Common IBA Data Format

Version 17-10-2020

Lead authors:

G. Dollinger (UNI BWM), N. Barradas (IST) and E. Alves (IST)

Work Package 6: Data Management (Lead Beneficiary IST)

Task T6.1: Common Standards for Data (Lead Beneficiary UNI BWM)

Milestone MS 6: Common IBA data format (M12, delayed)

1. Summary

A major goal of RADIATE is to establish a compatible data format that could help the Ion Beam Analysis (IBA) community to store and share their data guarantying its future use in the framework of the European Open Science Cloud. The format must allow the archive of the raw data together with all the related experimental information in a Metadata structure supporting all the different types of IBA data. In addition a fast and reliable mining procedure must be ensured.

We propose a common IBA data format that can be used to process data with common software and store data in internal and external storage capabilities.

To satisfy all types of experimental arrangements and techniques different organizations are used depending on the kind of data and analysis procedures. We will consider 3 levels of data storage to be defined depending on the complexity and number of detectors and parameters to be handled.

2. Common data Format

One common part for all types of data formats will be a header, providing all the necessary information required to fully reproduce the experimental conditions like scattering geometry, beam energy and dose, detector type, detector arrangements and calibration parameters. The data (of pre-defined data type) follow the header information.

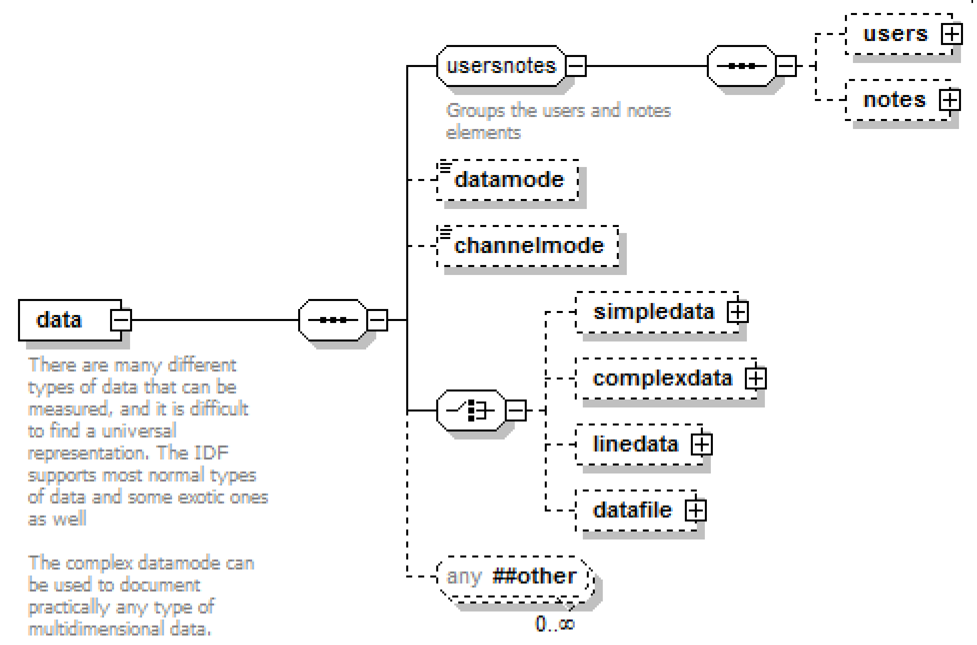
The IBA Data Format (IDF) is proposed the common format which can be extended by the users to support any other type of data not in the original design. In fact, it can support all types of data described below, albeit in some cases in a verbose way. By design, IDF supports all most common metadata needed for IBA, including:

Users: every single group of tags can start with the group “users”, where information on e.g. sample owner, experimentalist, and data analyst can be given. Any number of tags “user” can be given.

Notes: To introduce any needed text information. Every single group of tags can have the group “notes” just after the group “users”, where any information or comments can be given. Any number of tags “note” can be given. It can include the technique (RBS, ERDA, PIXE, etc), the reactions expected and even which cross section file would be appropriate for the given reaction as well as the data analysis done, including simulations (and which code and which code options were used), the Physics and models used in the analysis (e.g. Rutherford scattering, or screened Rutherford, energy loss, straggling, plural and multiple scattering models used, pileup, etc).

This means that one IDF file can have not only the data together with the experimental conditions and other metadata (such as lab and equipment used), but it can include all the data processing and analysis done a posteriori. The flowchart below shows the organization of the IBA Data Format proposed using the data tag:

element **data**

diagram	 usersnotes Groups the users and notes elements users notes datamode channelmode data There are many different types of data that can be measured, and it is difficult to find a universal representation. The IDF supports most normal types of data and some exotic ones as well The complex datamode can be used to document practically any type of multidimensional data. simpledata complexdata linedata datafile any ##other 0..∞
namespace	http://idf.schemas.itn.pt/
properties	content complex
children	users notes datamode channelmode simpledata complexdata linedata datafile
used by	element **spectrum**
annotation	documentation There are many different types of data that can be measured, and it is difficult to find a universal representation. The IDF supports most normal types of data and some exotic ones as well The complex datamode can be used to document practically any type of multidimensional data.

For each spectrum/map/experiment, the user has to choose (using tag <datamode>) which type of data will be used: <simpledata>, <complexdata>, <linedata> or <datafile>.

The 3 levels of data complexity can be grouped in the following categories:

a. Simple data: Single detector with a single parameter
(e.g. standard RBS,NRA or PIXE measurement)
This requires just to collect one histogram for the detector, channel based. Header contains: detector type, analyzing angle, solid angle of detection and its energy or time calibration. The suggested data format for this kind of measurements is IDF (common IBA data format) as suggested by Barradas et al, [1].

In IDF the experimental parameters are saved together with the measured histograms in an XML -environment (an example is given in [2]). If data are stored in another data format by a certain institute, there should be a program provided that makes possible the transformation of the data to the IDF data format.

b. Line Mode: PIXE data (yield per line)

For techniques such as PIXE or PIGE, it is usual to have line yields, i.e. the yield for a given X-ray line, extracted via some method from yield per channel data. Again the same conditions of the previous simple data applies here.

c. Complex data

c.1 *Multi detector arrangements with single parameter readout*

(e.g. standard RBS measurement with up to ~100 detectors simultaneously driven enlarging solid angle of detection and making angular dependent measurements possible by using multiple detectors of single parameters)

The detectors operate independently. One event contains just a single parameter of one of the detectors. In this case, the data of each detector are stored in a separate histogram. The file header contains the experimental conditions for all detectors in a similar way as for case 1, numbered for each detector. The histograms from all detectors are included in the data structure. The IDF format [1] is able to handle these multi detector measurements of single parameters per event and is also suggested as a common data format in this case.
If data are stored in another data format by a certain institute, there should be a program provided that makes possible the transformation of the data to the IDF data format.

c.2 *Multi detector arrangements with multiple parameter readout*

Data management has to be extended, when a detected particle produces two or more parameters per event. This is the case e.g. in a standard heavy ion Elastic Recoil Detection (ERD) analysis where at least two parameters are needed for the particle identificaton --- ΔE-E or TOF-E.
Further parameters simultaneously acquired may be: position or angular measurements, a multiple combination of several energy E, energy loss ΔE and/or TOF (time of flight) signals, pile up rejection signals from fast timing, etc..
In this case, there often is a need for an offline data optimization and multi-level data filtering that requires event by event data storage. The data format has to contain a header similar to the single detector arrangement as described before in cases 1 and 2. However, the data will not be filled directly into histograms but the data are stored event by event in a list mode structure. This means a matrix with the first row containing the event number and each line to this event number contains the signal values of all the detector parameters (=ADC channels). Histograms may be filled after data filtering (online or offline). 1-dim histograms filled from the list mode data may be adapted to the data format of [1] but 2-dim or higher dim histograms may be defined and filled according to the requirements of the performed experiment.

The IDF format [1] is able to handle these list mode measurements, including time stamp of events.

c.3 *Multi parameter multi detector arrangements*

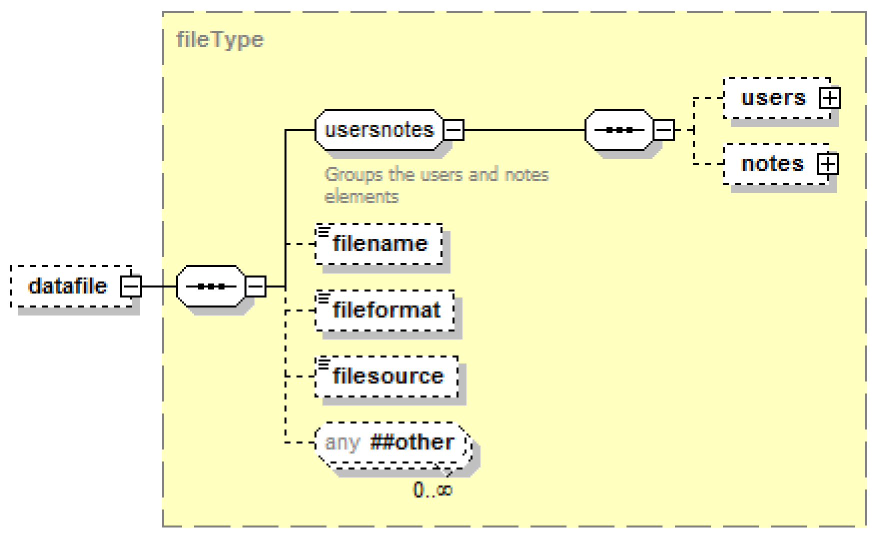
This type of measurements with a large number of parameters and detectors in the system the simple list mode files can get quite large. In particular, if gas or silicon strip or pixel detectors are used and a detected particle produces a signal just in more than one but only a few parameters (stripes, pixels, energy or timing signals), only a few parameters of a single event will contain relevant data. Thus, most of the written data in the list mode data format become zero's.

In this case, the data structure has to be adapted to reduce the amount of data avoiding the zero´s in the list mode data collected from all detector elements that did not receive any true signal for a trigger event. Also a time stamp has to be added to the data when operating a multi trigger arrangement.

The IDF data type <complexdata> has inbuilt support for the list mode, with the tag <timestamp>. It also supports as many x axes as needed (for instance, time of flight and energy), and as many measured quantities as needed.

It is clear that this <complexdata> supports data with nearly any type complexity, not only list mode. In principle, 2D and 3D maps can be stored with <complexdata>. This leads to very large files and may not be an efficient solution. To address this, IDF has one more way of introducing data, which is to refer to an external "data file" (see flowchart):

element **data/datafile**

diagram	

An external file is referred to, including the file name, the file format, and the file source (e.g., the program that originally produced it). That is, the IDF file would in practice contain the metadata; but it would be associated to the raw data via the <datafile> tag. Ideally, they would be stored in the same place (directory, drive, disk, …)

As said IDF can be extended ad infinitum with tags developed by users. Furthermore, any group of IDF tags can include further user-defined tags and for a given sample more than one spectrum can be given. For instance, if RBS with protons, with 4He, then ERDA, then NRA, were all collected from the same sample, then the IDF file can include all that data; and include all simulations as well.

Furthermore, one IDF file can have data for more than one sample. So, a single IDF file could contain all data collected in one day; in one month; or belonging to one experimental run, i.e. one group of samples.

To handle large amounts of data CERN has developed a “Root” based system structure which can be adapted to any experiment and detector arrangement [3]. Root is an open-source data analysis framework for data analysis and Input/Output management developed at the high energy facility CERN. Thus, Root can also be used for complex IBA setups and data managing in combination with IDF.
In particular ROOTS provides:- multi parameter data processing
- multi level data filtering
- fitting
- visualisation
- data storage in an optimum compressed tree system (quasi list mode data)
- independent on raw data input (VME, CAMAC, anything with LAN, USB, ...)
- all experimental parameters are written in the raw data file when starting a run

Examples for the power of root data processing can be found e.g. in [4,5].
Description of the data structure and all functions of root is given in [6].

3. Summary

IDF Is a versatile data format allowing handling ion beam data by the scientific community in a user friendly environment. It offers the possibility to store the raw data together with the experimental conditions and other metadata (such as lab and equipment used), as well as all the data processing and analysis done a posterior.
For complex data structures, Root based data structures and other, dedicated data structures are used as common data structures for complex detector arrangements. They are well adapted to the complex analysis software needed to handle the data. Thus, it is recommended to store the data structures with the well adapted analysis software in combination.
The IDF data format allows in principle also the data storage of complex detector setups and their complex data structures and it may serve as a common data structure for these complex data in future. However, transformation software has to be developed to transfer Root data files to the IDF format back and forth. As long as data manipulation and analysis software is based on other software, e.g. Root, data structures adapted to the specialized analysis software (e.g. Root based) is also recommended as common data structure
Since detector setups and data structures are under ongoing development the common data structure will be further adapted to the need of the community.

[1] N. Barradas et al,, NIM B268 (2010) 1824.
[2] http://idf.schemas.itn.pt/IAEA25_NDF.xml
[3] https://root.cern.ch
[4] P. Reichart et al., NIM B **371** (2016) 178-184.
[5] S. Eschbaumer et al., NIM B **406** (2017) 10 - 14.
[6] https://root.cern/primer/