ISAC
International Society for Advancement of Cytometry

Archival Cytometry Standard ACS

International Society for Advancement of Cytometry Candidate Recommendation DRAFT

Document Status

The Archival Cytometry Standard (ACS) has undergone several revisions since its initial development in June 2007. The current proposal is an ISAC Candidate Recommendation Draft. It is assumed, however not guaranteed, that significant features and design aspects will remain unchanged for the final version of the Recommendation.

This specification has been formally tested to comply with the W3C XML schema version 1.0 specification but no position is taken with respect to whether a particular software implementing this specification performs according to medical or other valid regulations.

Disclaimer of Liability

ACS 1.0, DRAFT of ISAC Candidate Recommendation, version 150428, April 28, 2015

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 vendors. It includes data and metadata describing the subject being analyzed as well as the acquisition conditions. There are several use cases requiring these raw FCS files to remain intact after the data have been collected. However, since data analysis is typically performed post-acquisition, it is commonly stored in additional files, such as proprietary project files and workspaces, or standard Gating-ML files. This practice introduces the need of being able to bundle data with related additional information.

This document represents the Archival Cytometry Standard (ACS), which has been developed to bundle data with different components describing cytometry experiments. It captures relations among data and other components and includes support for audit trails, versioning and digital signatures. The ACS container is based on the ZIP file format with an XML-based Table of Contents specifying relations among files in the container. The XML Signature W3C Recommendation has been adopted to allow for digital signatures of data and other components within the ACS container.

Keywords: flow cytometry, cytology, data standard, file format, FCS, ZIP, XML

TABLE OF CONTENTS

1. Overview 1
1.1 Introduction 1
1.2 Scope 1
1.3 Purpose 1
1.4 Normative References 2
1.5 The Content of this Specification 2
1.6 Acronyms and Abbreviations 2
1.7 Keywords Indicating Requirement Levels 3
1.8 Namespaces and their Prefixes within this Document 3
2. Design Principles and Rationale 3
2.1 Using ZIP as the Container File Format 3
2.2 Using XML for Container’s Table of Contents 4
2.3 Using W3C XML Signatures for Digital Signatures of ACS Components 5
3. Conformance 5
3.1 File Conformance 5
3.2 Software and Hardware Conformance 6
4. General 6
4.1 ACS File Name 6
4.2 ACS File Format 6
4.3 Naming Files and Directories within the ACS Container 6
4.4 ACS Contents and Audit Trail 7
4.4.1 ACS Table of Contents 7
4.4.2 Other Files in ACS 7
4.4.3 Audit Trail Implementation 7
5. ACS Table of Contents Format 8
5.1 General 8
5.2 Table of Contents File Structure 8
5.3 Parent Table of Contents 9
5.4 Listing Files in ACS Table of Contents 10
5.4.1 File Resource Identifiers 10
5.4.2 File Types 10
5.4.3 Free Text File Description 11
5.5 Associations Between Files 11
5.6 Additional Information in the Table of Contents 12
5.7 Digital Signatures 13
5.7.1 Detached Signatures 13
5.7.2 Embedded Signatures 13
Annex A - Bibliography [Informative] 15

ISAC
International Society for Advancement of Cytometry

Archival Cytometry Standard ACS

1. Overview

1.1 Introduction

In cytometry, such as flow or image cytometry, data is usually first acquired by an instrument and then analyzed using software tools provided either by the instrument manufacture or by third parties. Acquired data files typically contain metadata describing the subject being analyzed and the acquisition conditions. It is desirable to leave the original data intact and therefore the post-acquisition analysis is usually stored in separate files, such as proprietary project files, workspaces, Gating-ML files, etc.

The Archival Cytometry Standard (ACS) has been developed to bundle these various components describing cytometry experiments. It captures relations among data and metadata and includes support for audit trails, versioning and digital signatures.

1.2 Scope

This document provides detailed specifications on how to bundle cytometry data with metadata and analysis components in a single container file format using the ZIP [1] specification. It also specifies how to create an XML-based [2] Table of Contents within the container to describe data and metadata within the container as well as relations among all affected files. Finally, this specification covers the implementation of a change log (audit trail, revision history) within ACS containers and incorporates the XML Signature W3C Recommendation to allow for digital signatures of data and other components within the ACS container.

Except for the ACS Table of Contents file (section 5), this specification does not specify, neither restrict, the format of components placed within the ACS container. Several other ISAC specifications, such as the flow cytometry data file standard (FCS [3]) or Gating-ML [4] may be used as formats for the data and metadata components within ACS. Additional third party standards as well as proprietary file formats may also be used.

1.3 Purpose

When storing cytometry data and analysis components in separate files, it is essential to provide a mechanism to bundle these together and to capture relations among these files. In addition, certain use cases require digital signatures of both data and metadata files as well an audit trail in case files have been altered. The purpose of this document is to provide a standard means of addressing these requirements. Specifically, the purpose of ACS is to bundle data with metadata components while capturing relations among files within ACS and allowing for both an audit trail and digital signatures. Ultimately, ACS aims to facilitate cross-platform sharing of data, metadata and analysis descriptions between different software packages, and provides the means to integrate methods with results in cytometry reporting.

1.4 Normative References

The following referenced documents are indispensable for the application of this standard. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

— Application Note on the .ZIP file format, version 6.2.0. [1].

— W3C Recommendation, Extensible Markup Language (XML) 1.0 (Fourth Edition) [2].

— W3C Recommendation, XML Schema Part 0: Primer Second Edition [5].

— W3C Recommendation, XML Schema Part 1: Structures Second Edition [6].

— W3C Recommendation, XML Schema Part 2: Datatypes Second Edition [7].

— W3C Recommendation, XML Signature Syntax and Processing Second Edition [8].

The following documents are useful for the application of this standard. They represent other standards and standard proposals relevant for understanding of this specification and/or intended to describe or store cytometry data and metadata together with information about cytometry experiments and analyses.

— Spidlen J, Moore W, Parks D, Goldberg M, Bray C, Bierre P, Gorombey P, Hyun B, Hubbard M, Lange S, Lefebvre R, Leif R, Novo D, Ostruszka L, Treister A, Wood J, Murphy RF, Roederer M, Sudar D, Zigon R, Brinkman RR. Data File Standard for Flow Cytometry, version FCS 3.1 [3].

— Seamer LC, Bagwell CB, Barden L, Redelman D, Salzman GC, Wood JC, and Murphy RF. Proposed new data file standard for flow cytometry, version FCS 3.0 [9].

— Spidlen J, Leif RC, Moore W, Roederer M, Brinkman RR. Gating-ML: XML-based Gating Descriptions in Flow Cytometry [4].

1.5 The Content of this Specification

This specification consists of the following parts:

a) Normative: This document providing a detailed description of the ACS specification.

b) Normative: The XML schema TOC.v1.0.xsd defining syntax of the Table of Contents XML file in the ACS container.

c) Informative: Examples of ACS files.

All the components of this standard are available from World Wide Web at http://www.isac-net.org/. The specification may also be downloaded from http://flowcyt.sf.net/. *Note: this will be true upon completion and approval of the specification.*

1.6 Acronyms and Abbreviations

ACS	Archival Cytometry Standard
DSTF	Data Standards Task Force

ECMA	European Computer Manufacturers Association
FCS	Flow Cytometry Standard
HTML	Hypertext Markup Language
IANA	Internet Assigned Numbers Authority
IEEE	Institute of Electrical and Electronics Engineers
ISAC	International Society for Advancement of Cytometry
RFC	Request for Comments
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
W3C	World Wide Web Consortium
XML	Extensible Markup Language

1.7 Keywords Indicating Requirement Levels

The key words *shall*, *should*, and *may* in this document are to be interpreted as described in RFC 2119 [10] and are also compatible with the IEEE Standards Style Manual. The word *shall* is used to indicate mandatory requirements to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*). The word *should* is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited (*should* equals *is recommended that*). The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

1.8 Namespaces and their Prefixes within this Document

The following XML namespaces [11] are used within the XML schemas and XML examples in this specification:

Prefix	**Namespace**
none	`http://www.w3.org/2001/XMLSchema`
`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
`toc:`	`http://www.isac-net.org/std/ACS/1.0/toc/`
`sig:`	`http://www.w3.org/2000/09/xmldsig#`

2. Design Principles and Rationale

2.1 Using ZIP as the Container File Format

Reusing existing standards is one of the proposed requirements for a new cytometry data file standard [12]. The ZIP file format is a popular data compression and archival format. ZIP is a mature open standard that has been adopted in many different fields over the past 20 years. A ZIP file contains one or more files that have been compressed to reduce their file size, or stored as-is. The format was originally designed by Phil Katz for PKZIP. However, many software utilities other than PKZIP itself are now available to create, modify, or open (unzip, decompress) ZIP files, notably WinZip, BOMArchiveHelper, KGB Archiver, PicoZip, Info-ZIP, WinRAR, IZArc, 7-Zip, ALZip, TUGZip, Universal Extractor and Zip Genius. Microsoft has included built-in ZIP support (under the name "compressed folders") in later versions of its

Windows operating system. Apple has included built-in ZIP support in Mac OS X 10.3 and later via the BOMArchiveHelper utility.

ZIP files generally use the file extensions ".zip" or ".ZIP" and the MIME media type "application/zip". Many software applications use ZIP as a container (wrapper) for several files in a certain structure. Generally, when this is done a different file extension is used. Examples include Java JAR files, id Software .pk3/.pk4 files, package files for StepMania and Winamp/Windows Media Player skins, XPInstall, as well as OpenDocument and Office Open XML office formats, Google Earth KMZ files, Mozilla Firefox Add-ons, Nokia's mobile phone themes, or Magic Draw UML models.

Based on comparison [13], ZIP is supported on all common operating system platforms including Windows, MS-DOS, PC-DOS, Mac OS X, Linux, BSD, and Unix. The variety of available software tools shows the popularity of the ZIP format. ZIP is the most widely supported archive format. There is wide end user software supporting the reading and writing of ZIP files (e.g., 7-Zip, Alpha ZIP, ALZip, Basic Zip, Beezer, BitZipper, BOMArchiveHelper, Filzip, Info-ZIP, IZArc, jzip, KGB Archiver, PeaZip, PKZIP, PowerArchiver, Squeez, StuffIt, StuffIt Expander, The Extractor, The Unarchiver, TUGZip, WinAce, WinRAR, WinRK, WinZip, XAD, ZipGenius, Zipeg, ZipZag; based on comparison [13]). Software libraries supporting the ZIP file format are available for virtually all programming languages. It is also possible to create and extract zip-formatted archives with open source software, e.g., Info-ZIP [14].

The ZIP format represents an open standard with the exception of features described in Section XIV (the "Strong Encryption Specification") that are covered by a pending patent application. Portions of the Strong Encryption technology are available for use at no charge under certain terms and conditions. See License Agreement of the Application Note on .ZIP File Format Specification [1]. However, encryption of ACS containers is an institution-specific requirement and not within the scope of ISAC's DSTF. Encryption is not a required component of ACS.

The ZIP archive format compresses every file separately, which allows individual retrieval of files without reading through other data. Files can be stored either uncompressed or by one of the supported compression algorithms (See the Application Note on .ZIP File Format Specification [1] section IV. - General Format of a .ZIP file, subsection A - Local File Header / compression method). In practice, ZIP is almost always used with Katz's DEFLATE algorithm (see section IX. Deflating - Method 8, and section X. Enhanced Deflating - Method 9 of the Application Note on .ZIP File Format Specification [1]), except when files being added are already compressed or are resistant to compression.

Using the "ZIP64" format extensions available since version 4.5, there are virtually no limits to the uncompressed size of a file, compressed size of a file and total size of the archive. While the most recent version of the ZIP specification (i.e., Application note on the ZIP file format) is 6.3.2, the ACS file format is based on ZIP specification version 6.2.0. Version 6.2.0 is a more widely supported version of ZIP that is also used as by the ECMA International Technical Committee for the purpose of standard Office Open XML File Formats. We also performed our own tests with software under both Linux (Ubuntu) and Windows XP (WinZIP and Windows Compressed Folders) to review whether new features added after version 6.2.0, such as the UTF-8 encoded file names, are supported by existing software. Unfortunately, these features have not yet been widely incorporated in existing software, and therefore we believe that the updates and added features between version 6.2.0 and 6.3.2 do not justify the drawbacks of using a less supported version of the specification.

2.2 Using XML for Container's Table of Contents

XML technology was chosen for the implementation of the ACS Table of Contents since it is a well established and supported technology, free of any of legal encumbrances, and it provides a robust and durable format for information storage and transmission. It can be used to describe and identify information accurately and unambiguously, in a way that computers can easily interpret and manipulate it. XML allows for separation of the content from the visual appearance. Moreover, it allows documents to be created and handled consistently and without structural errors since it provides a standardized way of describing, controlling, and allowing or disallowing particular types of document structure. Generally, the usage of XML for the ACS Table of Contents facilitates the interchange and validation of ACS data between different software packages.

2.3 Using W3C XML Signatures for Digital Signatures of ACS Components

Being able to digitally sign data and metadata is especially important for clinical use cases with the need for unambiguously verifying the origin of electronic documents. Reusing an established external standard simplifies the development of ACS and brings many benefits to the user community as well as software and hardware vendors, in that they are able to apply existing knowledge and reuse existing tools and software libraries.

The usage of W3C XML Signatures [8] allows for seamless integration of digital signatures within the XML-based ACS Table of Contents. XML signatures can be used to sign data and metadata of any type as long as these are accessible via a URL. For the purpose of ACS, W3C XML signatures provide the option to sign any data within the ACS container as well as external data. In addition, the XML Signatures include provisions to also sign the document that the signature resides in (i.e, the ACS Table of Contents itself), which essentially allows for the digital signature of the whole ACS container.

3. Conformance

3.1 File Conformance

To be conformant with this standard, an ACS file shall meet the following requirements:

a) The ACS file name should include the `.acs` file name extension as stated in section 4.1.

b) The ACS file shall be a valid ZIP file conformant to PKWARE's Application Note on the .ZIP file format version 6.2.0 [1] as stated in section 4.2.

c) The names of the files within ACS shall meet requirements stated in section 4.3.

d) The ACS file (ZIP container) shall include one or more Table of Contents XML files placed in the root of the ZIP container. The name(s) of the Table of Contents XML file(s) shall conform to the `TOC`*`n`*`.xml` file naming scheme described in section 4.4.1.

e) The `TOC`*`n`*`.xml` file naming scheme is reserved for the Table of Contents XML files only; within ACS (i.e., the ZIP container), files other than the Table of Contents XML shall use different names so that they not conform to the `TOC`*`n`*`.xml` file naming scheme described in section 4.4.1; see also section 4.4.2.

f) The Table of Contents XML file(s) shall be valid XML files conformant to W3C Recommendation, Extensible Markup Language (XML) 1.0 (Fourth Edition) [2].

g) The Table of Contents XML file(s) shall be valid according to the provided TOC.v1.0.xsd XML schema.

h) The Table of Contents XML file(s) shall meet requirements stated in section 5.

i) The ZIP container should contain cytometry data and metadata files and it may contain additional files. These components may use arbitrary file formats (i.e., the file formats of these components are not specified, neither restricted, by this specification).

3.2 Software and Hardware Conformance

To be compliant with this standard, a software application or hardware instrument shall be able to “read”, “write”, or “read and write” the archival cytometry standard (ACS) files and it shall meet the following criteria:

- *Writing*: When ACS files are produced (written) then these shall be valid ACS files according to section 3.1.
- *Reading*: The software application (or the hardware instrument) shall be able to read any ACS file that is valid according to section 3.1 and understand the general structure of the file, process the Table of Contents XML file(s) in order to locate additional files in the container, understand the audit trail (revision history) in order to navigate among multiple versions and revisions (if these are present), and understand standard relations among files within ACS (as stated in section 5.5). The software application (or hardware instrument) should also be able to process at least one of the commonly used cytometry data file formats, such as the FCS data file [3] or any of the file formats used in image cytometry. Finally, it may or may not be able to understand or verify digital signatures (as described in section 5.7); however, even if digital signatures are not supported, the presence of digital signatures in ACS shall not negatively affect the ability of the software application (or hardware instrument) to read the ACS files.

4. General

4.1 ACS File Name

The ACS file name should be named with the `.acs` file name extension. Deviation from this recommendation is allowed in cases where a file name is not applicable (e.g., ACS being transformed as a byte stream, ACS being captured in a binary large object in a database, etc.) or where the specified file extension is impossible to create or if it would create undesirable effects on a specific platform. In all other cases, the usage of the standardized extension is strongly advised.

4.2 ACS File Format

The ACS file shall be a valid ZIP file conformant to PKWARE’s Application Note on the .ZIP file format version 6.2.0 [1]. If data are compressed at all, the “deflate” algorithm (IX. Deflating, compression method 8 in Application Note on the .ZIP file format version 6.2.0 [1]) should be preferred among other methods in order to maximize interoperability.

4.3 Naming Files and Directories within the ACS Container

In the ACS container, there shall be no two files or directories having full names that differ only in the case of (some or all) letters. For example, only one of “file1.fcs”, “File1.fcs”, “file1.FCS”, or “FILE1.FCS” may be present on the same path within an ACS container. While case sensitive file systems as well as ZIP would allow for these to be present simultaneously, such a ZIP file would not be properly extractable to case insensitive file systems. Therefore, full (i.e., including path) file (and directory) names in ACS shall be unique for each file (and directory) even if considered as case insensitive.

4.4 ACS Contents and Audit Trail

4.4.1 ACS Table of Contents

The ACS file (ZIP container) shall include one or more Table of Contents XML files placed in the root of the ZIP container. These Table of Contents XML files shall be named `TOC`*`n`*`.xml`, where *`n`* is an integer going from 1 to the number of various Table of Contents XML files present in the container.

In the simplest case, a single Table of Contents XML file named `TOC1.xml` is present. Files with n > 1 (e.g., `TOC2.xml`, `TOC3.xml`, etc.) are used to implement the revision history (audit trail) in ACS. If multiple Table of Contents XML file are present, these represent the evolution of the ACS contents and they shall be consecutively named `TOC1.xml`, `TOC2.xml`, `TOC3.xml`, etc. Each of these Table of Contents XML files represents a full (standalone) description of the contents of the ACS and its format shall correspond to the description stated in section 5. The file with the highest number shall always be the latest Table of Contents representing the latest state of the Table of Contents. In addition, the *parent_toc* attribute (see sections 4.4.3 and 5.3) of the Table of Contents file should be used to document the revision history of the container where applicable.

If file `TOC`*`n`*`.xml`, *`n`*`>1` is present, but `TOC`*`m`*`.xml`, $m = n - 1$ is missing, then it indicates that the audit trail is not captured internally within the ACS container and that consumers will need to fetch external resources in order to inspect the revision history as specified in sections 4.4.3 and 5.3.

4.4.2 Other Files in ACS

The ZIP container should contain cytometry data and metadata files and it may contain additional files. While the use of other ISAC-endorsed formats is preferred where applicable, these components may use arbitrary file formats, which are not specified, neither restricted, by this specification.

Data, metadata, and additional files in ACS may use any file names except for a character string constructed as a concatenation of the character string `TOC`, ASCII representation of any number, and a character string `.xml` (i.e., the `TOC`*`n`*`.xml` naming schema shall be used for the Table of Contents XML files only). This restriction assures that there will be no naming collision of the Table of Contents XML file(s) with other files within ACS.

4.4.3 Audit Trail Implementation

Except for cases where keeping the original data is not desirable (such as anonymization of clinical data), either one of these 2 approaches should be followed when modification of an ACS container needs to be made:

a) **External Audit Trail Implementation**

- Do not change the original ACS file, keep it immutable and assign a fixed URL to the ACS container. This URL should use one of the `http`, `https`, or `ftp` schemes, and it should be globally de-referenceable (i.e., localhost-based URLs should not be used).
- Create a new version of the ACS container and revise resources (files) in this container as necessary. Resources that haven't changed can either be copied to the new container or referenced using the URI fragment identifier syntax [15]. For example, file /fcs/file01.fcs in http://xy.com/acs1.acs can be referenced as http://xy.com/acs1.acs#/fcs/file01.fcs.
- Increment the version number in the name of the Table of Contents XML file, i.e., if *`n`* is the highest number of any `TOC`*`i`*`.xml` file, then the Table of Contents in the new ACS container shall be named `TOC`*`m`*`.xml` where $m = n + 1$.

— Fill in the `parent_toc` attribute in the new Table of Contents XML file with the URL of the Table of Contents XML file of the original ACS file (i.e., the ACS file that the changes are based on). This URL shall use the fragment identifier syntax [15]. For example, file TOC3.xml in http://xy.com/acs1.acs shall be referenced as http://xy.com/acs1.acs#/TOC3.xml. Note that this will allow for tracking of the previous version of the Table of Contents XML file as well as the whole ACS container.

b) **Internal Audit Trail Implementation**

— Do not physically change any of the files within ACS; this includes the Table of Contents file(s), data, metadata, and any other files that may be present in ACS. A new version of Table of Contents will reflect the updated structure and content of the ACS container, see below.

— Create a new version of the Table of Contents XML file reflecting the latest state of the ACS. If the latest version of the Table of Contents XML file was *n*, number your Table of Contents XML file as *n*+1 (even if the changes are based on an earlier version of the ACS that is different from *n*).

— Fill in the `parent_toc` attribute in the Table of Contents XML file with the URI of the Table of Contents file of the ACS version that the latest changes are based on (this may or may not be the *n* version of the Table of Contents); see section 5 for details on the Table of Contents XML file format.

— If data, metadata or other files within the container are being changed, keep the original files and save a new version with a different file name. Adding a number-based suffix that corresponds to the number of the current Table of Contents is recommended.

— Describe current data, metadata and other files in the new version of the Table of Contents XML file. The new Table of Contents XML file shall contain all information necessary to interpret the latest version of the ACS, i.e., the previous version of Table of Contents is needed for revision history (audit trail) purposes. All current information shall be available from the latest version of the Table of Contents.

— If data, metadata or other files are being deleted from the container, do not physically delete them. Instead, keep the original files in the container but do not describe these in the new version of the Table of Contents.

5. ACS Table of Contents Format

5.1 General

An ACS Table of Contents file is a valid XML [2] file placed in the root of the ACS container (root of the ZIP file) with a name constructed as a concatenation of the character string `TOC`, ASCII representation of a positive integer, and a character string `.xml`. For example, `TOC1.xml`, `TOC2.xml`, `TOC3.xml`, etc. are valid file names for the Table of Contents XML file. When a file name `TOC`*n*`.xml` is created ($n \in \mathbf{N}$, $n > 1$) is created, then the `parent_toc` attribute shall be present in that file.

5.2 Table of Contents File Structure

An ACS Table of Contents XML shall successfully validate against the provided `TOC.v1.0.xsd` XML schema file. The TOC is the main element expected in the XML file. This element should define namespace listed in section 1.6 so that these can be reused within the XML file.

An example of a header of an ACS Table of Contents XML file follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<toc:TOC
  xmlns:toc = "http://www.isac-net.org/std/ACS/1.0/toc/"
  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
  xmlns:sig = "http://www.w3.org/2000/09/xmldsig#"
  xsi:schemaLocation = "http://www.isac-net.org/std/ACS/1.0/toc/
                        http://flowcyt.sf.net/acs/toc/TOC.v1.0.xsd">
```

... Description of the contents of the ACS file ...

```
</toc:TOC>
```

5.3 Parent Table of Contents

If an ACS Table of Contents is being created based on another existing Table of Contents XML file, the URI of the original Table of Contents shall be stated in the `parent_toc` attribute in the TOC element of the Table of Contents XML file. This URI should use the `file`, `http`, `https`, or `ftp` schemas and it should be globally de-referenceable (i.e., localhost-based URLs should not be used). Typical values of the `parent_toc` attribute are `file:///TOC1.xml`, `file:///TOC2.xml`, etc. for internal audit trail implementations. External audit trail implementations shall reference the parent Table of Contents using a URI of the parent ACS container with the fragment identifier as specified in section 4.4.3 a), for example, `http://xy.com/acs1.acs#/TOC1.xml`.

For example, consider the following use case: the user opens an ACS file with a single Table of Contents XML file named `TOC1.xml`. Then, the user opens an FCS file present within the ACS container, manually gates data in the FCS file and saves these gates in a new Gating-ML file, all bundled in an updated version of the ACS container. In this case, a new Table of Contents XML file shall be created and named `TOC2.xml`. This shall include the original contents of the `TOC1.xml` plus the description of the new Gating-ML file with appropriate relation indicated between the FCS and Gating-ML instances. In addition, the new `TOC2.xml` should indicate that it has been created based on `TOC1.xml`. For that purpose, the `parent_toc` attribute in the TOC element shall be used.

If the audit trail is implemented internally (i.e., previous versions of the Table of Contents are kept in the updated ACS file), then the header of the `TOC2.xml` file may be created as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<toc:TOC toc:parent_toc="file:///TOC1.xml"
  xmlns:toc = "http://www.isac-net.org/std/ACS/1.0/toc/"
  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
  xmlns:sig = "http://www.w3.org/2000/09/xmldsig#"
  xsi:schemaLocation = "http://www.isac-net.org/std/ACS/1.0/toc/
                        http://flowcyt.sf.net/acs/toc/TOC.v1.0.xsd">
```

... Description of the contents of the ACS file ...

```
</toc:TOC>
```

If the audit trail is implemented externally (i.e., previous versions of the Table of Contents are not in the updated ACS file), then the header of the `TOC2.xml` file may be created as follows (replace `https://xy.com/acs1.acs` with a proper URL pointing to the previous version of the ACS container):

```
<?xml version="1.0" encoding="UTF-8"?>
<toc:TOC toc:parent_toc="https://xy.com/acs1.acs#/TOC1.xml"
  xmlns:toc = "http://www.isac-net.org/std/ACS/1.0/toc/"
  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
  xmlns:sig = "http://www.w3.org/2000/09/xmldsig#"
  xsi:schemaLocation = "http://www.isac-net.org/std/ACS/1.0/toc/
                        http://flowcyt.sf.net/acs/toc/TOC.v1.0.xsd">
```

... Description of the contents of the ACS file ...

```
</toc:TOC>
```

5.4 Listing Files in ACS Table of Contents

The ACS container should contain cytometry data and metadata files and may also contain additional files. All files that are considered present in the current version of the ACS container shall be listed in the ACS Table of Contents using the `file` element within the `TOC` element. For the purpose of an audit trail / revision history, there may also be files physically present in the ACS container, which are not listed in the latest version of the ACS Table of Contents. These files have either been deleted or replaced with an updated version and are considered not to be present in the container. However, if internal audit trail implementation is used, then those files should still be physically kept within ACS so that older versions of the ACS Table of Contents referencing these files remain valid.

5.4.1 File Resource Identifiers

The `file` element includes a single required attribute called `URI`. This attribute shall be used to provide a Uniform Resource Identifier (URI) of a file either present within the container or referenced externally. An example of a minimum entry of a file within ACS follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<toc:TOC ... TOC element definition ... >

  <!-- URI is the required minimum in order to list a file in the ACS TOC -->
  <toc:file toc:URI="file:///file01.fcs" />

      ... Other Description of the contents of the ACS file ...

</toc:TOC>
```

For files that are required to interpret the contents of the ACS container, the URI shall be in the form of a Uniform Resource Locator (URL) that is based on one of the `file`, `http`, `https`, or `ftp` schemes. The `file` scheme shall only be used to reference files within the container. In these cases, the container itself is understood as the local file system. The slash character shall be used to indicate the root of the container and to separate directories in the file path. File-based URLs referring to the local file system outside of the container (e.g., "file://C:\My Documents\FCS\file1.fcs") shall not be used in any case.

The `http`, `https`, or `ftp` schemes may be used to reference files outside of the container. In these cases, these URLs shall be globally de-referenceable (i.e., localhost-based URLs shall not be used).

URLs based on other than `file`, `http`, `https`, and `ftp` schemes and Uniform Resource Names (URNs) may only be used for files that are not required to interpret the contents of the ACS container. For example, the URN urn:issn:1552-4957 may be used to reference a publication related to the contents of the container.

5.4.2 File Types

The `file` element includes an optional attribute called `mimeType` that should be used to specify the types of listed files. The Multipurpose Internet Mail Extensions (MIME) Media Type [16] should be used whenever possible to specify the format of the listed file. Examples of useful MIME Media Types include: application/vnd.isac.fcs, application/dicom, application/postscript, application/msword, application/pdf, application/netCDF, image/bmp, image/gif, image/jpeg, image/jp2, image/png, image/svg+xml, image/tiff. If an appropriate MIME Media Type is not registered by the Internet Assigned Numbers Authority (IANA), then a format name should be created in the form that corresponds to the established MIME Media Type naming scheme, e.g., application/vnd.treestar.flowJo-ws+xml, application/vnd.isac.gating-ml+xml. An example of a `file` element that lists its MIME Media Type follows:

```
<toc:file
  toc:URI = "file:///file01.fcs"
  toc:mimeType = "application/vnd.isac.fcs" />
```

5.4.3 Free Text File Description

The `file` element includes an optional attribute called `description` that may be used to provide a free text description for a file resource. An example of how this may be used follows:

```
<toc:file
  toc:URI = "file:///file01.fcs"
  toc:mimeType = "application/vnd.isac.fcs"
  toc:description = "Mouse B6.Cg 1234; PB, C4, CD8, CD16/CD32" />
```

5.5 Associations Between Files

The `file` element includes an optional sub-element called `associated` that should be used to provide information about associations between files. Each file may be associated with zero, one, or multiple other files using zero, one, or multiple `associated` sub-elements, respectively. For example, a Gating-ML file may be associated with an FCS data file to indicate that gates described in the Gating-ML file are supposed to be applied on data in the FCS file (see below for further examples). The `associated` element includes two required attributes: `with` and `relationship`. The `with` attribute shall be used to reference the target of the association as a URI and the `relationship` to state the type of relation. The mentioned example may be encoded as follows:

```
<toc:file
  toc:URI = "file:///file01.fcs"
  toc:mimeType = "application/vnd.isac.fcs">
  <toc:associated
    toc:with = "file:///gates/gates01.xml"
    toc:relationship = "gating description" />
</toc:file>
```

For the `with` attribute, the same rules as stated in section 5.4.1 shall be followed to create the URI of the resource that is the target of the association.

The value of the `relationship` attribute shall indicate the type of relation between the two files. ISAC is maintaining a registry of relation types (see below). If possible, one of the pre-defined relations should be used using the exact relationship wording. If there is no suitable relationship for the required association in the ISAC registry, please submit a relationship type to the ISAC Data Standards Task Force; see http://www.isac-net.org for contact details and the updated contents of the registry.

The contents of the relationship registry as of the date of this document is as follows:

Relationship	**Description**
gating description	Relation from a data file (e.g., a list mode data file) to a gating description file, such as a Gating-ML file. This relationship indicates that gates described in a gating description file are applicable to data in the data file.
compensation	Relation from a list mode data file to an external file describing the fluorescence compensation that is supposed to be applied on data stored in the list mode data file.

description	In this case, data in the list mode data file shall be stored uncompensated. This relation shall not be used if data is stored compensated already. Also, this relation is not necessary if compensation details are included as part of the list mode data file, e.g., in the $SPILLOVER keyword according to the FCS 3.1 specification [3].
compensated version	Relation from an uncompensated data file to a compensated version of the same data.
classification results	Relation from a data file (e.g., a list mode data file) to a file with classification results, such as event-based classification (e.g., results of gating) of a list mode data file.
project/workspace	Relation from a data file to a project (sometimes also called workspace) description file of analytical software used to analyze data in the data file.
instrumentation settings description	Relation from a data file to a file describing the instrument settings used for acquisition of these data.
sample specimen description	Relation from a data file to a file describing the sample (or specimen) that has been used to generate data in the data file.
analysis description	Relation from a data file to a file describing the methodology of the data analysis.
results description	Relation from a data file to a file describing results of the data analysis.
related publication	Relation from a data file to a related publication. A Uniform Resource Name (URN) may be used in the `with` attribute to specify the referenced publication.
digital signature	Relation from file being signed to a file with a detached digital signature provided outside of the TOC file but still within the ACS container. The format of this signature file is not specified by ACS specification; external encryption standards may be used.

5.6 Additional Information in the Table of Contents

Additional information may be provided in the Table of Contents XML file using the `additional_info` element. This element may be used either as a sub-element of the `TOC` element to provide additional information related to the whole ACS, or as a sub-element of the `file` element to provide additional information related only to a specific file within the ACS container. There may be zero, one or multiple `additional_info` elements related to each file in ACS. Similarly, there may be zero, one or multiple `additional_info` elements related to the whole ACS container. As long as the Table of Contents remains a valid XML file, the contents of the `additional_info` element is not restricted by this specification. However, the `additional_info` element shall not be used as replacement for describing associations describable by the `associated` element. An example of how the `additional_info` element can be used follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<toc:TOC ... TOC element definition ... >

  <toc:file
    toc:URI = "file:///file01.fcs"
    toc:mimeType = "application/vnd.isac.fcs">

    <toc:additional_info>
      Optional additional information about file01.fcs
      Mixtures of free text and XML fragments may be used
      <marker>CD4</marker><marker>CD8</marker><marker>CD16</marker>
      <note>Here we may have any custom information in <any/> structure</note>
    </toc:additional_info>
  </toc:file>

  <toc:additional_info>
    Optional additional information about the whole ACS
    Again, mixtures of free text and XML fragments may be used
    <author>Josef Spidlen</author>
    <contact>
      <email>jspidlen@bccrc.ca</email><phone>+1.604.675.8000x7755</phone>
    </contact>
  </toc:additional_info>
```

... Other Description of the contents of the ACS file ...

```
</toc:TOC>
```

5.7 Digital Signatures

Digital signatures may be provided to digitally sign various components (files) within an ACS container. Two implementations are supported by the ACS specification: signatures may be placed either directly in the ACS Table of Contents XML file or they may be stored in separate (detached) files, i.e., outside of the Table of Contents XML file but still within the ACS container.

5.7.1 Detached Signatures

The ACS container may contain detached digital signatures of selected (or all) components (files) within ACS. In this case, a separate file with digital signature should be stored inside ACS and files that are being digitally signed should have a digital signature relation (see section 5.5) pointing to the appropriate signature file. In this case, the format of the digital signature file is not specified by the ACS specification and external encryption standards may be used.

5.7.2 Embedded Signatures

The ACS container may contain digital signatures placed directly in the ACS Table of Contents XML file. For that purpose, the `signature` element may be used within the main `TOC` element in the ACS Table of Contents. The type of the `signature` element is defined as the `SignatureType` type of the http://www.w3.org/2000/09/xmldsig# name space as defined by W3C Recommendation, XML Signature Syntax and Processing Second Edition [8]. Implementation details for this signature type are fully defined by the referenced W3C Recommendation. Further information in this section (5.7.2) is given only as informative introduction. For a normative reference, please refer to W3C Recommendation [8].

The XML Signature W3C Recommendation recognizes several algorithm types related to the digital signing and verification process:

- Canonicalization methods: Any XML document is part of a set of XML documents that are logically equivalent within an application context, but which vary in physical representation based on syntactic changes permitted by XML [2] and namespaces in XML. This would create an issue if XML documents were to be signed directly. Therefore, there are so called canonicalization methods that generate the physical representation, the canonical form, of an XML document that accounts for the permissible changes. For the purpose of XML signatures, the W3C Recommendation Canonical XML, Version 1.0 or Version 1.1 are commonly being used. These can either create a canonical XML with or without comments.
- Digest methods: The SHA1 digest is commonly being used.

— Signature methods: Two digital signature methods are supported by the XML signatures W3C Recommendation: DSA with SHA1 (DSS) and RSA with SHA1.

In addition, the XML Signature W3C Recommendation supports algorithms for encoding binaries within XML (i.e., Base64 encoding), message authentication codes (i.e., Hash-based message authentication codes) and transformations (e.g., XSLT, XPath, or Enveloped signature), which may be needed in certain cases. In a typical case, one selects a canonicalization method and creates a canonical form of the resource being signed, then selects a digest method and creates a digest of the resource in canonical form, and finally, the digest is digitally signed using one of the supported signature methods. Each of the supported algorithms has a unique identifier so that it can be unambiguously referenced. The example below demonstrates how these three algorithms can be combined within the `sig:SignedInfo` element in order to sign a resource (a Gating-ML file in this case). The actually signature value follows encoded as Base64 in the `sig:SignatureValue` element after the `sig:SignedInfo` element.

```
<?xml version="1.0" encoding="UTF-8"?>
<toc:TOC ... TOC element definition ... >

  ... Description of the contents of the ACS file ...

  <toc:signature Id="Signature1">
    <sig:SignedInfo>
      <sig:CanonicalizationMethod
        Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
      <sig:SignatureMethod
        Algorithm="http://www.w3.org/2000/09/sig#dsa-sha1" />
      <sig:Reference URI="file:///gates/gates01.xml">
        <sig:DigestMethod Algorithm="http://www.w3.org/2000/09/sig#sha1" />
        <sig:DigestValue>
          82947eb5aa1e52979f0acfb1bf663c78ae8bbe6d
        </sig:DigestValue>
      </sig:Reference>
    </sig:SignedInfo>
    <sig:SignatureValue>
LIpsnId5rpoi+mQNwxYWoTvc88spbgsB4HJpv/EvJGoTOj1U818wtWsPL/kU2NbiVvxUxKHdX4Ez
scGCESDz0yotn40qllB0WtB1PilG3wL+5SELdtFo88fzNp+HVht9RfrbhVrS5R1+SLYI+8xd0KO7
    ... Base64-encoded signature value ...
HEV6v4cGDCVMTVfhhJq8e6Iz9YFOaznJvmM6pqauqvDn3yH9wVfNd/KDmNxJyfmLNRftx+E0GA==
    </sig:SignatureValue>
  </toc:signature>
</toc:TOC>
```

The `sig:SignatureValue` element may also be followed by a `sig:KeyInfo` element to distribute the public key associated with the signature. X509 certificates, PGP keys and other options are supported by the XML Signature W3C Recommendation; please refer to W3C Recommendation, XML Signature Syntax and Processing Second Edition [8] for further details.

Annex A

Bibliography – Informative

[1] PKWARE. Application Note on the .ZIP file format version 6.2.0. Available at: http://www.pkware.com/documents/APPNOTE/APPNOTE-6.2.0.txt. Accessed 07/28, 2010.

[2] World Wide Web Consortium (W3C). Extensible Markup Language (XML). Available at: http://www.w3.org/TR/REC-xml/.

[3] Spidlen J, Moore W, Parks D, et al. Data file standard for flow cytometry, version FCS 3.1. Cytometry A. 2010;77:97-100.

[4] Spidlen J, Leif RC, Moore W, Roederer M, Brinkman RR, International Society for the Advancement of Cytometry Data Standards Task Force. Gating-ML: XML-based gating descriptions in flow cytometry. Cytometry A. 2008;73:1151-1157.

[5] World Wide Web Consortium (W3C). XML Schema Part 0: Primer Second Edition. Available at: http://www.w3.org/TR/xmlschema-0/.

[6] World Wide Web Consortium (W3C). XML Schema Part 1: Structures Second Edition. Available at: http://www.w3.org/TR/xmlschema-1/.

[7] World Wide Web Consortium (W3C). XML Schema Part 2: Datatypes. Available at: http://www.w3.org/TR/xmlschema-2/.

[8] World Wide Web Consortium (W3C). W3C Recommendation - XML Signature Syntax and Processing (Second Edition). Available at: http://www.w3.org/TR/xmldsig-core/2010.

[9] 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.

[10] 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.

[11] W3C. W3C Recommendation - Namespaces in XML 1.0 (Second Edition). Available at: http://www.w3.org/TR/xml-names/.

[12] Spidlen J, Leif RC, Brinkman RR. Requirements for a data file standard format to describe cytometry and related analytical cytology data. Available at: http://downloads.sourceforge.net/flowcyt/Requirements-v070920.pdf. Accessed 03/08, 2008.

[13] Comparison of file archivers. Available at: http://en.wikipedia.org/wiki/Comparison_of_file_archivers.

[14] Info-ZIP. Available at: http://www.info-zip.org/.

[15] Internet Engineering Task Force Network Working. Request for Comments (RFC) 3986. Uniform Resource Identifier (URI): Generic Syntax. Available at https://tools.ietf.org/html/rfc3986

[16] Internet Assigned Numbers Authority (IANA). MIME Media Types. Available at: http://www.iana.org/assignments/media-types/. Accessed 05/15, 2008.