topic data type that can be a multifield structure of primitive types, multidimensional arrays or combinations of all of them. In the example, the type is a multifield of primitive types.

```
GROUP "TEST_CPS_PF2_QVAL" {
  ATTRIBUTE "group" {
     DATATYPE  H5T_STRING
     DATA {
     (0): "239.0.154.177"
     }
  }
  ATTRIBUTE "version" {
    DATATYPE  H5T_STRING
    DATA {
    (0): "9"
    }
  }
  ...
  DATASET "data" {
    DATATYPE  H5T_COMPOUND {
      H5T_STRING "header_uid";
      H5T_STRING "header_version";
      H5T_STD_U32LE "header_size";
      H5T_STD_U32LE "topic_uid";
      H5T_STD_U32LE "topic_version";
      H5T_STD_U32LE "topic_size";
      H5T_STD_U64LE "topic_counter";
      H5T_STD_U64LE "send_time";
      H5T_STD_U64LE "recv_time";
      H5T_STD_U64LE "id";
      H5T_STD_U64LE "timestamp";
      H5T_STD_U16LE "converter3";
      H5T_STD_I32LE "xval";
    }
    DATASPACE  SIMPLE { ( 290489 ) / ( H5S_UNLIMITED ) }
    DATA {
    (0): {
        "SDNv", "2.x", 48, 0, 9, 22, 0,
        1538721385000028716, 1538721385000093656,
        0, 1538721385000000000, 0, 3
      },
    (1): {
        "SDNv",
         ...
```

Topic payload: Binary data of the topic. The example presents the text dump of the first sample with values for the different fields.

4. PON model

4.1 PON specific requirements

The is a set of specific requirements of PON data:

- Low sampling rate: Variables have a frequency up to 10Hz.

Figure 3 PON data model diagram

- 50K variables per file: Archiving system must be ready to archive up to 50K variables in one file.
- Interleaved state marks: New status and severity values can be defined along the stream
- Value structure composed by the fields: timestamp (nanosecond precision), value, status and severity.

4.2 PON Data model

The basic element in PON is the EPCIS PV (process variable). PON model is sample oriented because of the required sample timestamp precision.

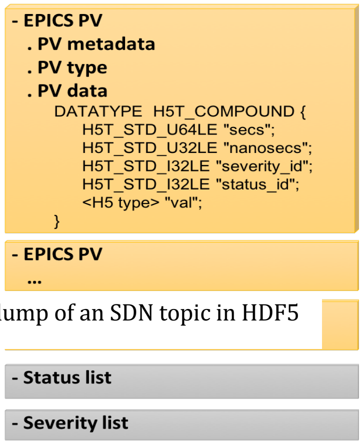

Figure 4 Example of text dump of an SDN topic in HDF5 file

Figure 4 presents PON data model that includes the following elements and sections:

EPICS PV: EPICS control variable whose data are archived.

PV metadata: Information about the EPCIS PV that is valid for all archived samples in the file.

PV Type: Type of data of the field "value" of the PV. It can be any of the valid value EPICS types.

PV Data: Sequence of PV samples. Every sample includes: the timestamp divided in two fields "secs" (for seconds) and "nanosecs" for (nanoseconds), a valid severity, a valid status and the value of the variable.

Status list: List of valid values for status field. This list can grow if one of the PVs publishes a new status.

Severity list: List of valid values for severity field. This list can grow if one of the PVs publishes a new severity.

5. Implementation keys

All the described data models are implemented in the current ITER CODAC Core System. It is not the objective of this article to present performance or test results, but some of them can be found published [4].

At this point, some specific implementation keys are described.

Data model flexibility

DAN and SDN types can be defined by data producers. HDF5 files include a complete mechanism to define new data types. This mechanism includes multifield and multidimensional structures. Consequently, it is possible to archive data from many different types of sources: multichannel DAQ cards, dynamic resolution cameras, spectroscopy DAQ systems, etc. The user-defined data type is transmitted to the data archiving service as part of stream metadata and it is automatically translated into HDF5 file.

Read while writing

HDF5 incorporates metadata and payload synchronization mechanisms that allow read while writing. Additionally, implemented backends (for DAN, SDN, and PON data types) include not only timeout mechanisms, which ensure in time flushing of data into file independently if data come from slow or fast sampling rate, but also refresh policies to warranty coherent and updated data.

Continuous data acquisition

The data model supports not only continuous data archiving for a long pulse, but also between pulses. The current ITER data archiving system implements automatic rotation technology to support nonstop data archiving. This mechanism creates a new file in parallel (while data is archived) with a copy of the main properties and structures. When the new file is ready, it is assigned to archiving while the original file is closed. In the case of PON data, a file can contain until 80K variables (80K structures) which have a long creation time. With this implementation technique, it is granted that no data loss will occur during the rotation process.

Zero copy model

For writing, operations don't require any type of data processing. Payload binary data are written directly to file as they are received from data acquisition systems. The HDF5 technology manages metadata and data types independent of their archived binary payload. At the same time, this feature ensures well defined structures and types, and fast individual or block data access. For reading and processing, every data field is accessible in a transparent way from its data model definition.

Dynamic sampling rate and resolution

In some diagnostics it is important to configure different sampling rates or resolutions depending on experimental factors. This feature allows to save bandwidth and to have higher resolution measures when it is really necessary. The ITER data model incorporates sampling rate and resolution attributes in every data block header, so they can be modified along data acquisition when it is necessary. This functionality is used by cameras or DAQ systems that vary data resolution based on experiment events.

The main issue when different sampling rate and data resolution is data access performance. When data have the same sampling rate and same data resolution, they can be accessed in big chunks (big payload binary chunks), but if one of these properties is changed, it would be necessary to read every data block. In order to preserve data reading performance, DAN files incorporate two new elements: time sections and structure sections. DAN data access library analyse structure and time sections to warranty homogeneous data and optimize the number of reading operations.

6. Conclusion

In this work, the data model which is currently implemented in ITER CODAC Core System is presented. This model covers the three ITER CODAC types of data (DAN, SDN and PON). Although the differences among these three data types, the implemented data model has fulfilled their main requirements.

References

[1] R. Castro, L. Abadie, Y. Makushok, et al., “Data archiving system implementation in ITER's CODAC Core System”, Fusion Engineering and Design, Volumes 96–97, 2015, Pages 751-755, ISSN 0920-3796.

[2] Gheni Abla, Gerd Heber, David P. Schissel, Dana Robinson, Lana Abadie, et al., “ITERDB—The Data Archiving System for ITER”, Fusion Engineering and Design, Volume 89, Issue 5, 2014, Pages 536-541.

[3] HDF5 File Format Specification, https://support.hdfgroup.org/HDF5/doc/H5.format.html.

[4] B. Penaflor, D. Piglowski, C. Liu, et al. “Evaluation of the ITER data archiving network API on the DIII-D tokamak”, Fusion Engineering and Design, Volume 130, 2018, Pages 11-15,