

Analytical Cytometry Standard

NetCDF Conventions for List Mode Binary Data File Component

Proposal version 080112

January 2, 2008

Abstract

The Flow Cytometry Standard (FCS) specification has been adopted for the common representation of flow cytometry data, and this standard is supported by all analytical instrument and third party software suppliers. It includes data, metadata and analysis components within the same file. However, metadata and analysis components, if included at all, are not recorded in a standardized fashion or in sufficient detail for use by independent parties.

We are proposing to address these shortcomings via series of related data standard proposals. Different components describing analytical cytometry experiments would be stored reusing standard formats such as XML or RDF, and all the components would be bundled together into an Analytical Cytometry Standard (ACS) container.

This document demonstrates how the Network Common Data Form (netCDF) data format could be used to capture the most important component of FCS files: the recordings of the fluorescent or light-scattering properties of hundred of thousands of individual particles. NetCDF is a set of freely available software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. Many groups, including NASA and the Los Alamos National Laboratory, have adopted netCDF as a standard way to represent some forms of scientific data. Conventions have been developed on how to use netCDF to store data for climate and weather forecast, aviation research, geographic information systems, ocean and atmosphere research and descriptions of molecular dynamics simulations. This document introduces the netCDF format and describes how to use netCDF to store list mode data files.

Status of this Document

This document is an unapproved draft of a proposed standard that is intended for an internal review by the International Society for Analytical Cytology (ISAC) Data Standards Task Force (ISAC DSTF). As such, this document is subject to change and must not be utilized for any conformance/compliance purposes.

Josef Spidlen, jspidlen@bccrc.ca
Robert C. Leif, rleif@rleif.com
Wayne Moore, wmoore@stanford.edu
Ryan Brinkman, rbrinkman@bccrc.ca

Table of Contents

Introduction.....	4
Rationale for Changes to FCS 3.0.....	4
What is NetCDF.....	5
Available NetCDF Software.....	5
Benefits of Using the NetCDF for Binary Data.....	5
Reusing Existing Standards.....	5
Support for Future Instruments and Other Analytical Cytology Data.....	6
Availability of Tools.....	6
Performance.....	6
NetCDF Conventions.....	6
ISAC / List Mode 1.0 Conventions.....	7
1. Introduction.....	7
1.1. Goals.....	7
1.2. Scope.....	7
1.3. Terminology.....	8
1.3.1. Parameter.....	8
1.3.2. Electronic Event.....	8
1.3.3. List Mode Data File.....	8
1.4. Informative Overview.....	8
2. NetCDF Files and Components.....	9
2.1. Filename.....	9
2.2. Data Types.....	9
2.3. Dimensions.....	9
2.4. Variables.....	10
2.5. Packed and Compressed Data.....	10
2.6. Global Attributes.....	11
2.6.1. Identification of Conventions.....	11
2.6.2. Identification of Files.....	11
2.7. Variable Attributes.....	11
2.7.1. Variable Names.....	12
2.7.2. Variable Range.....	12
2.7.3. Variable Units.....	13
3. Miscellaneous Compliance Requirements.....	14
3.1. Data Sets vs. Data Files.....	14
3.2. Underlying Data Format.....	14
Appendix A – Attributes.....	15
Appendix B – Examples.....	16
Binary List Mode Example Files.....	16
Appendix C – Software Provided with these Conventions.....	16
ISAC / List Mode 1.0 Conventions Checker.....	16
FCS to NetCDF Converter.....	16
NetCDF to Sequential Binary File and XML File Converter.....	16

Appendix D – NetCDF vs. Data File Requirements.....	17
1. Encode the required information to interpret a cytometry experiment.....	17
2. Facilitate the reproduction of a cytometry experiment.....	17
3. Efficiently store cytometry list mode data.....	17
4. Transparently store text-based data.....	17
5. Facilitate support for other analytical cytology data.....	18
6. Each type of information shall be uniquely identifiable.....	18
7. It shall be possible to resolve data files over the Internet.....	18
8. Each type of information shall only be stored in one file format.....	18
9. Support for future instruments.....	18
10. Provide semantic meaning to all required/standardized information.....	18
11. Extensible by third parties.....	18
12. Extensions shall be separated and removable from normative parts.....	18
13. Ensure that metadata can always be located.....	19
14. Support for use cases where new metadata are being added subsequently.....	19
15. Several instances of mutable information may share immutable information...	19
16. Some information may be encoded by a referenced semantic model or an existing standard.....	19
17. It shall be possible to refer to multiple semantic models or standards.....	19
18. Use a manufacturer-independent format.....	19
19. Use a programming language-independent format.....	19
20. Use a file/operating system-independent format.....	19
21. Use a simple format oriented on data interoperability.....	20
22. Describe the standard unambiguously.....	20
23. There shall be an open-review period before adopting the standard.....	20
24. There shall be a mechanism to formally validate the standard.....	20
25. The standard shall be public.....	20
26. An implementation of the standard shall be possible without charge.....	20
27. The implementation of the standard shall be non-restrictive.....	20
28. Reuse existing standards.....	21
29. Conformance to the standard shall be (easily) testable.....	21
30. The file format shall be as self-explanatory as possible.....	21
References.....	21

Introduction

Rationale for Changes to FCS 3.0

First developed in 1984, the Flow Cytometry Standard (FCS)¹ specification has kept pace with many years of technological evolution. It has been adopted for the common representation of flow cytometry data, and this standard is supported by all analytical instrument and third party software suppliers. Scientists can choose among instruments and software with no major compatibility issues for the raw fluorescence values that FCS captures.

FCS includes data, metadata and analysis components within the same file. However, metadata and analysis components, if included at all, are not recorded in a standardized fashion or in sufficient detail for use by independent parties. This is assuming that independent parties can access experimental results, as important data sets supporting publications are almost invariably unavailable. Finally, if metadata annotation takes place at any time subsequent to data capture, the all-inclusive format of FCS necessitates the generation of a new version of the file, which replicates the (hopefully unmodified) primary data.

Changes proposed in this document and related proposals (e.g., Gating-ML², Transformation-ML³, ACS containers⁴) are designed to address these shortcomings through the incorporation of technologies from standards bodies such as the World Wide Web Consortium (W3C)⁵, Object Management Group (OMG)⁶ the Dublin Core Metadata Initiative⁷, and the Unidata Community⁸, that were not available when FCS was conceived. The current proposal assumes an Analytical Cytometry Standard (ACS) container for different components describing analytical cytometry experiments. The ACS container may include XML-based components (such as based on MathML⁹, RDF¹⁰, XHTML¹¹, SVG¹², XSD¹³, CytometryML¹⁴, and others), as well as other types of data and formats including images, text-based documents, vendor-specific information (Figure 1).

Within this proposal, we only demonstrate how the netCDF binary data format could be used to capture the most important component of FCS files: the recordings of the fluorescent, light-scattering, and/or other properties of hundred of thousands of individual particles. Such a binary file would now be separately maintained as an immutable sequence of bytes¹⁵, identified with a globally unique identifier. Metadata as well as the analysis of the data (e.g., compensation, transformation and gating) would be recorded within separate files using technologies such as XML¹⁶ and RDF^{10, 16}. The actual checklist of what should be described is contained in the Minimum Information for a Flow Cytometry Experiment (MIFlowCyt¹⁷).

The support of these standards by additional software tools, journals and scientists will bring flow cytometry data to the semantic web^{18, 19} and significantly facilitate the reproduction of experiments and clinical measurements. Most importantly, these changes will allow scientists and software agents to search, automatically process, and in particular understand both flow cytometry data and metadata.

What is NetCDF

NetCDF²⁰ is a complex solution to create, store, access, and share array-oriented scientific data. It consists of a general data model, a file format, and a set of Application Programming Interfaces (APIs) and libraries for various programming languages. It allows for the storage and retrieval of data in the form of n-dimensional arrays. Auxiliary information about the data, such as variable labels or what units are used, may be stored with the data. Generic utilities and application programs can access netCDF datasets and combine, analyze, or display specified fields of the data. The development of such applications has led to improved accessibility of data and improved re-usability of software for array-oriented data management, analysis, and display. The netCDF format is:

- Portable and platform-independent – the netCDF file format is XDR-based²¹ so that data written on one platform can be read on other platforms.
- Fast with within file access – the netCDF file format uses indexing structures so that small subsets of a large dataset may be accessed efficiently, without first reading through all the preceding data.
- Appendable - data may be efficiently appended to a netCDF file without copying the dataset or redefining its structure.
- Self-Describing – a netCDF file may include metadata such as description of variables and dimensions, or units of measurements. Not all options should be part of the ISAC convention of use.
- Sharable – one writer and multiple readers may simultaneously access the same netCDF file. With Parallel netCDF, even multiple writers may efficiently and concurrently write into the same netCDF file.
- Extensible: Adding new dimensions, variables, or attributes to netCDF files does not require changes to existing programs that read the files.
- Archivable: Access to all earlier forms of netCDF data are promised to be supported by current and future versions of the software.

Available NetCDF Software

The netCDF library has been designed to read and write data that has been structured according to well-defined rules and has been ported across various computer platforms. The basic distribution contains the C, C++, F77, and F90 libraries. Further libraries can be downloaded for Ada, IDL, Java, MATLAB, Objective-C, Perl, Python, R, Ruby, and Tcl/Tk. More than 80 independent software applications have been developed for manipulating or displaying netCDF data²².

Benefits of Using the NetCDF for Binary Data

Reusing Existing Standards

Reusing existing standards is one of the proposed requirements for a new cytometry data file standard. NetCDF is a well matured open standard that has been adopted in

approximately 30 different fields to accommodate for array-oriented scientific data²³ over the past 20 years. The standard is also well described with high level quality documentation²⁴.

Support for Future Instruments and Other Analytical Cytology Data

NetCDF is flexible enough to accommodate different types of array-oriented scientific data. It has the potential to integrate list mode data with other types of analytical cytology data such as spectral data, raw and correlated waveform data and image data eventually.

Availability of Tools

The variety of available software tools²² shows the popularity of netCDF. Considering the available APIs and libraries, it is not necessary for developers of analytical software and hardware to handle the underlying netCDF binary file structure directly.

Performance

One of the goals of netCDF is to support efficient access to small subsets of large datasets. To support this goal, netCDF uses direct within-file access rather than sequential access. This can be much more efficient when the order in which data is read is different from the order in which it was written, or when it must be read in different orders for different applications.

An application can not only quickly process a subset of events but also a subset of parameters if not all parameters are required for a particular analytical step.

NetCDF Conventions

Specific conventions have been developed by different groups employing netCDF according to their own use cases²⁵. Following appropriate conventions ensure that data are not only decodable but also interpretable across variety of software applications. A software tool will be developed to test that files follow this convention. Examples of this are available from other groups that have adopted a similar approach²⁶.

The remainder of this document outlines proposed ISAC conventions for using netCDF to store binary list mode data. Extended sets) of conventions are expected to be developed for accommodation of other types of analytical cytology data such as spectral data, raw and correlated waveform data and image data eventually.

ISAC / List Mode 1.0 Conventions

1. Introduction

1.1. Goals

This standard is intended for use with list mode data such as that produced by flow cytometry instruments. The netCDF library has been designed to read and write data that has been structured according to well-defined rules and is easily ported across various computer platforms. The purpose of the ISAC / List Mode 1.0 conventions is to require conforming datasets to store data in a compliant and easily interpretable way.

1.2. Scope

This document represents a proposal for a component of the Analytical Cytometry Standard. It only addresses how binary list mode data could be stored within an ACS Container. There are many other components of the ACS Container (Figure 1) that are not addressed by this proposal and that yet need to be standardized.

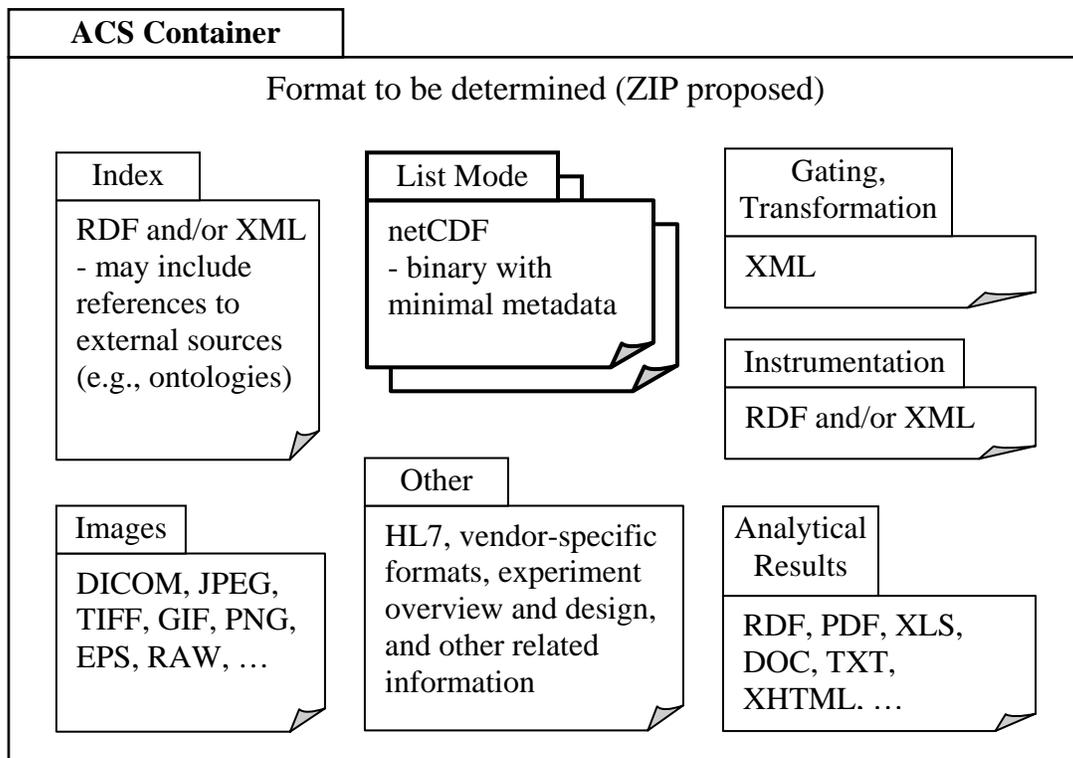


Figure 1 – Analytical Cytometry Standard. Overview of the different components of the proposed Analytical Cytometry Standard (ACS). The list mode component addressed by this proposal (netCDF) is highlighted in bold.

1.3. Terminology

The key words “shall”, “should”, and “may” in this document are to be interpreted as described in RFC 2119²⁷. Terms in this document that refer to components of a netCDF file are defined in the NetCDF Users’ Guide²⁸. This especially applies to the definition of dimensions, variables, and attributes that are essential for understanding of this document. Cytometry-specific terms are defined as described below:

1.3.1. Parameter

A parameter is understood as a type of measurement based on a signal or signals produced by a detector or several detectors of an analytical instrument. A parameter value is a digital representation of an instance of a parameter measurement. The area of the signal detected by the first photomultiplier tube (abbreviated as FL1-A) is an example of a parameter. Parameters may also encompass mathematically transformed measurements such as log fluorescence or calculated measurements such as electronic opacity (radio-frequency impedance divided by DC impedance) or other normalized measurements, such as fluorescence divided by DC impedance or low angle scatter. Note that a parameter description constructed as a concatenation of the detector type, a sequential number and how the signal was processed (e.g., “FL1-W”) is appropriate for engineering studies only and it is not suitable for publishing experimental results, labeling graph axes, etc.

1.3.2. Electronic Event

An event is a unit of data representing one particle (e.g., a cell). Events are typically stored in list-mode data files. An event is a vector (record) that has a value for each parameter.

1.3.3. List Mode Data File

A list mode data file is a file containing a sequential list of events.

1.4. Informative Overview

List mode data files conforming to these conventions are netCDF data files that shall be identified by the global attribute “Conventions” having the string value “ISAC/ListMode1.0”. This follows the NetCDF Users’ Guide²⁸ and Unidata⁸ offers a repository and maintain WWW links for sets of netCDF conventions²⁵. The global attribute “id” sets the identifier of the data file. List mode data are stored as one-dimensional netCDF data (dimension “Event”) with unlimited number of variables for this dimension (netCDF variables correspond to parameters as they are understood in cytometry). Each variable specifies explicitly its range. Values shall be on linear scales. Units shall be explicitly specified for time-related variables and these shall be in seconds and it should be since the beginning of data acquisition. Nothing but the described components shall be present in the file.

2. NetCDF Files and Components

2.1. Filename

NetCDF files shall have the file name extension “.nc”.

Rationale

This is a standard netCDF extension that makes it easier recognizable for the netCDF library. Different extensions may have different semantic. For example, if a “.zip”, a “.gz.”, or a “.bz2” file is opened by the netCDF library, this expects the file being a compressed netCDF file and it uncompress it transparently.

2.2. Data Types

Supported atomic data types within the binary list mode netCDF file are all netCDF-4 compatible numeric data types, which are the following data types: integer – 8-bit, 16-bit, 32-bit, and 64-bit in both, signed and unsigned version, and 32-bit and 64-bit floating point (signed).

Rationale

We have decided to allow for all netCDF-4 numeric data types in order to support for variety of use cases when storing analytical cytology data. We expect the 32-bit floating point data type being the most useful type for analytical cytometry instruments. However, there are still instruments producing data with a lower precision, which may benefit from some of the smaller integer-based data types. On the other hand, the 64-bit haven't been restricted in respect to the proposed “Requirements for a data file standard format to describe cytometry and related analytical cytology data” document, especially item 2.9 - support for future instruments.

Please note that some of these data types are not compatible with the netCDF v3.6 classic format. An alternative for discussion is to restrict some of these data types or to restrict to netCDF v3.6 data types only. This was our original proposal, which we have changed based on received comments (NetCDF v.3.6 does not include the 16-bit unsigned integer, which is useful for some cytometry data). The upside of netCDF-4 is a better support for variety of use cases. The downside of netCDF-4 is that “old” libraries and software cannot be used. This may not be an issue as there is no legacy netCDF software in cytometry that we needed to consider.

2.3. Dimensions

One and only one dimension shall be specified in the data file; the dimension shall be named “Event” and the length of this dimension shall correspond to the number of electronic events stored in the list mode data file. This dimension is used for all variables in the file. For our purposes, the “Event” dimension serves as index/record number in the binary file.

Rationale

Having a single dimension for all events and storing parameters as variables for each event seemed as the best solution considering both performance and semantic of how data are stored in the file¹⁵.

An option that we have considered was having two dimensions: “event” and “parameter”, with a single variable “value”. However, this is not how netCDF is intended to be used²⁸. Parameters encompass different types of measurements and thus the “value” variable would mix semantically heterogeneous information, which would prevent us from using standard netCDF features like variable name, valid range or unit.

A third option: a multidimensional space created as Event x Parameter1 x Parameter2 x ... x Parameter_n with true/false variables leads to a highly inefficient implementation (even with false values being omitted from the data file).

2.4. Variables

NetCDF variables shall correspond to parameters in list mode data files. Each variable shall only have the “Event” dimension.

Rationale

This fits into the semantic how variables in netCDF are typically used. A variable typically has a name, description, data type, valid range, units, etc., which captures the concept of a parameter in a list mode data file.

2.5. Packed and Compressed Data

An ISAC / List Mode 1.0 compliant file shall not use any methods of packing or compression within the netCDF file.

Rationale

There are two methods for reducing size: packing and compression. NetCDF employs the term packing to mean altering the data in a way that reduces its precision. By compression we mean techniques that store data more efficiently and result in no precision loss.

In netCDF, packing is implemented as automatic scaling and eventually moving the values of a variable (add_offset and scale_factor attributes) so that they fit into a smaller range and can be stored using a smaller data type (e.g., float instead of double). Packing introduces confusions about scales and ranges; it can eventually invoke automated and potentially inconsistent changes in read values (i.e., the actual read values may be dependent on what class/view is used to open a particular file). Moreover, packing is not very useful for the described list mode data files as fluorescence intensities and other parameter values are often stored using relative values, which can be moved or rescaled without packing.

Some methods have been described to allow for compression applied on a variable by variable basis in netCDF; however, these are not really standardized. The compression of the whole ACS container may eventually be a solution if data size is an issue. Since list

mode data is often in the form of sparse arrays, the development of lossless compression algorithms is challenging.

2.6. Global Attributes

Standard global attributes are global attributes that are further described in this document. An overview is also provided in Appendix A – Attributes. A file shall not contain any non-standard proprietary attributes.

Rationale

NetCDF allows for custom attributes and values, which is similar to custom keyword/value pairs as in FCS files. We would like to avoid proprietary information as well as any metadata in the binary file; alternative files within the ACS container may be used to store these in any format third parties desire, though XML and RDF are suggested formats to promote interoperability. Vendors are encouraged to approach ISAC with suggestions for additional required file formats.

2.6.1. Identification of Conventions

NetCDF files that follow these conventions shall indicate this by setting the global attribute “Conventions” to the string value “ISAC/ListMode1.0”.

Rationale

This follows the general netCDF guidelines on how a standard file following some conventions should be identified. netCDF also suggests the string to follow a directory-like syntax so that there may be more conventions created under the “ISAC” folder in the future²⁵.

2.6.2. Identification of Files

Each file shall include a global attribute “id” and its value shall be set to a string value identifying the data file. This identifier should be globally unique and it should be in the form of an URI²⁹.

Rationale

Global identification of each data file is essential for any referencing purposes; the URI format allows for integration with XML (and RDF) based documents.

Creating a globally unique ID should be technically feasible even in offline environments. It could for example use the cytometer serial number to generate the ID (i.e., manufactures would have their “namespaces” similar to manufactures of network card assigning globally unique MAC addresses). ISAC should develop guidelines to uniquely name files that will be suitable for deployment within both public and private networks.

2.7. Variable Attributes

NetCDF variables shall correspond to parameters in list mode data files. Standard variable attributes are described further in this document. An overview is also provided in

Appendix A – Attributes. Variables shall not contain any non-standard proprietary attributes.

Rationale

Similar to 2.6; we would like to avoid proprietary information as well as any metadata in the binary file.

2.7.1. Variable Names

In general, each variable of a netCDF file is identified by a name. This should be a short descriptive name that may be constructed as a concatenation of the detector type, a sequential number and an indication of how the signal was processed (e.g., “FL1-H”).

Time-related variables shall be identified by a name starting with the string “Time”, e.g., “Time”, “Time1”, “Time_A”, “Time-from-start”.

In addition, variables may also be described by a variable attribute “long_name” containing a string value that provides additional description of a variable (e.g., “Forward scatter - Height”, “CD45”, “Relative time stored in seconds from the beginning of data acquisition”).

Rationale

Having both a short and a long name for parameters has been shown useful in FCS. Having a single name/identifier for each variable would be technically sufficient and consistent with our efforts to move all metadata out of the binary file. However, allowing also for the “long_name” attribute has the advantage that general netCDF software recognizes it as a variable description. It is also useful for storing names intended for presentation purposes such as labeling axes on graphs.

Time-related parameters/variables have a different semantic than light-related parameters/variables, which is why these should be easily distinguishable. The rationale for allowing for more than a single “time” is motivated by future instruments that may eventually want to store more of these, such as time of the beginning of the signal generation, time of the peak of the signal generation, and time of the end of the signal generation.

2.7.2. Variable Range

The range for each variable shall be explicitly described by two numeric attributes: “valid_min” representing the smallest valid value of a variable, and “valid_max” representing the largest valid number of a variable. The data type of these attributes shall correspond to the data type of the related variable.

A floating-point based data type shall be used for “unbounded” variables, i.e., variables with at least one of the “valid_min” or “valid_max” set to plus or minus infinity. For example, a typical range for a time-related variable would be [0;+infinity], i.e., the value of the “valid_min” attribute would be specified as 0 and the value of the “valid_max” attribute would be specified as +infinity. Plus and minus infinity shall be encoded according to XDR/IEEE754^{21, 30}).

For floating point-based data types, missing values should be stored as NaN (“Not a Number” encoded according to XDR/IEEE754^{21,30}); however, also all values outside the range of a variable (i.e., smaller than “valid_min”, or greater than “valid_max”) are considered as missing values. For integer-based data types, missing values should be shall be stored as values outside of the specified valid range.

Rationale

The “valid_min” and “valid_max” are standard netCDF attributes that are recognized by general netCDF software to find out about the valid range for a variable. According to the netCDF guidelines, “Most generic applications that process netCDF datasets assume standard attribute conventions and it is strongly recommended that these be followed unless there are good reasons for not doing so.” Handling of missing values as well as NaN recommendations is also in accordance with the netCDF Users’ Guide²⁸.

Alternative

An alternative is to name these attributes as “minInclusive” and “maxInclusive” as used in XML schema¹³. This is against the netCDF guidelines and generic netCDF software would not understand these attributes. However, there may be a benefit of reusing XML schema terminology.

2.7.3. Variable Units

The units shall be explicitly specified for time-related variables. The attribute “units” shall be set to a string value “seconds since <timestamp>”, where <timestamp> should be replaced by a timestamp of the beginning of data acquisition that shall be formatted according to the recommendations in the Uduunits package³¹. The following excerpt from the Uduunits documentation explains the time unit encoding by example.

The specification:

“seconds since 1992-10-8 15:15:42.5 +00:00”

indicates seconds since October 8th, 1992 at 3 hours, 15 minutes and 42.5 seconds in the afternoon in the GMT time zone. The time zone specification can vary from -12:00 to +12:00, it can also be written without a colon using one or two digits (indicating hours) or three or four digits (indicating hours and minutes).

Specification of units is optional for other than time-related variable; however, the units shall be linear (i.e., values of these variables shall be on a linear scales).

Rationale

Explicitly described units support for better understandability. However, units may be difficult to describe for light-related or calculated parameters/variables, which is why they have been made optional for these.

The Uduunits package is typically used with netCDF software, which was the main reason for requiring a compliant syntax.

Specification of time relative to a timestamp different from the beginning of data acquisition is allowed (e.g., time difference from the preceding event).

Alternative

An alternative is the XML schema dateTime syntax (e.g., 1992-10-08T15:15:42.5-06:00). This is against the netCDF guidelines and the Uduunits package would not understand these values. However, XML-enabled applications could process this format easier.

Alternative

Another alternative is not to use units within the netCDF binary file. Again, this is not completely according to the netCDF guidelines and would prevent generic netCDF applications to process the file. However, it would enable us to move units to an external file (e.g., XML) that could eventually provide more formal mechanisms to express these.

3. Miscellaneous Compliance Requirements

3.1. Data Sets vs. Data Files

A data file shall store data corresponding to a single data acquisition process (i.e., single run of an analytical instrument) only. A single data set corresponds to a single data file.

Rationale

Multiple data sets within a single data file can now be achieved on a file level within the proposed ACS containers.

Moreover, the netCDF terminology for data files and data sets is slightly different than FCS users may expect. A netCDF dataset is a generalization of a netCDF file. It may be a netCDF file, an HDF5³² file, a collection of files, or anything else, which can be accessed through the netCDF API. A netCDF dataset serves as a semantic view of one or more files. NetCDF API allows to open a netCDF file as a netCDF dataset, which may automatically preprocess the data file (i.e., unpack the data, change data types, etc., which we do not use or need). We specify that a single data file shall correspond to a single data set to limit any possible confusion.

3.2. Underlying Data Format

NetCDF supports for some variants in the underlying data format: netCDFv3.6/classic, netCDFv3.6/64-bit offset and netCDF-v4/HDF5 format. Any of these variants are compliant with these conventions; however, consider the following when creating a list mode data file.

The format of a netCDF file is determined when creating the file. When opening an existing netCDF file the netCDF library will transparently detect its format and adjust accordingly. However, netCDF library versions earlier than 3.6.0 cannot read 64-bit offset format files, and library versions before 4.0 can not read netCDF-4/HDF5 files. NetCDF classic format files (even if created by version 3.6.0 or later) remain compatible with older versions of the netCDF library.

In order to maximize portability an ISAC / List Mode 1.0 compliant netCDF binary data file should use the netCDFv3.6/classic format for files significantly smaller than 2GB and the netCDFv3.6/64-bit offset format for files that are expected to exceed to 2GB in size. Please note that the large file support is dependent on the operating system and used

library facilities to support files larger than 2GB. On many 32-bit platforms the default size of a file offset is still a 32-bit signed integer, which limits the maximum size of a file to 2GB. Using large file support interfaces and the 64-bit file offset type, the maximum size of a file may be as large as 2^{63} bytes (or 8 EB). For many current platforms, large file macros or appropriate compiler flags have to be set to build a library with support for large files. This is handled automatically in netCDF v3.6 and newer.

The netCDF-4/HDF5 format should only be used when a feature of this format is required, e.g., the need of a netCDF-4 specific data type (e.g., unsigned 16-bit integer), when parallel I/O access is needed or when files are required to be readable by HDF5³² (version 1.8.0 or newer) compliant software.

Rationale

We are following the general NetCDF Users' Guide³³, which encourages using the netCDFv3.6/classic format unless some features of the new formats are needed.

Future Direction

In the future, we are considering restricting the variants of the underlying data format to the netCDF-v4/HDF5 format only. In the flow cytometry domain, we may not need to consider any legacy software, and having a single underlying format is an obvious advantage. We have not restricted to this format for this version of the Conventions proposal as it depends on HDF5 version 1.8.0, which is so far only available in beta release. The final version of netCDF-4 is promised along with the release version of HDF5 1.8.0.

Appendix A – Attributes

All standard attributes are listed in the following table. The “Type” values are “Str” for a string and “Num” for numeric (i.e., depending on the data type of the related variable). The “Use” values are “Glo” for global, “Var” for variables, “Req” for required, and “Opt” for Optional.

Attribute	Type	Use	Description
Conventions	Str	Glo/Req	Name of the conventions followed by the data file. Value shall be fixed to “ISAC/ListMode1.0”. See 2.6.1 for details.
id	Str	Glo/Req	Identifier of the data file. See 2.6.2 for details.
long_name	Str	Var/Opt	Long description of a variable. See 2.7.1 for details.
valid_min	Num	Var/Req	Smallest valid value of a variable. See 2.7.2 for details.
valid_max	Num	Var/Req	Largest valid value of a variable. See 2.7.2 for details.
units	Str	Var/Req/Opt	Units of a variable's content. The attribute is required for time-related variables and optional for other types of variables. See 2.7.3 for details.

Appendix B – Examples

Binary List Mode Example Files

Several example binary list mode data files following these conventions may be downloaded from:

https://sourceforge.net/project/showfiles.php?group_id=175725&package_id=202340&release_id=443566

Appendix C – Software Provided with these Conventions

The following software will be developed and provided with these conventions. The software will be provided as open-source software (i.e., providing source codes) with non-restrictive licenses (i.e., with no constraints on how the software can be used). ISAC will place these utilities on its website, where they will be freely available for download.

ISAC / List Mode 1.0 Conventions Checker

The ISAC / List Mode 1.0 Conventions Checker will be an open source software application that can verify compliance of a netCDF file with the ISAC / List Mode 1.0 conventions. All the breaches of these conventions will be reported as errors and all deviations from recommendations as warnings.

FCS to NetCDF Converter

The FCS to netCDF converter will be an open source software application that can convert an FCS2³⁴ or FCS3¹ data file to a netCDF file (or netCDF files) conforming to the ISAC / List Mode 1.0 conventions.

NetCDF to Sequential Binary File and XML File Converter

This converter will be an open source software application that can convert a netCDF file conforming to the ISAC / List Mode 1.0 conventions to a simple binary sequential file with the structure as currently used for the data segment in FCS¹, i.e., a file with measurements stored as a list of events, one event sequentially after another; each event stored as a list of measurement values; the number and order of measurement values corresponding to the number and order of variables within the netCDF data file.

This software will also extract metadata stored in netCDF files following these conventions (e.g., parameter names, ranges, units) to an XML (or eventually RDF) file conforming to an ISAC XML (or RDF) schema.

This software will also be able to combine the simple binary sequential file with the metadata in XML into a netCDF file conforming to the ISAC / List Mode 1.0 conventions.

Appendix D – NetCDF vs. Data File Requirements

This appendix discusses how the use of netCDF and the stated conventions addresses the proposed “Requirements for a data file standard format to describe cytometry and related analytical cytology data” (version 0.070920).

1. Encode the required information to interpret a cytometry experiment

NetCDF is intended to use one piece of the required information to interpret a cytometry experiment - the recordings of the fluorescent or light-scattering properties of hundred of thousands of individual particles. Other formats are being proposed to include further information essential to interpret a cytometry experiment.

2. Facilitate the reproduction of a cytometry experiment

Other metadata-oriented formats are being proposed to include information that facilitates the reproduction of a cytometry experiment.

3. Efficiently store cytometry list mode data

The size overhead of the underlying data file structure is not significant. Table 1 demonstrates the overhead based on used data types and number of events in the file.

Table 1 – netCDF file size overhead measured in percentage to the raw data size (i.e., the length of the data type multiplied by the number of parameters and by the number of events). Tests are based on 6 parameter files that can be downloaded specified in Appendix B – Examples.

Events	8 bit integer	16 bit integer	32 bit integer	32 bit floating point	64 bit floating point
100	217.3333 %	83.6666 %	45 %	42 %	21.9166 %
1000	66.7333 %	8.4 %	4.1833 %	4.2 %	2.2 %
10000	51.6733 %	0.8366 %	0.4183 %	0.4183 %	2.1916 %
100000	50.1673 %	0.084 %	0.04183 %	0.04183 %	0.022 %

4. Transparently store text-based data

NetCDF is not intended to store text-based data. We expect to use XML/RDF files such as proposed for description of gating or transformations. There is minimal amount of text-based data (such as parameter names, ranges, and units eventually) in netCDF files following these conventions, which could be considered as not stored transparently. We believe that this is a very minimal and essential set making the file self-describable, avoiding confusions and ambiguities. This data can be extracted into a transparent mode using the “NetCDF to Sequential Binary File and XML File Converter” (Appendix C).

5. Facilitate support for other analytical cytology data

The netCDF is a universal format for array-oriented scientific data and as such, it is possible to use it different types of analytical cytology data. These “Conventions” can be adjusted/extended for each of the types of cytology data. NetCDF has the potential to integrate list mode data with other types of analytical cytology data such as spectral data, raw and correlated waveform data and image data eventually.

6. Each type of information shall be uniquely identifiable

We suggest the usage of the global id attribute to uniquely identify the data.

7. It shall be possible to resolve data files over the Internet

In general, we suggest using the id attribute in form of an URI that makes it possible to eventually resolve data over the internet. However, we do not require it to be in the form of a URI, neither do we require making all data publicly available.

8. Each type of information shall only be stored in one file format

NetCDF is only intended to store the recordings of the fluorescent or light-scattering properties and other properties of individual particles. Further data formats are proposed to store different types of information. Minimal overlap can occur in cases like the name of a parameter, which rather than duplicating information allows to link the binary data to external metadata.

9. Support for future instruments

NetCDF is a flexible format that allows storing any array-oriented scientific data. As such, it is expected that it will be suitable for future instruments as well. NetCDF supports for a variety of data types to allow for various use cases including increased precision of future instruments or different types of analytical cytology data such as spectral data, raw and correlated waveform data and image data eventually.

10. Provide semantic meaning to all required/standardized information

These conventions describe unambiguously the semantic of all information standardized within netCDF to describe list mode data.

11. Extensible by third parties

It is possible to store different types of measurements (i.e., the number and types of parameters/variables are unlimited). However, proprietary metadata shall not be included in netCDF files as there are/will be other formats proposed to use for metadata storage.

12. Extensions shall be separated and removable from normative parts

NetCDF files following the ISAC / List Mode 1.0 conventions do not include extensions.

13. Ensure that metadata can always be located

Virtually no metadata are included in the netCDF file; however, there is a unique identifier that enables linking data and metadata. Also, the proposed ACS container is expected to serve the purpose of coupling data and metadata together.

14. Support for use cases where new metadata are being added subsequently

As metadata are not a part of the netCDF file, these may be added subsequently and independently.

15. Several instances of mutable information may share immutable information

Nothing in netCDF or these conventions precludes this requirement.

16. Some information may be encoded by a referenced semantic model or an existing standard

Nothing in netCDF or these conventions precludes this requirement.

17. It shall be possible to refer to multiple semantic models or standards

Nothing in netCDF or these conventions precludes this requirement.

18. Use a manufacturer-independent format

The format is manufacturer-independent in that is being developed and maintained by Unidata, which represents a diverse community of over 160 institutions vested in the common goal of sharing data, and tools to access and visualize that data. For 20 years Unidata has been providing data, tools, and support to enhance Earth-system education and research.

19. Use a programming language-independent format

The data format itself is programming language independent. Support, libraries, and interfaces are provided for many programming languages including C, C++, Fortran, F77, F90, Ada, Java, IDL, MATLAB, Objective-C, Perl, Python, R, Ruby, and Tcl/Tk. As it is possible to call a C function/library from virtually all other languages, it does not mean that developers are limited to the use of mentioned languages. While it is straightforward for some languages (C++, C#, ...) it may be more difficult (or limiting) for others. In these cases, specific bindings are being developed (e.g., Visual Basic) or a completely independent implementation is released (e.g., for Java).

20. Use a file/operating system-independent format

NetCDF is file and operating system independent. There are no known compatibility issues on existing platforms. Data storage is XDR-based²¹ (fixed Endianes and all other platform-specific issues) and data may be opened from local file systems as well as from resolvable URIs²⁹.

21. Use a simple format oriented on data interoperability

While the format itself may not be simple, the netCDF is an API designed to make reading and writing of scientific data as simple as possible. Using netCDF with available software libraries is simple (see examples³⁵). The format strongly focuses on interoperability in that it is platform independent (XDR-based) and support is provided for many programming languages including C, C++, F77, F90, Ada, Java, IDL, MATLAB, Objective-C, Perl, Python, R, Ruby, and Tcl/Tk. With respect to the available software support and netCDF usage in many scientific areas, it is unlikely that any group will have to write a netCDF parser/writer (even if a particular programming language is not supported directly, it is likely that it will be possible to call C/C++ or other external routines to handle the files). The available libraries represent well tested software that is in widespread use and that is maintained by developers of the netCDF standard. Furthermore, we propose to implement an open source FCS to NetCDF Converter as well as a NetCDF to Sequential Binary File and XML *File Converter*. Restricting to netCDF-4 (HDF5) may eventually even facilitate compliance with this requirement in that it is avoiding options in the underlying data format.

22. Describe the standard unambiguously

These conventions along with the netCDF specification³³ and other netCDF documentation²⁰ represent an unambiguous description of how list mode data can be stored.

23. There shall be an open-review period before adopting the standard

Nothing in netCDF or these conventions precludes this requirement.

24. There shall be a mechanism to formally validate the standard

The netCDF has been validated by a 20 years history of successful usage in various scientific domains to store different types of array-oriented scientific data. These conventions have been developed following proven methodologies as used by other groups to develop conventions for their data²⁵. Moreover, we suggest developing open source file format converters, which would also show the suitability of netCDF to store list mode data. There may be additional tests needed for medical devices.

25. The standard shall be public

Nothing within netCDF precludes the standard being public. NetCDF represents an open standard and the netCDF library is distributed without licensing or other significant restrictions.

26. An implementation of the standard shall be possible without charge

netCDF including netCDF libraries are distributed for free without licensing or other significant restrictions.

27. The implementation of the standard shall be non-restrictive

Usage of the netCDF format as well as libraries is non-restrictive.

28. Reuse existing standards

Using netCDF represents reusing of an existing standard that has been successfully used for many array-oriented scientific data. Also, while it focuses only on the binary list mode data component of an ACS container, it does not preclude being used with other existing standards within the ACS container or reusing data types from other standards within these other meta data files.

29. Conformance to the standard shall be (easily) testable

We propose to develop an open source conventions checker that can verify compliance of a netCDF file with the ISAC / List Mode 1.0 conventions. All the breaches of these conventions will be reported as errors and all deviations from recommendations as warnings.

30. The file format shall be as self-explanatory as possible

The format is self-explanatory when processed using netCDF libraries or when displayed via generic netCDF tools (see Appendix B – Examples).

References

1. Seamer LC, Bagwell CB, Barden L, et al. Proposed new data file standard for flow cytometry, version FCS 3.0. *Cytometry*. 1997;28:118-122.
2. Spidlen J, Brinkman RR, Bioinformatics Standards for Flow Cytometry Consortium. Gating-ML: Draft Standard for Gating Description in Flow Cytometry. Available at: <https://sourceforge.net/projects/flowcyt>. Accessed 07/31, 2007.
3. Spidlen J, Brinkman RR, Bioinformatics Standards for Flow Cytometry Consortium. Transformation-ML: Draft Standard for Transformation Description in Flow Cytometry. Available at: <https://sourceforge.net/projects/flowcyt>. Accessed 07 2007.
4. Spidlen J, Brinkman RR, Bioinformatics Standards for Flow Cytometry Consortium. A Proposal for the Analytical Cytometry Standard. Available at: <http://flowcyt.sourceforge.net/acs/>. Accessed 07/31, 2007.
5. World Wide Web Consortium. Available at: <http://www.w3.org/>.
6. The Object Management Group (OMG). Available at: <http://www.omg.org/>.
7. Dublin Core Meta Data Initiative. Available at: <http://dublincore.org/>.
8. Unidata. Available at: <http://www.unidata.ucar.edu/>.
9. W3C Math Working Group. Mathematical Markup Language (MathML) Version 2.0 (Second Edition), W3C Recommendation. Available at: <http://www.w3.org/TR/MathML2/>. Accessed 07/31, 2007.
10. W3C - RDF Core Working Group. Resource Description Framework (RDF). Available at: <http://www.w3.org/RDF/>.

11. W3C. XHTML 1.0 - The Extensible HyperText Markup Language (Second Edition). Available at: <http://www.w3.org/TR/xhtml1/>.
12. W3C. Scalable Vector Graphics (SVG). Available at: <http://www.w3.org/Graphics/SVG/>.
13. W3C. XML Schema. Available at: <http://www.w3.org/XML/Schema>.
14. Leif RC, Leif SB, Leif SH. CytometryML, an XML format based on DICOM and FCS for analytical cytology data. *Cytometry*. 2003;54A:56-65.
15. Leif RC. CytometryML, binary data standards. *Manipulation and Analysis of Biomolecules, Cells, and Tissues II, SPIE Proceeding*. 2005;5699:325-333.
16. World Wide Web Consortium (W3C). Extensible Markup Language (XML). Available at: <http://www.w3.org/TR/REC-xml/>.
17. Lee J, Spidlen J, Boyce K, et al. MIFlowCyt: Minimum Information for a Flow Cytometry Experiment. Available at: <http://flowcyt.sourceforge.net/miflowcyt/>. Accessed 07/31, 2007.
18. Shadbolt N, Berners-Lee T, Hall W. The semantic web revisited. *IEEE Intelligent Systems*. 2006;21:96-5.
19. Wang X, Gorlitsky R, Almeida JS. From XML to RDF: How semantic web technologies will change the design of 'omic' standards. *Nat Biotechnol*. 2005;23:1099-103.
20. NetCDF. Available at: <http://www.unidata.ucar.edu/software/netcdf/>.
21. RFC 4506 - XDR: External Data Representation. Available at: <http://tools.ietf.org/html/rfc4506>.
22. Software for Manipulating or Displaying NetCDF Data. Available at: <http://www.unidata.ucar.edu/software/netcdf/software.html>.
23. Where is NetCDF Used? Available at: <http://www.unidata.ucar.edu/software/netcdf/usage.html>.
24. NetCDF Documentation. Available at: <http://www.unidata.ucar.edu/software/netcdf/docs/>.
25. NetCDF Conventions. Available at: <http://www.unidata.ucar.edu/software/netcdf/conventions.html>.
26. CF-Convention compliance checker for NetCDF format. Available at: <http://titania.badc.rl.ac.uk/cgi-bin/cf-checker.pl>.
27. Bradner S., The Internet Engineering Task Force. Request for Comments: 2119 - Key words for use in RFCs to Indicate Requirement Levels. Available at: <http://www.ietf.org/rfc/rfc2119.txt>. Accessed 07/31, 2007.
28. The NetCDF Users' Guide. Available at: <http://www.unidata.ucar.edu/software/netcdf/docs/netcdf/index.html>.
29. RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax. Available at: <http://tools.ietf.org/html/rfc3986>.

30. IEEE. IEEE 754: Standard for Binary Floating-Point Arithmetic. Available at:
<http://grouper.ieee.org/groups/754/>.
31. The Unidata UDUNITS Package. Available at:
<http://www.unidata.ucar.edu/software/udunits/>.
32. HDF5. Available at: <http://www.hdf-group.org/products/hdf5/index.html>.
33. The NetCDF Users' Guide: How to Select the Format. Available at:
<http://www.unidata.ucar.edu/software/netcdf/netcdf-4/newdocs/netcdf.html#Which-Format>.
34. Data file standard for flow cytometry. data file standards committee of the society for analytical cytology. Cytometry. 1990;11:323-332.
35. Example netCDF programs. Available at:
<http://www.unidata.ucar.edu/software/netcdf/examples/programs/>.