INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11 MPEG2017/M42588
April 2018, La Jolla, US

Source: **DMAG-UPC**
Status: **Proposal**
Title: **Proposal for a Study of ISO/IEC CD 23092-3 Genomic Information Metadata and APIs**
Authors: **Jaime Delgado, Silvia Llorente, Daniel Naro (Distributed Multimedia Applications Group – Universitat Politècnica de Catalunya)**

See enclosed document.

ISO 23092-3:2018

ISO/IEC JTC1/SC 29/WG11

Secretariat: JISC

Information Technology — ISO/IEC 23092 — Part 3: Genomic Information Metadata and Application Programming Interfaces (APIs)

CD stage

Warning for WDs and CDs

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by ISO/JTC1, Subcommittee SC29, Working Group 11.

This is the first edition of ISO/IEC 23092 Part 3. ISO/IEC 23092, Genomic Information Representation, is composed of the following parts:

Part 1: Transport and Storage of Genomic Information

Part 2: Coding of Genomic Information

Part 3: Genomic Information Metadata and Application Programming Interfaces (APIs)

Part 4: Reference Software

Part 5: Conformance Testing

Introduction

The development of High-Throughput Sequencing (HTS) technologies enables the usage of genomic information as everyday practice in several fields. The growing volume of data generated requires efficient representation of the genomic information to support interoperability among tools and systems. The lack of appropriate standard representations and efficient compression technologies of genomic data is widely recognized as a critical element seriously limiting its application potential in all fields using or willing to use genomic data.

This document was developed in response to worldwide demand for new effective interoperable solutions for genomic information processing applications covering all the chain from sequencing to storage and analysis.

This document includes the specification of syntax and semantics for the metadata and APIs (Application Programming Interfaces) for genomic information representation.

Information Technology — Genomic Information Representation — Part 3: Genomic Information Metadata and Application Programming Interfaces (APIs)

1 Scope

This document includes the specification of information metadata, SAM interoperability, protection metadata and programming interfaces to genomic information:

- Metadata storage and interpretation for the different available layers are treated in Section 5.
- Protection elements (providing confidentiality, integrity and privacy rules at the different layers coded in compliance with Parts 1 and 2 of ISO/IEC 23092) are treated in Section 6.
- Mechanisms for backward compatibility with existing SAM content, and exportation to this format are treated in Section 7.
- Interfaces to access genomic information coded in compliance with Parts 1 and 2 of ISO/IEC 23092 are treated in Section 8.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 23092-1 Information technology -- Genomic Information Representation -- Part 1: Transport and Storage of Genomic Information

ISO/IEC 23092-2 Information technology -- Genomic Information Representation -- Part 2: Coding of Genomic Information

ISO/IEC 23092-4 Information technology -- Genomic Information Representation -- Part 4: Reference SW

ISO/IEC 23092-5 Information technology -- Genomic Information Representation -- Part 5: Conformance

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at https://www.iso.org/obp

3.1

access unit
Logical data structure containing a coded representation of information to facilitate the bit stream access and manipulation.

3.2

alignment
A sequence read mapped on a reference sequence

3.3

BAM
Compressed binary version of SAM

3.4

base
In the context of this document it is used as synonymous of nucleotide

3.5

CIGAR string
It is a sequence of base lengths and the associated operations used to indicate things like which bases align (either a match/mismatch) with the reference, are deleted from the reference, and are insertions that are not in the reference

3.6

dataset
Compression unit containing sequence reads and possibly alignment information

3.7

dataset group
Collection of one or more datasets

3.8

FASTA
GIR that includes read headers and sequence reads (nucleotides sequences)

3.9

FASTQ
GIR that includes FASTA plus quality scores

3.10

genomic record
Data structure encoding either a single sequence read or a paired sequence read optionally associated with alignment information, read identifier and quality values

3.11

genomic record length
Distance between the left-most mapped base coded in the record and the right-most mapped base coded in the record

3.12

genomic range
Interval of positions on a reference sequence defined by a start position s and an end position e such that s ≤ e. the start and the end positions of a genomic range are always included in the range

3.13

genomic record position
Position of the left-most mapped base of the genomic record on the reference genome

3.14

indel
An additional or missing nucleotide in a DNA sequence with respect to a reference DNA sequence

3.15

mapped base
It is either:

- a base of the aligned read matching the corresponding base on the Reference Sequence
  or
- a base of the aligned read that does not match the corresponding base (a.k.a. single nucleotide polymorphism)

3.16

quality score
It is assigned to each nucleotide base call in automated sequencing processes. It expresses the base-call accuracy

3.17

read header
Each sequence read stored in FASTA and FASTQ format starts with a textual field called “read header” containing a sequence identifier and an optional description

3.18

reference genome
It is a digital nucleic acid sequence database, assembled by scientists as a representative example of a species’ genetic material

3.19

reference sequence
It is a sequence of nucleotides associated to a one-dimensional integer coordinate system for which each integer coordinate is associated to a single nucleotide. Coordinate values can only be equal or larger than zero. This coordinate system in the context of this standard is zero-based (i.e. the first nucleotide has coordinate 0 and is said to be at position 0) and linearly increasing from left to right

3.20

SAM
GIR that is human readable and includes FASTQ plus alignment and analysis information

3.21

(genomic) segment
A contiguous sequence of nucleotides

3.22

sequence read
The readout, by a specific technology more or less prone to errors, of a continuous part of a segment of nucleotides extracted from an organic sample

3.23

template
A DNA sequence which is sequenced on a sequencing machine or assembled from sequence reads

4 Abbreviations

GIR Genomic Information Representation

5 Conventions

5.1 Character encoding

This specification utilizes UTF-8 character encoding.

6 Information metadata

6.1 Introduction

The metadata structure and the set of elements is specified using XML [1].

This standard defines a minimum core set of metadata elements, which can then be extended by users and applications by including extra information elements. Metadata sets are specified for a Dataset Group as specified in Part 1 of this Standard and for a Dataset also specified in Part 1 of this Standard.

Extensions to (i.e., new elements for) the metadata set specified in this standard are represented with an information type identifier, a value and a pointer to a resource documenting the semantics of the given information type.

Metadata profiles are specific subsets of metadata sets specified using mechanisms provided in the standard. A metadata profile specified in this Standard may correspond to well-known metadata sets specified or used out of this standard, such as those in ENA or EGA [2] and NCBI specifications [3], as examples. This allows easy interoperability with already existing systems. A metadata profile includes a subset of core elements described in this standard, and a set of new elements specified with the extensions mechanism (see Clause 6.5).

The rest of Clauses of this standard specify Dataset group metadata (Clause 6.2), Dataset metadata (Clause 6.3), Extensions (Clause 6.4) and Profiles (Clause 6.5).

6.2 Dataset Group metadata

Dataset group metadata is associated to a genomic study and is stored within the dataset group metadata container. Table 1 presents the core set of metadata elements in a dataset group metadata container. Those elements that are necessary to identify and process the dataset group are classified as mandatory.

Table 1: Dataset group's metadata core set

Element name	Element type	Mandatory
Title	String	Yes
Type	Controlled vocabulary	Yes
Abstract	String	No
Project centre name	ProjectCentre type	No
Description	String	No
Samples	ListOfSamples type	Yes
Extensions	ListOfExtensions type	No

The conversion of Table 1 to an XML schema is done as follows. Each row is translated into one element of the type indicated in the element type column, with a maximum occurrence of one and a minimum occurrence depending on the mandatory nature of the element. In the case where the type is controlled

vocabulary, the XML schema represents the data as a string, but all words not included in the list of controlled vocabulary are considered as ill-formed. The resulting schema is provided in Annex A.1.

As previously introduced, an extensions type is the combination of three fields: the value, the identifier of the extension, and a link to a resource documenting the interpretation of the field. In the XML schema, this is translated as an element with two attributes: the identifier (of type string) and the resource (a URL); the value is represented as the element's text as UTF-8 characters text (in case of binary information, Base64 encoding is used). Additionally, a Boolean attribute of the element indicates if the extension is only relevant to the dataset group, or if the dataset also inherits it. The resource documentation might be human readable, and the extensions parsing is not required.

As Table 1 indicates, certain elements can be described with basic types, but other element types require more complex descriptions, such as the sample type. For those elements, their respective core set of fields is provided. Table 2 lists those for the sample type, and Table 3 for the project centre type.

Table 2: `Sample`'s metadata core set

Field name	Field type	Mandatory
`TaxonId`	Integer	Yes
`Title`	String	No
`Extensions`	ListOfExtensions type	No

Table 3: `Project centre`'s metadata core set

Field name	Field type	Mandatory
`ProjectcentreId`	Integer	Yes
`Title`	String	No
`Extensions`	ListOfExtensions type	No

6.3 Dataset metadata

Dataset metadata is associated to a genomic analysis and is stored in the dataset's metadata element. A dataset metadata element overwrites the corresponding element whose values differ from the one indicated at the dataset group level (i.e., the new value in the dataset is a specialization of the value at the dataset group level).

Table 4 specifies the core elements for dataset metadata. No elements are mandatory since they are inherited (unless overwritten with new values) from the dataset group metadata.

For example, we might have datasets for patients A, B and C; therefore the dataset group's metadata includes a list of samples representing A, B and C. The datasets then provide only one sample description (respectively of A, B or C). The base set of elements in the dataset's metadata is the same as for the dataset group, but the elements are not mandatory (so there is no need to repeat them), since per default their values are considered equal to the values indicated in the dataset group. This is always

the case for the values belonging to the core set, or by default for the extensions except for those cases that have the inheritance parameter set to false.

As in the case of the dataset group metadata, the information is represented as an XML document, the schema of which is derived from Table 4, using the previously described methodology. The resulting schema is provided in Annex A.2.

Table 4: Dataset's metadata core set

Element name	Element type	Description	Mandatory
Title	String		No
Type	Controlled vocabulary		No
Abstract	String		No
Project centres	ListOfProjectCentres type	Contact information of centres participating in the generation of the described study's data.	No
Description	String		No
Samples	ListOfSamples type	Identification of the samples, based on taxonomy/scientific name, common name or anonymized name and further attributes defined in a controlled library.	No
Extensions	ListOfExtensions type		No

Also as in the case of the dataset group, the extension mechanism is available to include new attributes where necessary. See section on extensions for an example in the case of dataset's metadata.

6.4 Mechanism for extensions of the metadata set

A mechanism for adding new elements to the different core metadata sets (dataset group and dataset levels) is provided.

An extended element consists of:

- information type identifier (provided in the form of a URI),
- value.

In the case of extensions at the dataset group level, a fourth value, the inheritance flag of type Boolean is optionally present. By default, and even if not present, it is considered to be equal to True. In case of being set to True, the value of the extension is inherited by the datasets belonging to the group. If set to False, the value only applies to the dataset group.

The extension schema is defined in Annex A.1. Through the use of extensions, the core metadata sets can be adapted to multiple use cases. The standard defines profiles (see Clause 6.5), which rely on well-known extensions, defined in the standard and for which the URI pointer is known. To be compliant with a profile specified in Part 3 of this Standard, a tool has to implement the list of extensions included in the profile.

6.4.1 Examples of extensions

This sub-clause presents two examples:

- For dataset group, based on currently existing metadata sets, as those from ENA, EGA, NCBI or others.
- For dataset, using the concept of label from Part 1.

6.4.1.1 Example for Dataset group metadata extensions

In order to formalize the support of the broad sets of attributes used by the SRA schema [2] or NCBI [3] specifications, the extensions mechanism could be used. For example, in the case of SRA's sample metadata [2], the value of the pointer to the semantics would link to an additional schema defining the set of elements presented in Table 5 (taken over the schema provided by EBI in [2]). In this case, the fact that the semantic specification is provided as an XML schema would simplify an automatic integration of the content. The value of the extension would be a string containing the XML file name.

Table 5: `Sample`'s metadata element extended for a specific profile

Field name	Field type	Mandatory
Sample Name – scientific	String	No
Sample Name – common name	String	No
Sample Name – anonymized name	String	No
Sample Name – individual name	String	No
Description	String	No
Links	URI	No

In the case of NCBI's BioSample metadata, the specification is split in multiple cases [3]. Each of these subtypes is a different extension, the definition of which is constructed on the same principles: one XML element per attribute, using the data types indicated in the specification.

Although BioSample provides compatibility with the Minimum Information about any (x) Sequence (MIxS) [4], extensions dedicated to MIxS could be also specified, once again using the same strategy.

6.4.1.2 Example for Dataset metadata extensions

Part 1 of this Standard introduces the concept of dataset's `label`, which allows giving unique names to regions of the data. As such, this does not allow documenting what that region represents. A possible extension to the `sample` metadata is a translation tool from the label name to an ontology term, using the information indicated in Table 6.

Table 6: `Sample`'s metadata extended with a `label` linked to an ontology

Field name	Field type	Mandatory
Label name	Integer	Yes
Ontology term	URN	Yes

6.5 Metadata Profiles

Profiles are specific metadata sets. They are specified using the mechanisms provided in Clause 6.5.1. Clause 6.5.2 provides a formalized profile.

A profile corresponds to a well-known metadata set specified or used out of this standard, such as the one defined to support the Run sets of the SRA (Sequence Read Archive) schema [2].

A profile includes a subset of the core elements described in this standard, and a set of new elements specified with the extensions mechanism (see Clause 6.4).

6.5.1 Metadata profiles specification

The metadata schemas Annexes A.1 and A.2 define an attribute in the Dataset Group and Dataset metadata XML element, to define the profile being used. The profile is identified with a URI, but in case no profile is active, the attribute is not used.

6.5.2 Example of metadata profile: Run

The MPEG-G dataset metadata shares characteristics with both the concepts of run and analysis as used by EGA [5]. This MPEG-G profile provides interoperability with the existing metadata schemas, such that an MPEG-G dataset is interoperable with an EGA run element.

Table 7 presents the set of elements included in this profile. Some of them are already part of the core metadata set, and the rest are the extensions needed to match the elements specified in [2].

This profile is identified with the URI : "`urn:mpeg:mpeg-g:metadata:profile:ega:run`".

To be compliant with the profile, the following extensions must be present. The following extensions can either be stored in the Extensions element of the Dataset Group metadata (in which case it has to be marked as inheritable), or in the Extensions field of each Dataset. This means that either through inheritance or because the value is provided within the dataset, each dataset implementing the profile has the following extensions. In case the extension is provided at the dataset group level, an extension with different values can also be present at the dataset, thus overwriting the values.

- (Optional) One extension of type "`DatasetGroupAttributes`" as specified in Annex A.6, identified with a type URI "`urn:mpeg:mpeg-g:metadata:extension:ega:DatasetGroupAttributes`"
- (Optional) One extension of type "`DatasetGroupLinks`" as specified in Annex A.6, identified with a type URI "`urn:mpeg:mpeg-g:metadata:extension:ega:DatasetGroupLinks`"
- (Optional) One extension of type "`DatasetGroupRelatedStudies`" as specified in Annex A.6, identified with a type URI "`urn:mpeg:mpeg-g:metadata:extension:ega:RelatedStudies`"
- One extension of type "`StudyType`" as specified in Annex A.6, identified with a type URI "`urn:mpeg:mpeg-g:metadata:extension:ega:StudyType`"

- (Optional) One extension of type "ProjectIdentification" as specified in Annex A.6, identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:ProjectIdentification"

- One extension of type "ObjectType" as specified in Annex A.5, identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:DatasetGroupObjectType"

- Sample elements: for each sample one extension of type "EGA_SampleType" as specified in Annex A.3 from the namespace "urn:mpeg:mpeg-g:metadata:extension:ega", identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:EGA_SampleType"

- One extension of type "SpotDescriptorType" as described in "SRA.common.xsd", identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:SpotDescriptorType"

- One extension of type "PlatformType" as described in "SRA.common.xsd", identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:PlatformType"

- One extension of type "ProcessingType" as described in "SRA.common.xsd", identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:ProcessingType"

- (Optional) One extension of type "DatasetType" as specified in Annex A.7, identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:DatasetType"

- (Optional) One extension of type "DatasetAttributes" as specified in Annex A.7, identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:DatasetAttributes"

- (Optional) One extension of type "DatasetLinks" as specified in Annex A.7 identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:DatasetLinks"

- One extension of type "RunExtension" as specified in Annex A.8, identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:DatasetLinks"

- One extension of type "ExperimentExtensionElement" as specified in Annex A.4 from the namespace "urn:mpeg:mpeg-g:metadata:extension:ega", identified with a type URI "urn:mpeg:mpeg-g:metadata:extension:ega:ExperimentExtensionElement".

Furthermore, to be compliant, the following restrictions have to be observed:

- description element in the dataset group's metadata schema must be mandatory

- the TaxonID field can only be equal to 9096 (human).

The contents of SRA.common.xsd and SRA.run.xsd can be found at [2].

This profile extends the MPEG-G dataset core metadata to match the run schema used by EGA. The file element (that points towards the file from within the metadata) from SRA's run metadata schema is not needed in MPEG-G because the metadata is placed within the element it refers to.

7 Protection metadata

7.1 Introduction

Part 1 of this Standard defines containers (or `gen_info` structures) to support the protection of the information at the different layers of the hierarchy. These containers provide information to guarantee, if desired, the confidentiality and integrity of the information, alongside the privacy rules to be applied to the information they refer to. The protection `gen_info` elements are constructed as XML content, the root element of which is of type "Protection". The specific XML Schemas are included as Annex A.9 for Datasetgroup's protection and Annex A.10 for dataset's protection.

This clause is divided into the three main aspects of protection: encryption, privacy rules and integrity (in all cases, of `gen_info` and payload structures). For the first and third aspects, the sub-clauses are further divided into the cases concerning containers and headers.

7.2 Encryption of elements in the format structure and genomic data

This sub-clause specifies the details on how the encryption parameters are conveyed in the protection of the `gen_info` elements specified in Part 1 of this Standard. The *protection* `gen_info` structure conveys the information on how its sibling boxes and the protection boxes of a layer below are encrypted. This information is represented with a list of *xenc:EncryptedData*, as specified in [6]. The data reference element of the XML Encryption tag (*xenc:EncryptedData*) uses the same set of resources identifiers. These references are constructed using the URI syntax described in section 7.4.1 of this Part of the Standard. If an element is encrypted, then it has to be listed with its corresponding *xenc:EncryptedData* element, and the payload of its box is replaced by the corresponding ciphertext (obtained applying the steps described in the *xenc:EncryptedData* element) prepended with the Initialization Vector (IV) used. The `gen_info` identifier and length cannot be encrypted, but the length has to be corrected to take into consideration any size variation between plaintext and ciphertext plus IV.

In the case of encrypting blocks (which are not gen_info structures), there are two types of URI, one per blocks representation mode (DCS or AUC).

In the case of AUC, the URI specifies three parameters (type, id_start, id_last). The blocks of the Access unit of the type indicated, with an Id greater or equal than id_start and smaller than id_last, are concatenated and encrypted according to the description provided in the *xenc:EncryptedData*. The resulting cipher is stored at the original location of the blocks. If the encryption method chosen requires to store an IV, the IV is prepended to the cipher of theblocks.

In the case of DSC, all blocks are concatenated, and encrypted according to the description provided in the *xenc:EncryptedData*. The resulting cipher is stored at the original location of the blocks. If the encryption method chosen requires to store an IV, the IV is prepended to the cipher of the blocks.

7.3 Privacy Rules for the use of the genomic information

In the *protection* structure, the privacy rules tag has to be a valid policy element specified according to the XACML specification [7]. Exporting this tag as the root element of a new document, a valid policy document is obtained.

At each of the possible levels (Dataset Group, Dataset, and Access unit), the privacy rules specify who can execute a given action and under which conditions. The set of possible actions at each level corresponds to the ones defined in Table 25.

Actions are referred in privacy policies by using the name defined in the different sub-clauses of clause 9, APIs to MPEG-G data. Each action is described as an attributeValue with category "urn:oasis:names:tc:xacml:3.0:attribute-category:action" and attributeId "urn:oasis:names:tc:xacml:1.0:action:action-id".

The resources for which the rule apply are determined from the definition of the operation in clause 9, and where the rule is stored. For example, a rule governing updateHeaderField stored in the first Dataset, is understood to apply to the header of the first Dataset, and cannot be used to determine the right to execute other actions on the header or any other elements of the first Dataset, or any element of any other Dataset.

Granularity is achieved by limiting the range of accepted input parameters. This is represented as additional attributes of the action. The attribute ids and types are the same as those used to define the action in clause 9. It is up to the rules to define what decision to take if some filtered parameter is not provided, or a parameter which is not filtered is provided.

For those operations in the API that use other operations present in the API to return the corresponding information, privacy rules must be checked in the internal calls. For example, if a user requests access to a Dataset Group with the getDataDatasetGroup operation, getDataDataset operation must be called in order to access the different Datasets inside it. If the information to be accessed is governed by privacy rules (either in the Dataset Group or in the different Datasets), authorization mechanism has to be called prior to accessing any information in order to guarantee that access is granted. This means that at least a privacy rule has to "Permit" access to the requested information to the user, granting the access to the required information.

In the case of internal calls, the request is translated to match the internal operation to be performed (i.e. the action-id is modified, for example from getDataDataset to getDataAccessUnit). Additionaly, a new attribute is added to the request: permitedByCaller is added and set to true. This attribute can only be generated within an internal call. Any other origin is prohibited. This attribute can be used to define default behaviours: if allowed at the dataset level, allow at the access unit level.

The following pseudocode shows how to perform a getDataDataset:

```
getDataDataset(parameters){
     request = parameters.translateToRequestForGetDataDataset()
     dataset = getDataset(parameters)
     isAuthorized = dataset.authorize(request)
     if(!isAuthorized){
          throw notAuthorized
     }
     data = empty
     for(AccessUnit accessUnit : dataset.accessUnits){
          requestAU = request.translateToRequestForGetAccessUnit()
          requestAU.add(permitedByCaller)
          isAuthorized = accessUnit.authorize(requestAU)
          if(isAuthorized){
               data.append(accessUnit.decode())
          }
     }
}
```

As the rules might need a permitedByCaller attribute to work properly, in case of requesting access to an operation used in the definition of another operation, and being denied the access, the

authorization process needs to start from the top and verify if with `permitedByCaller` the action is permited. An example follows for the case of a denied getDataAccessUnit. The pseudocode tests if for the parent DatasetGroup the operation is permited, if yes then for the parent Dataset, and finally for the AccessUnit, taking into account the *permitedByCaller* parameter.

```
if(accessUnit.authorize(request)==FALSE){
     datasetGroup = accessUnit.getDatasetGroup()
     requestDG = request.translateForGetDataDatasetGroup()
     if (datasetGroup.authorize(requestDG) == FALSE){
          return FALSE
     }
     dataset = accessUnit.getDataset()
     requestDT = requestDG.translateForGetDataDataset()
     requestDT.add(permitedByCaller)
     if (dataset.authorize(requestDT) == FALSE){
          return FALSE
     }
     requestAU = requestDT.translateForGetDataset()
     requestAU.add(permitedByCaller)
     return accessUnit.authorize(requestAU)
}
```

Annex A.11 contains a privacy rule with an example of authorization request.

7.4 Digital Signature of `gen_info` structure

7.4.1 General case

At each level, the protection container may include authentication information in the form of a digital signature. This includes signing a subset of the `gen_info` elements listed in Table 8.

Table 7: Definition of `gen_info` elements

Protection `gen_info` at hierarchy level	Can sign the content of
Dataset group (`dgcn`)	• Dataset group header (`dghd`) • Reference genome (`rfgn`) • Dataset group's metadata (`dgmd`) • All Dataset protection within the dataset group (`dtpr`)
Dataset (`dtcn`)	• Dataset header (`dthd`) • Master index table (`mitb`) • Parameters sets (`pars`) • Dataset's metadata (`dtmd`) • Descriptors stream protection within the dataset (`dspr`) • Access unit protection (`aupr`) • Access units' blocks
Descriptors stream (`dscn`)	• Descriptors stream header (`dshd`) • Descriptors stream's metadata (`dsmd`) • Descriptors stream's blocks

Access units (aucn)	• Access unit header (auhd) • Access unit information (auin)

Each signature is provided as an XML detached signature [8]. No canonicalization of the data is performed: the input of the authentication algorithm is the byte stream as stored on the storage medium following the standard [9]. The reference URI's are constructed as described in Table 9.

Table 8: URI construction

Protection box at hierarchy level	Content to point to	URI construction:
Dataset group (dgcn)	dghd	<file URI>/datasetgroup/{id}/header
	rfgn	<file URI>/datasetgroup/{id}/refgen
	dgmd	<file URI>/datasetgroup/{id}/metadata
	dtpr	<file URI>/datasetgroup/{id}/dataset/{d_id}/protection
Dataset (dtcn)	dthd	<file URI>/datasetgroup/{id}/dataset/{d_id}/header
	mitb	<file URI>/datasetgroup/{id}/dataset/{d_id}/mitb
	pars	<file URI>/datasetgroup/{id}/dataset/{d_id}/pars
	dtmd	<file URI>/datasetgroup/{id}/dataset/{d_id}/metadata
	dspr	<dataset URI>/destream/{id}/protection
	aupr	<dataset URI>/aunit/{id}/protection
	(access units' Blocks) for AUC mode	<dataset URI>/blocks/{type AU}/{id_start}/{id_end}
Descriptors stream (dscn)	dshd	<dataset URI>/destream/{id}/header
	dsmd	<dataset URI>/destream/{id}/metadata
	(descriptors streams' blocks) for DSC mode	<dataset URI>/destream/{id}/blocks
Access unit (aucn)	auhd	<dataset URI>/aunit/{id}/header
	auin	<dataset URI>/aunit/{id}/info

The content to sign for each box corresponds to the payload in each `gen_info` structure, without including the Key and the Length.

In the case of referencing blocks (which are not gen_info structures), two types of URIs are specified depending on the dataset mode (either AUC or DSC).

In the case of AUC mode, three parameters (type, id_start, id_last) are specified. The blocks of the Access units of the type indicated, within an Id greater or equal than id_start, and smaller than id_last are concatenated and signed. The resulting signature is stored in the signature element. In case where some of the blocks were encrypted, the cipher text is used.

In the case of DSC mode, all blocks of the specified descriptor stream are concatenated and signed. If the blocks are encrypted, the cipher text is used.

There are no requirements on which boxes to sign and each element can be signed multiple times.

7.4.2 Authenticity of the dataset group protection `gen_info`

Optionally, an enveloped signature can be provided, which will be located within the protection tag of the dataset group `gen_info`, such that the rest of the Protection tag is authenticated.

8 SAM interoperability

This clause aims at providing backward compatibility with the SAM format specification da805be [10].

In this specification a Key, Length, Value format is used for the data structures defined in this document.

8.1 SAM Header

The information contained in a SAM file header can be encoded in the DT_metadata `gen_info` structure defined in Part 1 of this standard. To store this information we define a new extension for metadata. Clauses 8.1.1-8.1.5 summarizes the fields needed. The extension can be found at clause 8.1.6.

8.1.1 HD field

Table 9: HD Field definition

Type	Description	SAM header tag
`char[Length]`	Format version. Accepted format: /^[0-9]+\.[0-9]+$/.	`VN`
`uint8`	Sorting order of alignments. Valid values: • 0x00: unknown (default), • 0x01: unsorted, • 0x02: queryname, • 0x03: coordinate. For coordinate sort, the major sort key is the RNAME field, with order defined by the order of @SQ lines in the header. The minor sort key is the POS field. For alignments with equal RNAME and POS, order is arbitrary. All alignments with `*' in RNAME field follow alignments with some other value but otherwise are in arbitrary order.	`SO`
`uint8`	Grouping of alignments, indicating that similar alignment records are grouped together but the file is not necessarily sorted overall. Valid values: • 0x00: none (default), • 0x01: query (alignments are grouped by QNAME), • 0x02: reference (alignments are grouped by RNAME/POS).	`GO`

8.1.2 SQ section

SN tag

The SN tag is replaced by the Ref_ID field in the Dataset Header.

LN tag

When transcoding from SAM to this standard, the LN tag values shall be used to validate the provided references to be encoded in the Dataset Header.

When transcoding from this standard to SAM, the value of the LN tags shall be calculated from the retrieved reference.

AS tag

This is encoded in the Reference_genome field of the Reference Genome `gen_info` defined in Part 1 of this standard.

M5 tag

When transcoding from SAM to this standard, the value of the MD5 checksum shall be replaced with the SHA256 checksum as defined in Part 1 of this standard.

When transcoding from this standard to SAM, the MD5 checksum shall be re-calculated.

UR tag

The URI of the sequence shall be encoded in the Ref_URI field of the Reference Genome `gen_info` defined in Part 1 of this standard.

SP tag

Table 10: SP tag description

Type	Description	SAM header tag
`char[Length]`	Species	`SP`

8.1.3 Read Group (RG)

Table 11: Read Group definition

Type	Description	SAM header tag
`char[Length]`	Read group identifier. Each read group must have a unique identifier. The value of this field is used in the 0x001c auxiliary field of alignment records. Must be unique among all read groups in header section. These fields may be modified when merging SAM files in order to handle collisions.	`RG-ID`
`char[Length]`	Name of sequencing center producing the read.	`CN`
`char[Length]`	Description	`DN`

char[Length]	Date the run was produced (ISO8601 date or date/time).	DT
char[Length]	Flow order. The array of nucleotide bases that correspond to the nucleotides used for each flow of each read. Multi-base flows are encoded in IUPAC format, and non-nucleotide flows by various other characters. Format: /*|[ACMGRSVTWYHKDBN]+/	FO
char[Length]	The array of nucleotide bases that correspond to the key sequence of each read.	KS
char[Length]	Library	LB
char[Length]	Programs used for processing the read group.	PG
uint32	Predicted median insert size.	PI
char[Length]	Platform/technology used to produce the reads. Valid values: CAPILLARY, LS454, ILLUMINA, SOLID, HELICOS, IONTORRENT, ONT, and PACBIO.	PL
char[Length]	Platform model. Free-form text providing further details of the platform/technology used.	PM
char[Length]	Platform unit (e.g. flowcell-barcode.lane for Illumina or slide for SOLiD). Unique identifier.	PU
char[Length]	Sample. Use pool name where a pool is being sequenced.	SM

8.1.4 Program Records (PG)

Table 12: Program Records

Type	Description	SAM header tag
char[Length]	Program record identifier. The value of this identifier is used in the alignment 0x001e field and 0x14 fields of other program records. Program record identifiers may be modified when merging SAM files in order to handle collisions.	PG-ID
char[Length]	Program Name	PN
char[Length]	Command Line	CL
char[Length]	Previous program record identifier. Must match another 0x11 field. Program records may be chained using 0x14 fields, with the last record in the chain having no 0x14 field. This chain defines the order of programs that have been applied to the alignment. Values of the 0x14 field may be modified when merging SAM files in order to handle collisions of Program record identifiers. The first Program Record in a chain (i.e. the one referred to by the PG tag in a SAM record) describes the most recent program that	PP

	operated on the SAM record. The next program record in the chain describes the next most recent program that operated on the SAM record. The Program record identifier on a SAM record is not required to refer to the newest program record in a chain. It may refer to any program record in a chain, implying that the SAM record has been operated on by the program in that PG record, and the program(s) referred to via the 0x14 field.	
`char[Length]`	Description	`DS`
`char[Length]`	Program version	`VN`

8.1.5 Comments (CO)

Table 13: Comments Field Definition

Type	Description	SAM header tag
`char[Length]`	Text comment.	`CO`

8.1.6 SAM interoperability extension

In the SAM specification, the header of the file defines fields for metadata information. In order to make the round trip possible (from SAM to MPEG-G and back to SAM), the following extension defines a mechanism for MPEG-G to contain the information present in the SAM header, in order to populate the same section in case of exporting the content back to the SAM format.

Table 14: SAM interoperability extension

Field name		Field type	Mandatory
SAM_format_version		String /^[0-9]+\.[0-9]+$/	No
sorting_order		Restricted vocabulary: unknown, unsorted, queryname, coordinate	No
grouping alignment		Restricted vocabulary: none, query, reference	No
Species		String	No
read_group (multiple allowed)			No
	read_group_identifier	String (uniqueness constraint)	Yes
	sequencing_center	String	No
	Description	String	No
	date_run	Date/Time	No

	flow_order	String /*|[ACMGRSVTWYHKDBN]+/	No
	key_sequence	String	No
	Library	String	No
	programs	String	No
	predicted_median_insert_size	Integer	No
	Platform	Restrited values: CAPILLARY, LS454, ILLUMINA, SOLID, HELICOS, IONTORRENT, ONT, and PACBIO	No
	platform_model	String	No
	platform unit	String	No
	Sample	String	No
programs (multiple allowed)			No
	program:identifier	String	Yes
	program name	String	No
	command_line	String	No
	previous_program	String (value must be listed in some program_identifier field)	No
	description	String	No
	program_version	String	No
Comments		String	No

8.2 Auxiliary fields mapping

This section aims at providing backward compatibility with the specification of the optional fields in the alignment section of the SAM format specification. The information is stored in the `AU_info` `gen_info` element defined in Part 1, as a sequence of `gen_aux` structures as defined below. The `read_identifier` used in the `gen_aux` corresponds to the `read_name`. The `gen_tag` associate for each segment associated to a `read_identifier` the auxiliary fields. The information is stored in the order of appearance of the segments in the Access Unit.

```
struct gen_aux
{
      string     read_identifier;
      uint8      segments;
      gen_tag[]  aux_fields[segments];
}
```

The array of `gen_tag` elements specify which auxiliary fields apply to the read segment specified by the `read_identifier`, segment tuple. The possible values for the key element are listed in Clause 8.2.1.

```
struct gen_tag
{
      char        Key[2];
      uint64      Length;
      uint8       Value[];
}
```

The fields in bold encode information already encoded in Part 2 of this standard. If the information associated to the field does not match the one encoded according to Part 2, priority should be given to the latter.

Key values from 0x0000 to 0x03ff are reserved to fields corresponding to the tags defined in the SAM specification, while values from 0x0400 to 0xffff are reserved for user defined fields.

Table 15: Key values range

Key values range	Scope
0x0000 – 0x03ff	Reserved for SAM tags
0x0400 – 0xffff	User defined fields

8.2.1 SAM auxiliary fields

This sections lists the elements to be used to support SAM auxiliary fields.

Table 16: Keys for SAM auxiliary fields

Key	Type	Description
`AM`	uint8	The smallest template-independent mapping quality of segments in the rest.
`AS`	uint8	Alignment score generated by the aligner
`BQ`	char[Length]	Offset to base alignment quality (BAQ), of the same length as the read sequence. At the i-th read base, $BAQ_i = Q_i - (BQ_i - 64)$ where Qi is the i-th base quality.
`BD`	char[Length]	Indels quality scores
`BI`	char[Length]	Indels quality scores
`E2`	char[Length]	The 2nd most likely base calls. Same encoding and same length as QUAL.
`FS`	char[Length]	Segment suffix. It identifies different readouts from the same template, e.g. if the read was read out from the forward or reverse

		strand.
PQ	uint8	Phred likelihood of the template, conditional on both mappings being correct.
SM	uint8	Template independent mapping quality
U2	char[Length]	Phred probability of the 2nd call being wrong conditional on the best being wrong. The same encoding as the quality values.
UQ	uint8	Phred likelihood of the segment, conditional on the mapping being correct.
LB	char[Length]	The library from which the read has been sequenced. *If the DT_Metadata structure contains a list of Libraries, this field must match one of the Libraries present in the DT_metadata structure as defined in section 8.1.3 of this document.*
PG	char[Length]	Value matches the header PG-ID tag if @PG is present.
PU	char[Length]	The platform unit in which the read was sequenced. If @RG headers are present, then platform unit must match the RG-PU field of one of the headers.
CO	char[Length]	Free-text comments
BC	char[Length]	Barcode sequence, with any quality scores stored in the 0x0022 field.
QT	char[Length]	Phred quality of the barcode sequence in the 0x0021 (or 0x0023) tag. Same encoding as the quality values.
RT	char[Length]	Deprecated alternative to 0x0021 field originally used at Sanger.
OC	char[Length]	Original CIGAR string, usually before realignment.
OP	uint64	Original mapping position, usually before realignment.
OQ	char[Length]	Original base quality, usually before recalibration.
CT	char[Length]	*strand ;type (;key (=value))** Complete read annotation tag, used for consensus annotation dummy features. The CT tag is intended primarily for annotation dummy reads, and consists of a strand, type and zero or more key=value pairs, each separated with semicolons. The strand field has four values as in GFF3 (GenericFeature Format v3) [11] and supplements FLAG bit 0x10 to allow unstranded (`.'), and stranded but unknown strand (`?') annotation. For these and annotation on the forward strand (strand set to `+'), do not set FLAG bit 0x10. For annotation on the reverse strand, set the strand to `-' and set FLAG bit 0x10. The type and any keys and their optional values are all percent

		encoded according to RFC3986 to escape meta-characters `=', `%', `;', `|' or non-printable characters not matched by the isprint() macro (with the C locale). For example a percent sign becomes `%2C'.
PT	char[Length]	*start ;end ;strand ;type (;key (=value))*(|start ;end ;strand ;type (;key (=value))*)** Read annotations for parts of the padded read sequence. This field value has the format of a series of tags separated by `|', each annotating a sub-region of the read. Each tag consists of start, end, strand, type and zero or more key=value pairs, each separated with semicolons. Start and end are 1-based positions between one and the sum of the M/I/D/P/S/=/X CIGAR operators, i.e. sequence length plus any pads. Note any editing of the CIGAR string may require updating this field coordinates, or even invalidate them. As in GFF3, strand is one of `+' for forward strand tags, `-' for reverse strand, `.' for unstranded or `?' for stranded but unknown strand. The type and any keys and their optional values are all percent encoded as in the 0x0027 field.
FZ	uint16[Length/2]	Flow signal intensities on the original strand of the read, stored as (uint16) round(value * 100.0).
CM	uint32	Edit distance between the color sequence and the color reference (see also 0x0013)
CS	char[Length]	Color read sequence on the original strand of the read. The primer base must be included.
CQ	char[Length]	Color read quality on the original strand of the read. Same encoding as the quality values; same length as 0x002b.

8.2.2 User defined fields

The key values in the range 0x0100 – 0xffff can be used for user-defined fields such as those defined in the SAM specification as tags starting with 'X', 'Y', 'Z'. [Update according to previous changes]

8.3 Transcoding to/from SAM

8.3.1 MPEG-G record and SAM record

Table 18 shows how SAM record fields are supported by fields in the MPEG-G record specified in ISO/IEC 23092-2.

Table 17 – Mapping between SAM record fields and MPEG-G record fields

SAM record field	MPEG-G record field
QNAME	read_name
FLAG	flags

RNAME	seq_ID
POS	mapping_pos
MAPQ	mapping_score
CIGAR	ecigar_string
RNEXT	seq_ID in pairs coded in the same record split_seq_ID in split pairs
PNEXT	mapping_pos in pairs coded in the same record split_pos in split pairs
TLEN	read_len
SEQ	sequence
QUAL	quality_values

8.3.2 SAM ambiguities resolution

This section describes how transcoding, to/from SAM, genomic data compliant with Part 2 of this standard shall be performed when a unique mapping in both directions is not possible due to ambiguities of the SAM file.

8.3.2.1 SAM Flags

This section contains a list of wrong SAM flags configuration that express alignment characteristics that cannot be associated at the same time to one mapped read or read pair.

The values of SAM flags according to the SAM specification this document refers to are reported in Table 19.

Table 18: SAM flags values

Int value	Hex value	Description
1	0x1	template having multiple segments in sequencing
2	0x2	each segment properly aligned according to the aligner
4	0x4	segment unmapped
8	0x8	next segment in the template unmapped
16	0x10	SEQ being reverse complemented
32	0x20	SEQ of the next segment in the template being reverse complemented

64	0x40	the first segment in the template
128	0x80	the last segment in the template
256	0x100	secondary alignment
512	0x200	not passing filters, such as platform/vendor quality controls
1024	0x400	PCR or optical duplicate
2048	0x800	supplementary alignment

8.3.2.1.1 Flag 151

The value 151 for the SAM flags corresponds to the case where the read is supposed to be at the same time mapped in a proper pair AND unmapped.

In this case the other SAM fields providing information on the read mapping (POS, MPOS, RNAME, CIGAR, RNEXT) shall be parsed to evaluate if they are concordant and represent a properly mapped read.

If the alignment information is consistent with the read sequence the 0x4 flag shall be ignored.

This choice is justified by the fact that one single flags is contradicting several SAM fields describing consistently a proper mapping and can therefore be supposed to have been wrongly generated by the aligner.

8.3.2.1.2 Unmapped mate with PNEXT value

A SAM record with the 0x8 flag set (next segment unmapped) may present a valid value for the PNEXT field. In case the SAM record containing the next segment (read pair) contains an unmapped read (flag 0x4 set and no mapping information) the PNEXT value in the first record should be discarded and the correct transcoding to SAM from MPEG shall set PNEXT = 0.

8.3.2.2 Duplicate records

In some SAM/BAM files identical records may appear as in the example shown below. The two records are identical and refer to a mate that is present only as a single record. The data are therefore inconsistent as they do not represent two separate pairs but one pair with one duplicate read.

In this case one of the replicated reads shall be discarded.

```
HSQ1004:134:C0D8DACXX:4:1305:12191:72218        99      chr1    247278658       60      101M    =       247278690       133
TCGGGGGAAGCCCAGGGATCTCTGTCACTGGGATCTCTGTCAGTGAGACAGTCAACTGTGATGCAGGCACCCCAGGGGGCCAGAGGCCAGGACAGCAGTGG
CCCFFFFFHHGHHJJJJJJJJJIJJJJJJJJJJJJJJJJJJJIJHHHHEHFFFFFFEDEDDEDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD>   RG:Z:NA12878
XT:A:U  NM:i:1  SM:i:37 AM:i:37 X0:i:1  X1:i:0  XM:i:1  XO:i:0  XG:i:0  MD:Z:49C51

HSQ1004:134:C0D8DACXX:4:1305:12191:72218        99      chr1    247278658       60      101M    =       247278690       133
TCGGGGGAAGCCCAGGGATCTCTGTCACTGGGATCTCTGTCAGTGAGACAGTCAACTGTGATGCAGGCACCCCAGGGGGCCAGAGGCCAGGACAGCAGTGG
CCCFFFFFHHGHHJJJJJJJJJIJJJJJJJJJJJJJJJJJJJIJHHHHEHFFFFFFEDEDDEDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD>   RG:Z:NA12878
XT:A:U  NM:i:1  SM:i:37 AM:i:37 X0:i:1  X1:i:0  XM:i:1  XO:i:0  XG:i:0  MD:Z:49C51

HSQ1004:134:C0D8DACXX:1:2306:12242:78140        1107    chr1    247278690       60      101M    =       247278658       -
```



```
133     ATCTCTGTCAGTGAGACAGTCAACTGTGATGCAGGCACCCCAGGGGGCCAGAGGCCAGGACAGCAGTGGATCCTGGGATAGGATGAGAATTATTTTGGCTG
:CCC@::(CCDDCCCCDDDCC>@CA>@EDEEFEDB=;HGGJHJIIIGGGHHGIJJJJJIIIIJJJJJJJJJIJJJJIJJJJJJJJJJHHHGGFFFFF@CC    RG:Z:NA12878
XT:A:U  NM:i:1  SM:i:37 AM:i:37 X0:i:1  X1:i:0  XM:i:1  XO:i:0  XG:i:0  MD:Z:17C83

HSQ1004:134:C0D8DACXX:4:1305:12191:72218        147     chr1    247278690       60      101M    =       247278658       -
133     ATCTCTGTCAGTGAGACAGTCAACTGTGATGCAGGCACCCCAGGGGGCCAGAGGCCAGGACAGCAGTGGATCCTGGGATAGGATGAGAATTATTTTGGCTG
DDDDDDDDDDDDDDDDCDDDDDDDDDEEEEEEFFFFEBJIHIIJJJJJJHHGJJJJJJJJIIJJIJJIJJJJJJJJJJJJJJJJJJJJJHHHHHFFFFFCCC    RG:Z:NA12878
XT:A:U  NM:i:1  SM:i:37 AM:i:37 X0:i:1  X1:i:0  XM:i:1  XO:i:0  XG:i:0  MD:Z:17C83
```

8.3.2.3 SAM headers errors

In some cases the names of sequences in the SAM file header don't match the reads. In the example below, the SAM file header says that the reads use a reference sequence named "chr11_gl000202_random". Note that the third fields for the two SAM records shown below have respectively the values "chr11" and "chr12". When transcoded to MPEG-G records, priority will be given to the information contained in the SAM record, if no reference labelled with the value carried by the third field of the SAM record (i.e. "chr11" and "chr12" in the example) is found, then the transcoder shall generate an error and skip the SAM record as corrupted.

A transcoding tool from SAM to MPEG-G cannot fix any SAM inconsistency found in a SAM record. Tools exists to try to do this in the SAM content itself before transcoding can take place such as for example Picard ValidateSamFile (http://broadinstitute.github.io/picard/command-line-overview.html#ValidateSamFile).

```
MICHAELJACKSON_0007:5:110:10401:1393#0          89      chr11   134801779       50      76M     *       0       0
TCCTGCTTTAGAAATCCAGAAATTGGGAGGCCGAGGCAGGTAGATCATGAGGTCAGGAGATCAAGACCATCCTGGC
HEGEEEB>CGFGGGBDGG2EGE@DD@GGEBGBHHHDHHHHGHEHHHHFHHHG@GGBHHDGHGGGGGEHHGHHHHHH  AS:i:0  XN:i:0  XM:i:0  XO:i:0
XG:i:0  NM:i:0  MD:Z:76 YT:Z:UU NH:i:1  XS:A:+

MICHAELJACKSON_0007:5:42:9610:3853#0         97      chr12   60101   50      76M     =       60355   330
GTCCATTCCCTAGAAGGCTGGCTGCCCCTGGGGATGTTTTGCACCAAGCCACTGTCTCCAGCTGGGGACTAGCATC
HHHFHHHHHHGHHHHGHH>HHHBHHHHHDHDGGCEHCHDHFFF@FFHDEEGGCEDEDDCA@@@@B@@>B<AAAA>>  AS:i:0  XN:i:0  XM:i:0  XO:i:0
XG:i:0  NM:i:0  MD:Z:76 YT:Z:UU NH:i:1  XS:A:-
```

8.3.2.4 Mapping position error

Experiment ERX276880 on the ENA repository (https://www.ebi.ac.uk/ena/data/view/ERX276880) contains IonTorrent sequence data where the following record can be found.

```
XOH00:00970:02945       0       MT      12017   19      72M     *       0       0
      CACCCACCACATTAACAACATAAAACCCTCATTCACACGAGAAAACACCCTCATGTTCATACACCTATCCCC        ABA@=774654<4-
44,4:5444(3=2::><:66545=<5555(4::>474744<9<>>=:==8200000&       XA:Z:map2-1     MD:Z:72 XE:i:3  XF:i:1
 PG:Z:tmap       RG:Z:ID NM:i:0  AS:i:72 XS:i:60
```

According to the information contained in the fourth SAM field, the read is supposed to map to position 12016 (0-indexed) in the reads. However, in the available reference genome, the read actually maps perfectly to position 12015 (0-indexed). A transcoding tool from SAM to MPEG-G cannot fix any SAM inconsistency found in a SAM record. Tools exists to try to do this in the SAM content itself before transcoding can take place such as for example Picard ValidateSamFile (http://broadinstitute.github.io/picard/command-line-overview.html#ValidateSamFile).

8.3.2.5 Unmapped reads with reference and position values set

In the case the original SAM file had unmapped reads with a position or a reference set (despite being unmapped), these values are lost. In case of half mapped pairs, the conversion from MPEG-G to SAM might not respect the recommended practice of setting RNAME and POS to the values of the mapped one.

9 APIs to MPEG-G data

9.1 Introduction

This clause contains the Application Programming Interface (API) specification. It specifies the operations to be available in a tool implementing the API.

This API is specified in order to facilitate access to and manipulation of MPEG-G compliant genomic content and the fields it contains. This API may be implemented locally or remotely.

In case the information is protected, operations are only executed if the corresponding authorization is obtained, according to the privacy rules, as specified in Section 7 of the standard.

9.2 Structure of the API

The API specifies operations that affect MPEG-G data structures, as specified in ISO/IEC 23092-1. Operations are applied to data structures that are organized in hierarchy levels. In the context of this part of the Standard, a hierarchy level defines the scope of an operation. Hierarchy levels include File, Dataset Group, Dataset, Access Unit and Genomic record, as described in ISO/IEC 23092-1 and ISO/IEC 23092-2.

Table 20 shows the classification of the different kinds of operations defined in the API. Each operation category may contain different operations. Each operation is instantiated for some of all possible levels of genomic information.

Table 19: Operations classification

Category	Description
Access	Operations that return content to the requester.
Modification	Operations that change content as indicated by the requester.
Authorization	Operations that check that the user has permission to perform an operation.
Verification	Operations that check the integrity of some content indicated by the requester.
Conversion	Operations that convert some content from/to MPEG-G to/from other formats.
Beacon-like	Operations that provides MPEG-G content in the form of beacons (statistical, appearance, etc.).

Based on the classification in categories presented in Table 20, the operations of the API are organized in several groups. This facilitates interoperability with other existing API's, like the one defined by GA4GH [12]. There is one core group and several extension groups, as specified below.

The four groups are:

- Access (core). These operations allow getting content from one or more fields, listing content and searching for content. This is the core group of this specification.

- Access (extended). These operations allow getting content based on MPEG-G specific features.

- Modification. These operations modify the structure of the genomic information, including addition and deletion of existing content.

- Other, which contains the following types of operations:

    - Authorization. These operations check if a user is allowed to perform some operation over genomic information.

    - Verification. These operations check the integrity of some information indicated by the user.

    - Conversion. These operations perform conversions between MPEG-G and other existing formats.

    - Beacon-like. These operations provide MPEG-G content in the form of beacons (statistical, appearance, etc.).

Tables 21 to 24 identify and briefly describe the operations considered in the different categories. Specifically, Table 21 lists access core operations, Table 22 lists access extended operations, Table 23 lists modification operations and Table 24 lists the rest of operations, indicating which category they belong to.

Table 20: Access Core Operations

Operation name	Brief description
`GetData`	Returns the content of a hierarchy level in decoded form.
`GetHeader`	Returns the complete header of the corresponding hierarchy level.
`GetMetadata`	Returns the complete metadata elements set of the corresponding hierarchy level.
`GetMetadataField`	Returns the content of a specific metadata field of the corresponding hierarchy level.
`GetReference`	Returns the reference used in a dataset group.
`IsSetField`	Checks if a field has a value in the corresponding hierarchy level.
`SearchData`	Searches for some value inside the data contained in the corresponding hierarchy level.
`SearchMetadata`	Searches for some value inside the metadata contained in the corresponding hierarchy level.

Table 21: Access Extended Operations

Operation name	Brief description
GetByLabel	Returns the content referenced by a specific label inside the corresponding level in decoded form.
GetEncodedByLabel	Returns the content referenced by a specific label inside the corresponding level in MPEG-G encoded form.
GetEncodedData	Returns the content of a hierarchy level in MPEG-G encoded form.
GetProtection	Returns the content of the complete protection element of the corresponding hierarchy level.
ListData	Lists all data contained in the corresponding hierarchy level.
ListLabel	List labels inside the corresponding hierarchy level.
SearchLabel	Searches for some value inside labels in the corresponding hierarchy level.

Table 22 : Modification Operations

Operation name	Brief description
AddData	Adds new content to the corresponding hierarchy level in decoded form.
AddEncodedData	Adds new encoded content at the corresponding hierarchy level in MPEG-G encoded form.
AddLabel	Adds a new label at the corresponding hierarchy level.
AddMetadata	Adds a new metadata element at the corresponding hierarchy level.
AddMetadataField	Adds a new metadata field at the corresponding hierarchy level.
DeleteData	Deletes a complete data structure from the corresponding hierarchy level.
UpdateData	Updates the content for the corresponding hierarchy level in decoded form.
UpdateEncodedData	Updates the content for the corresponding hierarchy level in MPEG-G encoded form.
UpdateHeader	Updates the header of the corresponding hierarchy level.
UpdateLabel	Updates a label at the corresponding hierarchy level.

UpdateMetadata	Updates the metadata element of the corresponding hierarchy level.
UpdateMetadataField	Updates a metadata field in the corresponding hierarchy level.

Table 23: Other Operations

Category	Operation name	Brief description
Authorize	Authorize	Checks if it is possible to perform an operation over some information contained in the file, applying the privacy rules defined at the corresponding hierarchy level.
Verify	Verify	Checks the integrity of the corresponding hierarchy level.
Conversion	ConvertFrom	Converts genomic information from a specified format into MPEG-G.
Conversion	ConvertTo	Extracts information from a genomic information file and converts it to the specified format.
Beacon-like	Beacon	Allows performing remote questions in a beacon-like form.

Table 25 contains the mapping matrix between operations and hierarchy levels, indicating which operation is available at each hierarchy level.

Operations are presented in alphabetical order.

Table 24: Operations matrix

Hierarchy level / Operation	File	Dataset group	Dataset	Access Unit	Descriptor stream	Genomic record
AddData				x		
AddEncodedData	x	x	x	x		
AddHeaderField	x	x	x	x		
AddLabel		x				
AddMetadata		x	x			
AddMetadataField		x	x			
Authorize		x	x	x		
Beacon		x				
ConvertFrom		x	x			
ConvertTo		x	x			
GetByLabel		x				

GetData	x	x	x	x		x
GetEncodedByLabel		x	x			
GetEncodedData	x	x	x	x	x	x
GetHeader	x	x	x	x		
GetMetadata		x	x			
GetMetadataField		x	x			
GetProtection		x	x	x	x	
IsSetField		x	x			
ListData	x	x	x			
ListLabel		x	x			
SearchData	x	x	x			
SearchLabel		x	x			
SearchMetadata	x	x				
UpdateData				x		x
UpdateEncodedData	x	x	x	x		
UpdateHeader	x	x	x	x		
UpdateLabel		x				
UpdateMetadata		x	x			
UpdateMetadataField		x	x			
Verify		x	x	x		

9.3 Detailed specification of the API

9.3.1 Data types

For data representation the following data types are defined:

uint: Unsigned integer number.

float: Floating point number as defined in ISO/IEC 23092-2.

st(v): String, as defined in ISO/IEC 23092-1.

bool: Boolean value.

9.3.2 Global error codes

This subclause describes the global error codes that can be returned by any operation. They are described in Table 26. Some operations can also provide specific error codes that are described in subclause 9.3.3.

Table 25: Global error codes

Return error code	Error name	Description
0	Not authorized	
1	Integrity error	
2	Verification error	
3	Key not found	
4	Decryption error	
5	Resource not found	
6	Operation not implemented	
7	Operation not implemented due to lack of encoder	This error is an extension to the more general error 6, and informs the requester that the action could be performed if a variant of the call is used where no recoding is required.
8	Malformed input	
Any other value lower than 1000	System error	

9.3.3 Specific error codes

This subclause describes the specific error codes that can be returned by some operations. They are described in Table 27.

Operations subclauses indicate when one of these specific error codes is used, if any.

Table 26: Specific error codes

Return error code	Error name	Description
1000	Field not set, no default	The element does not exist. If the element exists but is empty, this error is not returned.
1001	Invalid value	The value of the returned element does not comply with the schema.
1002	Invalid element	The provided element does not comply with the schema.
2000	Provided data of incompatible type	
3000	Existing element	Provided data already exists where it has to be added
3001	Invalid genomic reference	The provided genomic reference is not compatible with the genomic information.

9.3.4 Access core operations

9.3.4.1 GetData operations

9.3.4.1.1 GetDataFile

Input parameters:

- st(v) file_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2, where all genomic data complies with the filter criteria.

9.3.4.1.2 GetDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)

- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2, where all genomic data complies with the filter criteria.

9.3.4.1.3 GetDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2, where all genomic data complies with the filter criteria.

9.3.4.1.4 GetDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- uint read_count

- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2, where all genomic data complies with the filter criteria.

9.3.4.1.5 GetDataRecord

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- uint record_index
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2, where all genomic data complies with the filter criteria.

9.3.4.2 GetHeader operations

9.3.4.2.1 GetHeaderFile

Input parameters:

- st(v) file_ID

Output parameters:

- Header of the identified file as defined in ISO/IEC 23092-1.

9.3.4.2.2 GetHeaderDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- Header of the identified dataset group as defined in ISO/IEC 23092-1.

9.3.4.2.3 GetHeaderDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

Output parameters:

- Header of the identified dataset as defined in ISO/IEC 23092-1.

9.3.4.2.4 GetHeaderAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint acces_unit_ID

Output parameters:

- Header of the identified access unit as defined in ISO/IEC 23092-1.

9.3.4.3 GetMetadata operations

9.3.4.3.1 GetMetadataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- Complete metadata of the identified dataset group as defined in ISO/IEC 23092-1.

9.3.4.3.2 GetMetadataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

Output parameters:

- Complete metadata of the identified dataset as defined in ISO/IEC 23092-1, including values inherited from dataset group. For fields defined both at dataset and dataset group levels, dataset level value prevails.

9.3.4.4 GetMetadataField operations

9.3.4.4.1 GetMetadataFieldDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) field_name

Output parameters:

- st(v) containing the value of the requested metadata field of the identified dataset group or in the definition of the extension, which has to be provided externally. If the value is not provided, the default value as specified in the XML schema definition is provided.

Specific errors:

- This operation can return specific errors 1000 and 1001.

9.3.4.4.2 GetMetadataFieldDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- st(v) field_name

Output parameters:

- st(v) containing the value of the requested metadata field of the identified dataset or in the definition of the extension, which has to be provided externally. If the field is not defined at the dataset level, dataset group level value of the metadata field is returned. If it is not provided at the dataset group level, the default value as specified in the XML schema definition is returned.

Specific errors:

- This operation can return specific errors 1000 and 1001.

9.3.4.5 GetReference operations

9.3.4.5.1 GetReferenceDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- Genomic reference used in the identified dataset group as defined in ISO/IEC 23092-1.

9.3.4.6 isSetField operations

9.3.4.6.1 isSetFieldDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) field_name

Output parameters:

- bool with true or false value depending on if the identified metadata field is present at the dataset group metadata.

Specific errors:

- This operation can return specific error 1000.

9.3.4.6.2 isSetFieldDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- st(v) field_name

Output parameters:

- bool with true or false value depending on if the identified metadata field is present at the dataset group metadata.

Specific errors:

- This operation can return specific error 1000.

9.3.4.7 SearchData operations

9.3.4.7.1 SearchDataFile

Input parameters:

- st(v) file_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A list of concatenated data structures containing {uint dataset_group_ID, uint dataset, uint reference_id, uint sequence_id, uint class_type, uint au_id, uint read_id} accomplishing the search criteria.

9.3.4.7.2 SearchDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A list of concatenated data structures containing {uint dataset, uint reference_id, uint sequence_id, uint class_type, uint au_id, uint read_id} accomplishing the search criteria.

9.3.4.7.3 SearchDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)

- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A list of concatenated data structures containing {uint reference_id, uint sequence_id, uint class_type, uint au_id, uint read_id} accomplishing the search criteria.

9.3.4.8 SearchMetadata operations

9.3.4.8.1 SearchMetadataFile

Input parameters:

- st(v) file_ID
- st(v) title
- st(v) type
- st(v) abstract
- st(v) project_centre_name
- st(v) description
- st(v) samples
- st(v) standard_extensions
- st(v) other_extensions

Output parameters:

- A list of concatenated pairs {uint dataset_group_ID, uint dataset} accomplishing the search criteria.

Note:

- Parameters sample, standard_extensions and other_extensions contain an XML document describing the information to be searched.

9.3.4.8.2 SearchMetadataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) title
- st(v) type
- st(v) abstract
- st(v) project_centre_name

- st(v) description
- st(v) samples
- st(v) standard_extensions
- st(v) other_extensions

Output parameters:

- A list of concatenated uint dataset_ID's accomplishing the search criteria.

Note:

- Parameters sample, standard_extensions and other_extensions contain an XML document describing the information to be searched.

9.3.5 Access extended operations

9.3.5.1 GetByLabel operations

9.3.5.1.1 GetByLabelDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) label

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2 that are inside a region identified by the label within the specified dataset group.

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- st(v) label

Output parameters:

- Concatenated sequence of MPEG-G records as defined in ISO/IEC 23092-2 that are inside a region identified by the label within the specified dataset.

9.3.5.2 GetEncodedByLabel operations

9.3.5.2.1 GetEncodedByLabelDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) label

Output parameters:

- A byte stream containing the information inside a region identified by the label within the dataset group.

Additional notes:

- In case of receiving an error due to the lack of an encoder (error 7), GetByLabelDatasetGroup can provide a similar functionality.

9.3.5.2.2 GetEncodedByLabelDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- st(v) label

Output parameters:

- A byte stream containing the information inside a region identified by the label within the dataset.

Additional notes:

- In case of receiving an error due to the lack of an encoder (error 7), GetByLabelDataset can provide a similar functionality.

9.3.5.3 GetEncodedData operations

9.3.5.3.1 GetEncodedDataFile

Input parameters:

- st(v) file_ID
- uint class_type
- uint read_count

- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A file, as defined in ISO/IEC 23092-1, where all genomic data complies with the filter criteria

9.3.4.1.3 GetEncodedDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A byte stream compliant with a dataset group container definition, as defined in ISO/IEC 23092-1, where all genomic data complies with the filter criteria

9.3.4.1.3 GetEncodedDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

- uint dataset_ID
- uint class_type
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A byte stream compliant with a dataset container definition, as defined in ISO/IEC 23092-1, where all genomic data complies with the filter criteria

9.3.4.1.3 GetEncodedDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A byte stream compliant with an access unit container definition, as defined in ISO/IEC 23092-1, where all genomic data complies with the filter criteria

9.3.4.1.3 GetEncodedDataStream

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint descriptor_ID
- uint read_count
- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A byte stream compliant with a stream container definition, as defined in ISO/IEC 23092-1, where all genomic data complies with the filter criteria

9.3.4.1.3 GetEncodedDataBlock

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint acces_unit_ID
- uint descriptor_ID
- uint read_count

- uint start_position (only if class type != U)
- uint end_position (only if class type != U)
- uint reference_sequence (only if class type != U)
- bool presence_of_multiple alignments (only if class type != U)
- float alignment_score (only if class_type != U)
- list of st(v) signatures (only if class_type = U)
- uint mismatch_threshold (only if class_type = M or class_type = N)

Output parameters:

- A byte stream compliant with a block container definition, as defined in ISO/IEC 23092-1, where all genomic data complies with the filter criteria

9.3.5.4 GetProtection operations

9.3.5.4.1 GetProtectionDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- A byte stream compliant with protection container definition at dataset group level as defined in ISO/IEC 23092-1.

9.3.5.4.2 GetProtectionDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

Output parameters:

- A byte stream compliant with protection container definition at dataset level as defined in ISO/IEC 23092-1.

9.3.5.4.3 GetProtectionAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

- uint dataset_ID
- uint class_type
- uint access_unit_ID

Output parameters:

- A byte stream compliant with protection container definition at access unit level as defined in ISO/IEC 23092-1.

9.3.5.4.4 GetProtectionStream

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint descriptor_ID

Output parameters:

- A byte stream compliant with protection container definition at descriptor stream level as defined in ISO/IEC 23092-1.

9.3.5.5 ListData operations

9.3.5.5.1 ListDataFile

Input parameters:

- st(v) file_ID

Output parameters:

- A concatenation of 32-bit integers encoded in big endian representing the dataset set group ID's contained in the file.

9.3.5.5.2 ListDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- A concatenation of 32-bit uint's encoded in big endian representing the dataset ID's contained in the dataset group.

9.3.5.5.3 ListDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type

Output parameters:

- A concatenation of 32-bit integers encoded in big endian representing the access unit ID's contained in the dataset.

9.3.5.6 ListLabel operations

9.3.5.6.1 ListLabelDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- A concatenation of null-terminated st(v) representing the labels contained in the dataset group.

9.3.5.6.2 ListLabelDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type

Output parameters:

- A concatenation of null-terminated st(v) representing the labels contained in the dataset.

9.3.5.7 SearchLabel operations

9.3.5.7.1 SearchLabelDatasetGroup

Input parameters:

- st(v) file_ID

- uint dataset_group_ID
- st(v) label_substring

Output parameters:

- A concatenation of null-terminated st(v) representing the labels contained in the dataset group that have part of the label_substring parameter in their name.

9.3.5.7.2 SearchLabelDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- st(v) label_subtring

Output parameters:

- A concatenation of null-terminated st(v) representing the labels contained in the dataset that have part of the label_substring parameter in their name.

9.3.6 Modification operations

9.3.6.1 AddData operations

9.3.6.1.1 AddDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint acces_unit_ID
- a concatenated sequence of MPEG-G record(s) as defined in ISO/IEC 23092-2.

Output parameters:

- None.

9.3.6.2 AddEncodedData operations

9.3.6.2.1 AddEncodedDataFile

Input parameters:

- st(v) file_ID
- a byte stream compliant with a dataset group container as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.2.2 AddEncodedDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- An optional genomic_reference compliant with a reference genome element as defined in ISO/IEC 23092-1.
- a byte stream compliant with a dataset container as defined in ISO/IEC 23092-1.

Output parameters:

- None.

Specific errors:

- This operation can return specific error Reference.1.

9.3.6.2.3 AddEncodedDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- a byte stream compliant with an access unit container as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.2.4 AddEncodedDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- a concatenated sequence of MPEG-G record(s) as defined in ISO/IEC 23092-2.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 2000.

9.3.6.3 AddLabel operations

9.3.6.3.1 AddLabelDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- A byte stream compliant with a label container as defined in in ISO/IEC 23092-1.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 3000.

9.3.6.4 AddMetadata operations

9.3.6.4.1 AddMetadataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- a byte stream compliant with a metadata container at dataset group level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 3000.

9.3.6.4.2 AddMetadataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- a byte stream compliant with an metadata container at dataset level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 3000.

9.3.6.5 AddMetadataField operations

9.3.6.5.1 AddMetadataFieldDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) metatadata_field
- st(v) metadata_value (optional)

Output parameters:

- None.

Specific errors:

- This operation can return specific errors 3000 and 1002.

9.3.6.5.2 AddMetadataFieldDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

- st(v) metatadata_field
- st(v) metadata_value (optional)

Output parameters:

- None.

Specific errors:

- This operation can return specific errors 3000 and 1002.

9.3.6.6 DeleteData operations

9.3.6.6.1 DeleteDataFile

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- None.

9.3.6.6.2 DeleteDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

Output parameters:

- None.

9.3.6.6.3 DeleteDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID

Output parameters:

- None.

9.3.6.6.4 DeleteDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- uint record_index

Output parameters:

- None.

9.3.6.7 UpdateData operations

9.3.6.7.1 UpdateDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint acces_unit_ID
- a concatenated sequence of MPEG-G record(s) as defined in ISO/IEC 23092-2.

Output parameters:

- None.

9.3.6.7.2 UpdateDataRecord

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type

- uint acces_unit_ID
- uint stream_ID
- an MPEG-G record as defined in ISO/IEC 23092-2.

Output parameters:

- None.

9.3.6.8 UpdateEncodedData operations

9.3.6.8.1 UpdateEncodedDataFile

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- a byte stream compliant with a dataset group container as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.8.2 UpdateEncodedDataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- a byte stream compliant with a dataset container as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.8.3 UpdateEncodedDataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID

- a byte stream compliant with an access unit container as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.8.4 UpdateEncodedDataAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- uint record_index
- a concatenated sequence of MPEG-G record(s) as defined in ISO/IEC 23092-2.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 2000.

9.3.6.9 UpdateHeader operations

9.3.6.9.1 UpdateHeaderFile

Input parameters:

- st(v) file_ID
- a byte stream compliant with a header container at file level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.9.2 UpdateHeaderDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- a byte stream compliant with a header container at dataset group level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.9.3 UpdateHeaderDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- a byte stream compliant with a header container at dataset level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.9.4 UpdateHeaderAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID
- a byte stream compliant with a header container at access unit level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

9.3.6.10 UpdateLabel operations

9.3.6.10.1 UpdateLabelDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) label

Output parameters:

- None.

9.3.6.11 UpdateMetadata operations

9.3.6.11.1 UpdateMetadataDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- a byte stream compliant with an metadata container at dataset group level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 3000.

9.3.6.11.2 UpdateMetadataDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- a byte stream compliant with an metadata container at dataset level as defined in ISO/IEC 23092-1.

Output parameters:

- None.

Specific errors:

- This operation can return specific error 3000.

9.3.6.12 UpdateMetadataField operations

9.3.6.12.1 UpdateMetadataFieldDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) metatadata_field
- st(v) metadata_value (optional)

Output parameters:

- None.

Specific errors:

- This operation can return specific error 1002.

9.3.6.12.2 UpdateMetadataFieldDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- st(v) metatadata_field
- st(v) metadata_value (optional)

Output parameters:

- None.

Specific errors:

- This operation can return specific errors 1002.

9.3.7 Authorization operations

9.3.7.1 AuthorizeDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- st(v) authorization_context

Output parameters:

- bool containing true or false value depending on if the action requested to be performed over the dataset group is authorized or not.

9.3.7.2 AuthorizeDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

- uint dataset_ID
- st(v) authorization_context

Output parameters:

- bool containing true or false value depending on if the action requested to be performed over the dataset is authorized or not.

9.3.7.3 AuthorizeAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint class_type
- uint access_unit_ID
- st(v) authorization_context

Output parameters:

- bool containing true or false value depending on if the action requested to be performed over the access unit is authorized or not.

9.3.8 Verification operations

9.3.8.1 VerifyDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- bool containing true or false value depending on if the dataset group integrity is verified or not.

9.3.8.2 VerifyDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

Output parameters:

- bool containing true or false value depending on if the dataset integrity is verified or not.

9.3.8.3 VerifyAccessUnit

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID
- uint class_type
- uint access_unit_ID

Output parameters:

- bool containing true or false value depending on if the access unit integrity is verified or not.

9.3.9 Conversion operations

9.3.9.1 ConvertFromDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID

Output parameters:

- SAM/BAM compliant file containing the information contained in the dataset group.

9.3.9.2 ConvertFromDataset

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_ID

Output parameters:

- SAM/BAM compliant file containing the information contained in the dataset.

9.3.9.3 ConvertToDatasetGroup

Input parameters:

- SAM/BAM compliant file containing genomic information.

Output parameters:

- A byte stream compliant with a dataset group container definition as defined in ISO/IEC 23092-1.

9.3.9.4 ConvertToDataset

Input parameters:

- SAM/BAM compliant file containing genomic information.

Output parameters:

- A byte stream compliant with a dataset container definition as defined in ISO/IEC 23092-1.

9.3.10 Beacon-like operations

9.3.10.1 BeaconDatasetGroup

Input parameters:

- st(v) file_ID
- uint dataset_group_ID
- uint dataset_list_ID
- uint position

Output parameters:

- a float value containing the ratio number reported mismatches at position / total number of reads aligned with the position.

Annex A
(informative)

XML Schemas and XML-based data

This annex describes XML Schemas corresponding to metadata information and protection elements. It also includes privacy rules and authorization requests.

A.1 Dataset group metadata dgmd XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema                    targetNamespace="urn:mpeg:mpeg-g:metadata:dataset_group:2017"
xmlns="http://www.w3.org/2001/XMLSchema"          xmlns:mpg-meta-data-gr="urn:mpeg:mpeg-
g:metadata:dataset_group:2017">

  <complexType name="ProjectCentreType">
    <sequence>
      <element name="ProjectCentreName" type="string"/>
      <element       name="Extensions"       type="mpg-meta-data-gr:ChildExtensionsType"
minOccurs="0" maxOccurs="1"/>
    </sequence>
  </complexType>

  <element name="DatasetGroup" type="mpg-meta-data-gr:DatasetGroupType"/>

  <complexType name="DatasetGroupType">
    <sequence>
      <element name="Title" type="string" minOccurs="1" maxOccurs="1"/>
      <element name="Type" type="string" minOccurs="1" maxOccurs="1"/>
      <element name="Abstract" type="string" minOccurs="0" maxOccurs="1"/>
      <element      name="ProjectCentre"      type="mpg-meta-data-gr:ProjectCentreType"
minOccurs="0" maxOccurs="1"/>
      <element name="Description" type="string" minOccurs="0" maxOccurs="1"/>
      <element name="Samples" type="mpg-meta-data-gr:SamplesType"/>
      <element name="Extensions" type="mpg-meta-data-gr:ExtensionsType"/>
    </sequence>
    <attribute name="profile" type="anyURI" use="optional"/>
  </complexType>

  <complexType name="SamplesType">
    <sequence>
      <element    name="Sample"    type="mpg-meta-data-gr:SampleType"    minOccurs="1"
maxOccurs="unbounded"/>
    </sequence>
  </complexType>

  <complexType name="SampleType">
    <sequence>
      <element name="TaxonId" type="int" minOccurs="1" maxOccurs="1"/>
      <element name="Title" type="string" minOccurs="0" maxOccurs="1"/>
      <element       name="Extensions"       type="mpg-meta-data-gr:ChildExtensionsType"
minOccurs="0" maxOccurs="1"/>
    </sequence>
  </complexType>
```

```
  <complexType name="ExtensionType">
    <sequence>
      <element name="Type" type="anyURI"/>
      <element name="Inheritable" type="boolean"/>
      <any minOccurs="1"/>
    </sequence>
  </complexType>

  <complexType name="ExtensionsType">
    <sequence>
      <element   name="Extension"   type="mpg-meta-data-gr:ExtensionType"   minOccurs="0"
maxOccurs="unbounded"/>
    </sequence>
  </complexType>

  <complexType name="ChildExtensionType">
    <sequence>
      <element name="Type" type="anyURI"/>
      <any minOccurs="1"/>
    </sequence>
  </complexType>

  <complexType name="ChildExtensionsType">
    <sequence>
      <element        name="Extension"        type="mpg-meta-data-gr:ChildExtensionType"
minOccurs="0" maxOccurs="unbounded"/>
    </sequence>
  </complexType>
</schema>
```

A.2 Dataset metadata `dtmd` XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema                          targetNamespace="urn:mpeg:mpeg-g:metadata:dataset:2017"
xmlns="http://www.w3.org/2001/XMLSchema"          xmlns:mpg-meta-data-gr="urn:mpeg:mpeg-
g:metadata:dataset_group:2017"
xmlns:mpg-meta-dataset="urn:mpeg:mpeg-g:metadata:dataset:2017">
<import                          namespace="urn:mpeg:mpeg-g:metadata:dataset_group:2017"
schemaLocation="DatasetGroupSchemaOneProfile.xsd"/>
    <complexType name="DatasetType">
    <sequence>

       <element name="Title" type="string" minOccurs="1"
              maxOccurs="1">
       </element>

       <element name="Type" type="string" minOccurs="1"
              maxOccurs="1">
       </element>

       <element name="Abstract" type="string" minOccurs="0"
              maxOccurs="1">
       </element>

       <element name="ProjectCentre"
              type="mpg-meta-data-gr:ProjectCentreType" minOccurs="0"
              maxOccurs="1">
       </element>
```



```
        <element name="Description" type="string" minOccurs="0"
                maxOccurs="1">
        </element>
        <element   name="Samples"   type="mpg-meta-data-gr:SamplesType"   minOccurs="1"
maxOccurs="1">
        </element>
        <element  name="Extentions"  type="mpg-meta-dataset:ExtensionsType"  maxOccurs="1"
minOccurs="0"></element>
    </sequence>
    <attribute name="profile" type="anyURI" use="optional"></attribute>
    </complexType>

    <element name="Dataset" type="mpg-meta-dataset:DatasetType"></element>

    <complexType name="ExtensionType">
    <sequence>
        <element name="Type" type="anyURI"></element>
        <any minOccurs="1"/>
    </sequence>
    </complexType>

    <complexType name="ExtensionsType">
    <sequence>
        <element  name="Extention"  type="mpg-meta-dataset:ExtensionType"  minOccurs="1"
maxOccurs="unbounded"></element>
    </sequence>
    </complexType>
</schema>
```

A.3 EGA sample extension XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema                          targetNamespace="urn:mpeg:mpeg-g:metadata:extension:ega"
xmlns="http://www.w3.org/2001/XMLSchema"                        xmlns:tns="urn:mpeg:mpeg-
g:metadata:extension:ega">
    <complexType name="linkType">
    <sequence>
        <element name="URI" type="anyURI" minOccurs="1" maxOccurs="1"/>
    </sequence>
    </complexType>

    <complexType name="linksType">
        <sequence>
              <element       name="link"       type="tns:linkType"       minOccurs="1"
maxOccurs="unbounded"/>
        </sequence>
    </complexType>

    <complexType name="EGA_SampleType">
    <sequence>
        <element name="scientific_name" type="string" minOccurs="0" maxOccurs="1"/>
        <element name="common_name" type="string" minOccurs="0" maxOccurs="1"/>
        <element name="anonymized_name" type="string" minOccurs="0" maxOccurs="1"/>
        <element name="individual_name" type="string" minOccurs="0" maxOccurs="1"/>
        <element name="description" type="string" minOccurs="0" maxOccurs="1"/>
        <element name="links" type="tns:linksType" minOccurs="0" maxOccurs="1"/>
    </sequence>
    </complexType>
```

```
    <element name="egaSample" type="tns:EGA_SampleType"></element>
</schema>
```

A.4 EGA experiment extension XML schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:mpeg:mpeg-g:metadata:extension:ega"
    elementFormDefault="qualified"
    xmlns="urn:mpeg:mpeg-g:metadata:extension:ega"
    xmlns:tns="urn:mpeg:mpeg-g:metadata:extension:ega"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:com="SRA.common">
  <xs:import    schemaLocation="ftp://ftp.sra.ebi.ac.uk/meta/xsd/sra_1_5/SRA.common.xsd"
namespace="SRA.common"/>

  <!-- STRING ENUMERATIONS BEGIN -->
  <xs:simpleType name="typeLibraryStrategy">
    <xs:annotation>
      <xs:documentation>Sequencing       technique       intended       for       this
library.</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="WGS">
        <xs:annotation>
          <xs:documentation>
            Whole Genome Sequencing - random sequencing of the whole genome (see pubmed
10731132 for details)
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="WGA">
        <xs:annotation>
          <xs:documentation>
            Whole  Genome  Amplification  followed  by  random  sequencing.  (see  pubmed
1631067,8962113 for details)
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="WXS">
        <xs:annotation>
          <xs:documentation>
            Random sequencing of exonic regions selected from the genome. (see pubmed
20111037 for details)
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RNA-Seq">
        <xs:annotation>
          <xs:documentation>
            Random sequencing of whole transcriptome, also known as Whole Transcriptome
Shotgun Sequencing, or WTSS). (see
            pubmed 18611170 for details)
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ssRNA-seq">
        <xs:annotation>
          <xs:documentation>
```



```
            Strand-specific RNA sequencing.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="miRNA-Seq">
        <xs:annotation>
          <xs:documentation>
            Micro RNA sequencing strategy designed to capture post-transcriptional RNA
elements and include non-coding
            functional elements. (see pubmed 21787409 for details)
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ncRNA-Seq">
        <xs:annotation>
          <xs:documentation>
            Capture of other non-coding RNA types, including post-translation
modification types such as snRNA (small
            nuclear RNA) or snoRNA (small nucleolar RNA), or expression regulation
types such as siRNA (small interfering RNA) or
            piRNA/piwi/RNA (piwi-interacting RNA).
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="FL-cDNA">
        <xs:annotation>
          <xs:documentation>    Full-length    sequencing    of    cDNA    templates
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="EST">
        <xs:annotation>
          <xs:documentation>    Single    pass    sequencing    of    cDNA    templates
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Hi-C">
        <xs:annotation>
          <xs:documentation> Chromosome Conformation Capture technique where a biotin-
labeled nucleotide is incorporated at the ligation junction, enabling selective
purification of chimeric DNA ligation junctions followed by deep sequencing.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ATAC-seq">
        <xs:annotation>
          <xs:documentation> Assay for Transposase-Accessible Chromatin (ATAC) strategy
is used to study genome-wide chromatin accessibility. alternative method to DNase-seq
that uses an engineered Tn5 transposase to cleave DNA and to integrate primer DNA
sequences into the cleaved genomic DNA. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="WCS">
        <xs:annotation>
          <xs:documentation> Random sequencing of a whole chromosome or other replicon
isolated from a genome. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RAD-Seq"/>
```

```
      <xs:enumeration value="CLONE">
        <xs:annotation>
          <xs:documentation>   Genomic   clone   based   (hierarchical)   sequencing.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="POOLCLONE">
        <xs:annotation>
          <xs:documentation> Shotgun of pooled clones (usually BACs and Fosmids).
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="AMPLICON">
        <xs:annotation>
          <xs:documentation>
            Sequencing of overlapping or distinct PCR or RT-PCR products. For example,
metagenomic community profiling
            using SSU rRNA .
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="CLONEEND">
        <xs:annotation>
          <xs:documentation>   Clone   end   (5',   3',   or   both)   sequencing.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="FINISHING">
        <xs:annotation>
          <xs:documentation> Sequencing intended to finish (close) gaps in existing
coverage. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ChIP-Seq">
        <xs:annotation>
          <xs:documentation> ChIP-seq, Chromatin ImmunoPrecipitation, reveals binding
sites of specific proteins, typically transcription factors (TFs) using antibodies to
extract DNA fragments bound to the target protein. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MNase-Seq">
        <xs:annotation>
          <xs:documentation> Identifies well-positioned nucleosomes. uses Micrococcal
Nuclease (MNase) is an endo-exonuclease that processively digests DNA until an
obstruction, such as a nucleosome, is reached. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="DNase-Hypersensitivity">
        <xs:annotation>
          <xs:documentation>
            Sequencing of hypersensitive sites, or segments of open chromatin that are
more readily cleaved by DNaseI.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Bisulfite-Seq">
        <xs:annotation>
          <xs:documentation>
            MethylC-seq. Sequencing following treatment of DNA with bisulfite to
```



```
convert cytosine residues to uracil
            depending on methylation status.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="CTS">
        <xs:annotation>
          <xs:documentation> Concatenated Tag Sequencing </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MRE-Seq">
        <xs:annotation>
          <xs:documentation>  Methylation-Sensitive  Restriction  Enzyme  Sequencing.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MeDIP-Seq">
        <xs:annotation>
          <xs:documentation>   Methylated   DNA   Immunoprecipitation   Sequencing.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MBD-Seq">
        <xs:annotation>
          <xs:documentation> Methyl CpG Binding Domain Sequencing. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Tn-Seq">
        <xs:annotation>
          <xs:documentation>
            Quantitatively determine fitness of bacterial genes based on how many times
a purposely seeded transposon gets
            inserted into each gene of a colony after some time.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="VALIDATION">
        <xs:annotation>
          <xs:documentation>CGHub  special  request:  Independent  experiment  to  re-
evaluate putative variants. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="FAIRE-seq">
        <xs:annotation>
          <xs:documentation>Formaldehyde  Assisted  Isolation  of  Regulatory  Elements.
Reveals regions of open chromatin. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="SELEX">
        <xs:annotation>
          <xs:documentation>Systematic  Evolution  of  Ligands  by  Exponential
enrichment</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RIP-Seq">
        <xs:annotation>
          <xs:documentation>Direct sequencing of RNA immunoprecipitates (includes CLIP-
Seq, HITS-CLIP and PAR-CLIP). </xs:documentation>
        </xs:annotation>
```

```
      </xs:enumeration>
      <xs:enumeration value="ChIA-PET">
        <xs:annotation>
          <xs:documentation>Direct sequencing of proximity-ligated chromatin
immunoprecipitates.</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Synthetic-Long-Read">
        <xs:annotation>
          <xs:documentation>binning and barcoding of large DNA fragments to facilitate
assembly of the fragment</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Targeted-Capture">
        <xs:annotation>
          <xs:documentation>Enrichment of a targeted subset of loci.</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Tethered Chromatin Conformation Capture"/>
      <xs:enumeration value="OTHER">
        <xs:annotation>
          <xs:documentation> Library strategy not listed. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="typeLibrarySource">
    <xs:annotation>
      <xs:documentation> The LIBRARY_SOURCE specifies the type of source material that
is being sequenced. </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="GENOMIC">
        <xs:annotation>
          <xs:documentation> Genomic DNA (includes PCR products from genomic DNA).
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="GENOMIC SINGLE CELL"/>
      <xs:enumeration value="TRANSCRIPTOMIC">
        <xs:annotation>
          <xs:documentation> Transcription products or non genomic DNA (EST, cDNA, RT-
PCR, screened libraries). </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="TRANSCRIPTOMIC SINGLE CELL"/>
      <xs:enumeration value="METAGENOMIC">
        <xs:annotation>
          <xs:documentation> Mixed material from metagenome. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="METATRANSCRIPTOMIC">
        <xs:annotation>
          <xs:documentation> Transcription products from community targets
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="SYNTHETIC">
```



```
        <xs:annotation>
          <xs:documentation> Synthetic DNA. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="VIRAL RNA">
        <xs:annotation>
          <xs:documentation> Viral RNA. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="OTHER">
        <xs:annotation>
          <xs:documentation> Other, unspecified, or unknown library source material.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="typeLibrarySelection">
    <xs:annotation>
      <xs:documentation> Method used to enrich the target in the sequence library
preparation </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="RANDOM">
        <xs:annotation>
          <xs:documentation>No Selection or Random selection</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="PCR">
        <xs:annotation>
          <xs:documentation>target enrichment via PCR</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RANDOM PCR">
        <xs:annotation>
          <xs:documentation>Source material was selected by randomly generated
primers.</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RT-PCR">
        <xs:annotation>
          <xs:documentation>target enrichment via </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="HMPR">
        <xs:annotation>
          <xs:documentation>Hypo-methylated partial restriction
digest</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MF">
        <xs:annotation>
          <xs:documentation>Methyl Filtrated</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="repeat fractionation">
        <xs:annotation>
          <xs:documentation>
```

```
            Selection for less repetitive (and more gene rich) sequence through Cot
filtration (CF) or other fractionation
            techniques based on DNA kinetics.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="size fractionation">
        <xs:annotation>
          <xs:documentation>  Physical  selection  of  size  appropriate  targets.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MSLL">
        <xs:annotation>
          <xs:documentation>Methylation Spanning Linking Library</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="cDNA">
        <xs:annotation>
          <xs:documentation>PolyA  selection  or  enrichment  for  messenger  RNA  (mRNA);
synonymize with PolyA </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="cDNA_randomPriming"/>
      <xs:enumeration value="cDNA_oligo_dT"/>
      <xs:enumeration value="PolyA">
        <xs:annotation>
          <xs:documentation>PolyA  selection  or  enrichment  for  messenger  RNA  (mRNA);
should replace cDNA enumeration. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Oligo-dT">
        <xs:annotation>
          <xs:documentation>enrichment  of  messenger  RNA  (mRNA)  by  hybridization  to
Oligo-dT. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Inverse rRNA">
        <xs:annotation>
          <xs:documentation>depletion  of  ribosomal  RNA  by  oligo  hybridization.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Inverse rRNA selection">
        <xs:annotation>
          <xs:documentation>depletion of ribosomal RNA by inverse oligo hybridization.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ChIP">
        <xs:annotation>
          <xs:documentation>Chromatin immunoprecipitation</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ChIP-Seq">
        <xs:annotation>
          <xs:documentation>Chromatin  immunoPrecipitation,  reveals  binding  sites  of
specific proteins, typically transcription factors (TFs) using antibodies to extract
DNA fragments bound to the target protein.</xs:documentation>
```



```
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MNase">
        <xs:annotation>
          <xs:documentation>Identifies well-positioned nucleosomes. uses Micrococcal
Nuclease (MNase) is an endo-exonuclease that processively digests DNA until an
obstruction, such as a nucleosome, is reached.</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="DNase">
        <xs:annotation>
          <xs:documentation>DNase I endonuclease digestion and size selection reveals
regions of chromatin where the DNA is highly sensitive to DNase I.</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Hybrid Selection">
        <xs:annotation>
          <xs:documentation>Selection by hybridization in array or
solution.</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Reduced Representation">
        <xs:annotation>
          <xs:documentation>
            Reproducible genomic subsets, often generated by restriction fragment size
selection, containing a manageable
            number of loci to facilitate re-sampling.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Restriction Digest">
        <xs:annotation>
          <xs:documentation> DNA fractionation using restriction enzymes.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="5-methylcytidine antibody">
        <xs:annotation>
          <xs:documentation>
            Selection of methylated DNA fragments using an antibody raised against 5-
methylcytosine or 5-methylcytidine
            (m5C).
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MBD2 protein methyl-CpG binding domain">
        <xs:annotation>
          <xs:documentation> Enrichment by methyl-CpG binding domain.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="CAGE">
        <xs:annotation>
          <xs:documentation> Cap-analysis gene expression. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RACE">
        <xs:annotation>
          <xs:documentation> Rapid Amplification of cDNA Ends. </xs:documentation>
```

```
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MDA">
        <xs:annotation>
          <xs:documentation>
            Multiple Displacement Amplification, a non-PCR based DNA amplification
technique that amplifies a minute
            quantifies of DNA to levels suitable for genomic analysis.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="padlock probes capture method">
        <xs:annotation>
          <xs:documentation>
            Targeted sequence capture protocol covering an arbitrary set of
nonrepetitive genomics targets. An example is
            capture bisulfite sequencing using padlock probes (BSPP).
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="other">
        <xs:annotation>
          <xs:documentation> Other library enrichment, screening, or selection process.
</xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="unspecified">
        <xs:annotation>
          <xs:documentation>  Library  enrichment,  screening,  or  selection  is  not
specified. </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <!-- STRING ENUMERATIONS END -->


  <xs:complexType name="PoolMemberType">
    <xs:complexContent>
      <xs:extension base="com:RefObjectType">
        <xs:sequence>
          <xs:element name="READ_LABEL" minOccurs="0" maxOccurs="unbounded">
            <xs:complexType>
              <xs:simpleContent>
                <xs:extension base="xs:string">
                  <xs:attribute name="read_group_tag" type="xs:string">
                    <xs:annotation>
                      <xs:documentation> Assignment of read_group_tag to decoded read
</xs:documentation>
                    </xs:annotation>
                  </xs:attribute>
                </xs:extension>
              </xs:simpleContent>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
        <xs:attribute name="member_name" type="xs:string" use="optional">
          <xs:annotation>
            <xs:documentation>  Label  a  sample  within  a  scope  of  the  pool
```



```
</xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="proportion" type="xs:float" use="optional">
          <xs:annotation>
            <xs:documentation> Proportion of this sample (in percent) that was included
in sample pool. </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SampleDescriptorType">
    <xs:complexContent>
      <xs:extension base="com:RefObjectType">
        <xs:choice minOccurs="0" maxOccurs="1">
          <xs:element name="POOL">
            <xs:annotation>
              <xs:documentation>
                Identifies a list of group/pool/multiplex sample members.  This implies
that
                this sample record is a group, pool, or multiplex, but is continues to
receive
                its own accession and can be referenced by an experiment.  By default
if
                no match to any of the listed members can be determined, then the
default
                sampel reference is used.
              </xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element  name="DEFAULT_MEMBER"  type="PoolMemberType"  minOccurs="0"
maxOccurs="1">
                  <xs:annotation>
                    <xs:documentation>
                      Reference to the sample that is used when read membership cannot
be determined. A default member should
                      be provided if there exists a possibility that some reads will be
left over from barcode/MID resolution. A default member
                      is not needed when defining a true pool (where individual samples
are not distinguished in the reads), or the reads have
                      been partitioned among the pool members (no leftovers).
                    </xs:documentation>
                  </xs:annotation>
                </xs:element>
                <xs:element    name="MEMBER"    type="PoolMemberType"    minOccurs="1"
maxOccurs="unbounded">
                  <xs:annotation>
                    <xs:documentation> Reference to the sample as determined from
barcode/MID resolution or read partition. </xs:documentation>
                  </xs:annotation>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:choice>
      </xs:extension>
    </xs:complexContent>
```

```
  </xs:complexType>
  <xs:complexType name="LibraryDescriptorType">
    <xs:annotation>
      <xs:documentation>
        The LIBRARY_DESCRIPTOR specifies the origin of the material being
        sequenced and any treatments that the material might have undergone that affect
the
        sequencing result. This specification is needed even if the platform does not
        require a library construction step per se.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="LIBRARY_NAME" type="xs:string" maxOccurs="1" minOccurs="0">
        <xs:annotation>
          <xs:documentation>
            The submitter's name for this library.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element   name="LIBRARY_STRATEGY"   type="typeLibraryStrategy"   minOccurs="1"
maxOccurs="1"/>
      <xs:element    name="LIBRARY_SOURCE"    type="typeLibrarySource"    minOccurs="1"
maxOccurs="1"/>
      <xs:element  name="LIBRARY_SELECTION"  type="typeLibrarySelection"  minOccurs="1"
maxOccurs="1"/>
      <xs:element name="LIBRARY_LAYOUT">
        <xs:annotation>
          <xs:documentation>
            LIBRARY_LAYOUT  specifies  whether  to  expect  single,  paired,  or  other
configuration of reads.
            In the case of paired reads, information about the relative distance and
orientation is specified.
          </xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:choice>
            <xs:element name="SINGLE">
              <xs:complexType>
                <xs:annotation>
                  <xs:documentation>
                    Reads are unpaired (usual case).
                  </xs:documentation>
                </xs:annotation>
              </xs:complexType>
            </xs:element>
            <xs:element name="PAIRED">
              <xs:complexType>
                <xs:attribute name="NOMINAL_LENGTH" type="xs:nonNegativeInteger"/>
                <xs:attribute name="NOMINAL_SDEV" type="xs:double"/>
              </xs:complexType>
            </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
      <xs:element name="TARGETED_LOCI" minOccurs="0" maxOccurs="1">
        <xs:complexType>
          <xs:annotation>
            <xs:documentation>
              Names the gene(s) or locus(loci) or other genomic feature(s) targeted by
```



```
the sequence.
            </xs:documentation>
          </xs:annotation>
          <xs:sequence>
            <xs:element name="LOCUS" maxOccurs="unbounded" minOccurs="1">
              <xs:complexType>
                <xs:sequence>
                  <xs:element    name="PROBE_SET"    type="com:XRefType"    maxOccurs="1"
minOccurs="0">
                    <xs:annotation>
                      <xs:documentation>
                        Reference to an archived primer or
                        probe set. Example: dbProbe
                      </xs:documentation>
                    </xs:annotation>
                  </xs:element>
                </xs:sequence>
                <xs:attribute name="locus_name">
                  <xs:simpleType>
                    <xs:restriction base="xs:string">
                      <xs:enumeration value="16S rRNA">
                        <xs:annotation>
                          <xs:documentation>
                            Bacterial small subunit ribosomal RNA, a locus used for
                            phylogenetic studies of bacteria and as a target for random
target PCR in
                            environmental biodiversity screening.
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="18S rRNA">
                        <xs:annotation>
                          <xs:documentation>
                            Eukaryotic small subunit ribosomal RNA, a locus used for
                            phylogenetic  studies  of  eukaryotes  and  as  a  target  for
random target PCR in
                            environmental biodiversity screening.
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="RBCL">
                        <xs:annotation>
                          <xs:documentation>
                            RuBisCO large subunit : ribulose-1,5-bisphosphate
                            carboxylase/oxygenase  large  subunit,  a  locus  used  for
phylogenetic studies
                            of plants.
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="matK">
                        <xs:annotation>
                          <xs:documentation>
                            Maturase K gene, a locus used for phylogenetic studies of
                            plants.
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="COX1">
```

```
                        <xs:annotation>
                          <xs:documentation>
                            Mitochondrial cytochrome c oxidase 1 gene, a locus used for
                            phylogenetic studies of animals
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="ITS1-5.8S-ITS2">
                        <xs:annotation>
                          <xs:documentation>
                            Internal transcribed spacers 1 and 2 plus 5.8S rRNA region,
                            a locus used for phylogenetic studies of fungi.
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="exome">
                        <xs:annotation>
                          <xs:documentation>  All  exonic  regions  of  the  genome.
</xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                      <xs:enumeration value="other">
                        <xs:annotation>
                          <xs:documentation>
                            Other locus, please describe.
                          </xs:documentation>
                        </xs:annotation>
                      </xs:enumeration>
                    </xs:restriction>
                  </xs:simpleType>
                </xs:attribute>
                <xs:attribute name="description" type="xs:string">
                  <xs:annotation>
                    <xs:documentation>
                      Submitter supplied description of alternate locus and auxiliary
                      information.
                    </xs:documentation>
                  </xs:annotation>
                </xs:attribute>
              </xs:complexType>
            </xs:element>

          </xs:sequence>


        </xs:complexType>
      </xs:element>
      <xs:element name="POOLING_STRATEGY" minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation>
            The optional pooling strategy indicates how the library or libraries are
organized if multiple samples are involved.
          </xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string"> </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element  name="LIBRARY CONSTRUCTION PROTOCOL"  type="xs:string"  minOccurs="0"
```



```
maxOccurs="1">
        <xs:annotation>
          <xs:documentation>
            Free form text describing the protocol by which the sequencing library was
constructed.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="LibraryType">
    <xs:sequence>
      <xs:element name="DESIGN_DESCRIPTION" type="xs:string">
        <xs:annotation>
          <xs:documentation>Goal and setup of the individual library including library
was constructed.</xs:documentation>
        </xs:annotation>
      </xs:element>

      <xs:element name="SAMPLE_DESCRIPTOR" type="SampleDescriptorType">
        <xs:annotation>
          <xs:documentation>
            Pick a sample to associate this experiment with. The sample may be an
individual or a pool,
            depending on how it is specified.
          </xs:documentation>
        </xs:annotation>
      </xs:element>

      <xs:element name="LIBRARY_DESCRIPTOR" type="LibraryDescriptorType">
        <xs:annotation>
          <xs:documentation>
            The LIBRARY_DESCRIPTOR specifies the origin of the material being sequenced
and any
            treatments that the material might have undergone that affect the
sequencing result. This specification is
            needed even if the platform does not require a library construction step
per se.
          </xs:documentation>
        </xs:annotation>
      </xs:element>

      <xs:element name="SPOT_DESCRIPTOR" type="com:SpotDescriptorType" minOccurs="0"
maxOccurs="1">
        <xs:annotation>
          <xs:documentation>
            The SPOT_DESCRIPTOR specifies how to decode the individual reads of
interest from the
            monolithic spot sequence. The spot descriptor contains aspects of the
experimental design, platform, and
            processing information. There will be two methods of specification: one
will be an index into a table of
            typical decodings, the other being an exact specification. This construct
is needed for loading data and for
            interpreting the loaded runs. It can be omitted if the loader can infer
read layout (from multiple input
            files or from one input files).
          </xs:documentation>
        </xs:annotation>
```

```
      </xs:element>
    </xs:sequence>

  </xs:complexType>

  <xs:complexType name="ExperimentExtensionElement">
    <xs:sequence>
      <xs:element name="title" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="design" type="LibraryType" minOccurs="1" maxOccurs="1"/>
      <xs:element name="platform" type="com:PlatformType" minOccurs="1" maxOccurs="1"/>
      <xs:element     name="processing"     type="com:ProcessingType"     minOccurs="0"
maxOccurs="1"/>
      <xs:element name="EXPERIMENT_LINKS" minOccurs="0" maxOccurs="1">
        <xs:complexType>
          <xs:sequence minOccurs="1" maxOccurs="unbounded">
            <xs:element name="EXPERIMENT_LINK" type="com:LinkType"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="EXPERIMENT_ATTRIBUTES" minOccurs="0" maxOccurs="1">
        <xs:complexType>
          <xs:sequence maxOccurs="unbounded" minOccurs="1">
            <xs:element name="EXPERIMENT_ATTRIBUTE" type="com:AttributeType"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>

    <xs:attribute name="alias" type="xs:string" use="optional"/>
    <xs:attribute name="center_name" type="xs:string" use="optional"/>
    <xs:attribute name="broker_name" type="xs:string" use="optional"/>
    <xs:attribute name="accession" type="xs:string" use="optional"/>
  </xs:complexType>
</xs:schema>
```

A.5 Object identifiers extension

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:mpeg:mpeg-g:metadata:extension:ega"
  elementFormDefault="qualified"
  xmlns="urn:mpeg:mpeg-g:metadata:extension:ega"
  xmlns:tns="urn:mpeg:mpeg-g:metadata:extension:ega"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:com="SRA.common"
>
  <xs:import    schemaLocation="ftp://ftp.sra.ebi.ac.uk/meta/xsd/sra_1_5/SRA.common.xsd"
namespace="SRA.common"/>
  <xs:complexType name="ObjectType">
    <xs:sequence>
      <xs:element         maxOccurs="1"         minOccurs="0"         name="IDENTIFIERS"
type="com:IdentifierType"/>
    </xs:sequence>
    <xs:attribute name="alias" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation>
          Submitter designated name for the object. The name must be unique within the
```



```
submission account.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="center_name" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation>
          The center name of the submitter.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="broker_name" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation>
          The center name of the broker.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="accession" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation>
          The object accession assigned by the archive.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:schema>
```

A.6 Dataset group extensions

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:mpeg:mpeg-g:metadata:extension:ega"
  elementFormDefault="qualified"
  xmlns="urn:mpeg:mpeg-g:metadata:extension:ega"
  xmlns:tns="urn:mpeg:mpeg-g:metadata:extension:ega"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:com="SRA.common"
>
  <xs:import    schemaLocation="ftp://ftp.sra.ebi.ac.uk/meta/xsd/sra_1_5/SRA.common.xsd"
namespace="SRA.common"/>

  <xs:complexType name="DatasetGroupLinks">
    <xs:annotation>
      <xs:documentation>
        Links to resources related to this study (publication, datasets, online
databases).
      </xs:documentation>
    </xs:annotation>
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
      <xs:element name="STUDY_LINK" type="com:LinkType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="DatasetGroupAttributes">
    <xs:annotation>
      <xs:documentation>
        Properties and attributes of the study.  These can be entered as free-form
        tag-value pairs. For certain studies, submitters may be asked to follow a
```

```
        community established ontology when describing the work.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
      <xs:element name="STUDY_ATTRIBUTE" type="com:AttributeType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="RelatedStudies">
    <xs:sequence>
      <xs:element name="RELATED_STUDY" maxOccurs="unbounded" minOccurs="1">
        <xs:complexType>
          <xs:sequence>
            <xs:element     name="RELATED_LINK"     type="com:XRefType"     minOccurs="1"
maxOccurs="1">
              <xs:annotation>
                <xs:documentation>
                Related study or project record from a list of supported databases.
                The study's information is derived from this project record rather
                than stored as first class information.
              </xs:documentation>
              </xs:annotation>
            </xs:element>
            <xs:element      name="IS_PRIMARY"      type="xs:boolean"      minOccurs="1"
maxOccurs="1">
              <xs:annotation>
                <xs:documentation>
                Whether this study object is designated as the primary source
                of the study or project information.
              </xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="StudyType">
    <xs:annotation>
      <xs:documentation>The STUDY_TYPE presents a controlled vocabulary for expressing
the overall purpose of the study.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="existing_study_type" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="Whole Genome Sequencing">
            <xs:annotation>
              <xs:documentation>
                Sequencing of a single organism.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Metagenomics">
            <xs:annotation>
              <xs:documentation>
                Sequencing of a community.
              </xs:documentation>
            </xs:annotation>
```



```
          </xs:enumeration>
          <xs:enumeration value="Transcriptome Analysis">
            <xs:annotation>
              <xs:documentation>
                Sequencing and characterization of transcription elements.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Resequencing">
            <xs:annotation>
              <xs:documentation>
                Sequencing of a sample with respect to a reference.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Epigenetics">
            <xs:annotation>
              <xs:documentation>
                Cellular differentiation study.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Synthetic Genomics">
            <xs:annotation>
              <xs:documentation>
                Sequencing of modified, synthetic, or transplanted genomes.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Forensic or Paleo-genomics">
            <xs:annotation>
              <xs:documentation>
                Sequencing of recovered genomic material.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Gene Regulation Study">
            <xs:annotation>
              <xs:documentation>
                Study of gene expression regulation.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Cancer Genomics">
            <xs:annotation>
              <xs:documentation>
                Study of cancer genomics.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Population Genomics">
            <xs:annotation>
              <xs:documentation>
                Study of populations and evolution through genomics.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="RNASeq">
            <xs:annotation>
```

```
              <xs:documentation>
                RNA sequencing study.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Exome Sequencing">
            <xs:annotation>
              <xs:documentation>
                The study investigates the exons of the genome.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Pooled Clone Sequencing">
            <xs:annotation>
              <xs:documentation>
                The study is sequencing clone pools (BACs, fosmids, other constructs).
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Transcriptome Sequencing">
            <xs:annotation>
              <xs:documentation>     Sequencing     of     transcription     elements.
</xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Other">
            <xs:annotation>
              <xs:documentation>
                Study type not listed.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="new_study_type" use="optional" type="xs:string">
      <xs:annotation>
        <xs:documentation>
          To propose a new term, select Other and enter a new study type.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>

  <xs:complexType name="projectIdentification">
    <xs:sequence>
      <xs:element    type="xs:nonNegativeInteger"    name="project_id"    minOccurs="0"
maxOccurs="1"/>
      <xs:element     type="xs:string"     name="center_project_name"     minOccurs="0"
maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

A.7 Dataset type extension

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:mpeg:mpeg-g:metadata:extension:ega"
```

```
    elementFormDefault="qualified"
    xmlns="urn:mpeg:mpeg-g:metadata:extension:ega"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:com="SRA.common"
>
  <xs:import    schemaLocation="ftp://ftp.sra.ebi.ac.uk/meta/xsd/sra_1_5/SRA.common.xsd"
namespace="SRA.common"/>

  <xs:simpleType name="DatasetType">
      <xs:restriction base="xs:string">
        <xs:enumeration value="Whole genome sequencing"/>
        <xs:enumeration value="Exome sequencing"/>
        <xs:enumeration value="Genotyping by array"/>
        <xs:enumeration value="Transcriptome profiling by high-throughput sequencing"/>
        <xs:enumeration value="Transcriptome profiling by array"/>
        <xs:enumeration value="Amplicon sequencing"/>
        <xs:enumeration value="Methylation binding domain sequencing"/>
        <xs:enumeration value="Methylation profiling by high-throughput sequencing"/>
        <xs:enumeration value="Phenotype information"/>
        <xs:enumeration value="Study summary information"/>
        <xs:enumeration value="Genomic variant calling"/>
        <xs:enumeration  value="Chromatin  accessibility  profiling  by  high-throughput
sequencing"/>
        <xs:enumeration  value="Histone  modification  profiling  by  high-throughput
sequencing"/>
        <xs:enumeration value="Chip-Seq"/>
      </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="DatasetLinks">
    <xs:annotation>
      <xs:documentation>Links to related resources.</xs:documentation>
    </xs:annotation>
    <xs:sequence maxOccurs="unbounded" minOccurs="1">
      <xs:element name="DATASET_LINK" type="com:LinkType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="DatasetAttributes">
    <xs:annotation>
      <xs:documentation>Properties and attributes of the data set. These can be entered
as free-form tag-value pairs. Submitters may be asked to follow a community established
ontology when describing the work.          </xs:documentation>
    </xs:annotation>
    <xs:sequence maxOccurs="unbounded" minOccurs="1">
      <xs:element name="DATASET_ATTRIBUTE" type="com:AttributeType"/>
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

A.8 Run extension

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:mpeg:mpeg-g:metadata:extension:ega"
    elementFormDefault="qualified"
    xmlns="urn:mpeg:mpeg-g:metadata:extension:ega"
```

```
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:com="SRA.common"
>
  <xs:import    schemaLocation="ftp://ftp.sra.ebi.ac.uk/meta/xsd/sra_1_5/SRA.common.xsd"
namespace="SRA.common"/>

  <xs:complexType name="RunExtension">
    <xs:sequence>
      <xs:element name="RUN_LINKS" minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation>
              Links to resources related to this RUN or RUN set (publication, datasets,
online databases).
          </xs:documentation>
          </xs:annotation>
          <xs:complexType>
              <xs:sequence minOccurs="1" maxOccurs="1">
                  <xs:element name="RUN_LINK" type="com:LinkType"
                      maxOccurs="unbounded"/>
              </xs:sequence>
          </xs:complexType>
      </xs:element>

      <xs:element name="RUN_ATTRIBUTES" minOccurs="0" maxOccurs="1">
          <xs:annotation>
              <xs:documentation>
              Properties and attributes of a RUN.  These can be entered as free-form
              tag-value pairs. For certain studies, submitters may be asked to follow a
              community established ontology when describing the work.
          </xs:documentation>
          </xs:annotation>
          <xs:complexType>
              <xs:sequence maxOccurs="1" minOccurs="1">
                  <xs:element name="RUN_ATTRIBUTE" type="com:AttributeType"
                      maxOccurs="unbounded"/>
              </xs:sequence>
          </xs:complexType>
      </xs:element>

      <xs:element name="SPOT_DESCRIPTOR" type="com:SpotDescriptorType" maxOccurs="1"
                       minOccurs="0"/>

      <xs:element name="PLATFORM" type="com:PlatformType" maxOccurs="1" minOccurs="0"/>

      <xs:element         name="PROCESSING"         maxOccurs="1"         minOccurs="0"
type="com:ProcessingType"/>

      <xs:element maxOccurs="1" minOccurs="0" name="RUN_TYPE">
        <xs:annotation>
          <xs:documentation>The type of the run. </xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:choice>
            <xs:element name="REFERENCE_ALIGNMENT"
                type="com:ReferenceSequenceType"> </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
```



```
    <xs:attribute name="run_date" use="optional" type="xs:dateTime">
      <xs:annotation>
        <xs:documentation>
          ISO date when the run took place.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="run_center" use="optional" type="xs:string">
      <xs:annotation>
        <xs:documentation>
          If applicable, the name of the contract sequencing center that executed the
run.
          Example: 454MSC.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:schema>
```

A.9 Dataset group protection `gen_info` XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema       attributeFormDefault="unqualified"       elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="urn:mpeg:mpegen/protection_datasetgroup"
xmlns="urn:mpeg:mpegen/protection_datasetgroup">
  <xs:import                                  namespace="http://www.w3.org/2001/04/xmlenc#"
schemaLocation="https://www.w3.org/TR/2002/REC-xmlenc-core-20021210/xenc-schema.xsd"/>
  <xs:import                                 namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="https://www.w3.org/TR/2002/REC-xmldsig-core-20020212/xmldsig-core-
schema.xsd#enveloped-signature"/>
  <xs:element name="protection" type="protectionType"/>

  <xs:complexType name="protectionType">
    <xs:sequence>
      <xs:element type="encryptionsType" name="encryptions"/>
      <xs:element type="signaturesType" name="signatures"/>
      <xs:element      type="xd:SignatureType"      name="signature"      minOccurs="0"
xmlns:xd="http://www.w3.org/2000/09/xmldsig#"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="encryptionsType">
    <xs:sequence>
      <xs:element     ref="xe:EncryptedData"     maxOccurs="unbounded"     minOccurs="0"
xmlns:xe="http://www.w3.org/2001/04/xmlenc#"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="signaturesType">
    <xs:sequence>
      <xs:element      ref="xd:Signature"      maxOccurs="unbounded"      minOccurs="0"
xmlns:xd="http://www.w3.org/2000/09/xmldsig#"/>
    </xs:sequence>
  </xs:complexType>
```

```
</xs:schema>
```

A.10 Dataset protection `gen_info` XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema       attributeFormDefault="unqualified"       elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="protection" type="protectionType"/>
  <xs:complexType name="protectionType">
    <xs:sequence>
      <xs:element type="encryptionsType" name="encryptions"/>
      <xs:element type="signaturesType" name="signatures"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="encryptionsType">
    <xs:sequence>
      <xs:element     ref="xe:EncryptedData"     maxOccurs="unbounded"     minOccurs="0"
xmlns:xe="http://www.w3.org/2001/04/xmlenc#"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="signaturesType">
    <xs:sequence>
      <xs:element      ref="xd:Signature"      maxOccurs="unbounded"      minOccurs="0"
xmlns:xd="http://www.w3.org/2000/09/xmldsig#"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

A.11 Privacy rule and authorization request

Example of privacy rule.

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
    http://docs.oasis-open.org/xacml/3.0/xacml-core-v3-schema-wd-17.xsd"
    PolicyId="urn:isdcm:policyid:1"
    RuleCombiningAlgId=
    "urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable"
    Version="1.0">
    <Description> Policy getDataDataset </Description>
    <Target/>

    <Rule RuleId="urn:oasis:names:tc:xacml:2.0:ejemplo:RuleGen" Effect="Permit">
        <Description> Get Data from Dataset </Description>
        <Target>
            <AnyOf>
                <AllOf>
                    <!-- Which kind of user: researcher -->
                    <Match MatchId=
                        "urn:oasis:names:tc:xacml:1.0:function:string-equal">
                        <AttributeValue DataType=
                           "http://www.w3.org/2001/XMLSchema#string"
                            >researcher</AttributeValue>
                        <AttributeDesignator MustBePresent="false"
                         Category=
                         "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
```



```
                    AttributeId=
                    "urn:oasis:names:tc:xacml:3.0:example:attribute:role"
                    DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </Match>

                <!-- Which resource -->
                <Match MatchId=
                    "urn:oasis:names:tc:xacml:1.0:function:regexp-string-match">
                    <AttributeValue DataType=
                     "http://www.w3.org/2001/XMLSchema#string"
                        >urn:mpgen:file.gen</AttributeValue>
                    <AttributeDesignator MustBePresent="false"
                        Category=
                        "urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
                        AttributeId="urn:oasis:names:tc:xacml:1.0:resource:
                        simple-file-name"
                        DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </Match>

                <!-- Which action  -->
                <Match MatchId=
                    "urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue DataType=
                     "http://www.w3.org/2001/XMLSchema#string"
                        >GetDataDataset</AttributeValue>
                    <AttributeDesignator MustBePresent="false"
                        Category=
                        "urn:oasis:names:tc:xacml:3.0:attribute-category:action"
                        AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
                        DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </Match>
            </AllOf>
        </AnyOf>
    </Target>
    <Condition>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
            <Apply FunctionId=
                "urn:oasis:names:tc:xacml:1.0:function:date-less-than-or-equal">
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:date-one-and-only">
                    <AttributeDesignator MustBePresent="false" Category="date"
                        AttributeId="accessDate"
                        DataType="http://www.w3.org/2001/XMLSchema#date"/>
                </Apply>
                <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#date"
                    >2019-01-01</AttributeValue>
            </Apply>

            <Apply FunctionId=
                "urn:oasis:names:tc:xacml:1.0:function:integer-equal">
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                    <AttributeDesignator
                        MustBePresent="true" Category="dgidentifier"
                        AttributeId="dataset_group_ID"
                        DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                </Apply>
                <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                    >1</AttributeValue>
```

```
                </Apply>
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator
                            MustBePresent="true" Category="dsidentifier"
                            AttributeId="dataset_ID"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >1</AttributeValue>
                </Apply>
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="type"
                            AttributeId="class_type"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >1</AttributeValue>
                </Apply>
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="count"
                            AttributeId="read_count"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >1</AttributeValue>
                </Apply>
                <Apply FunctionId=
                 "urn:oasis:names:tc:xacml:1.0:function:integer-greater-than-or-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="position"
                            AttributeId="start_position"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >1</AttributeValue>
                </Apply>
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-less-than-or-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="position"
                            AttributeId="end_position"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >100</AttributeValue>
                </Apply>
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-equal">
```



```
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="sequence"
                            AttributeId="reference_sequence"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >1</AttributeValue>
                </Apply>

                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:boolean-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:boolean-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="alignment"
                            AttributeId="presence_of_multiple_alignments"
                            DataType="http://www.w3.org/2001/XMLSchema#boolean"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#boolean"
                        >true</AttributeValue>
                </Apply>
                <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:double-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:double-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="score"
                            AttributeId="alignment_score"
                            DataType="http://www.w3.org/2001/XMLSchema#double"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#double"
                        >0.9</AttributeValue>
                </Apply>
                <Apply FunctionId=
                    "urn:oasis:names:tc:xacml:1.0:function:integer-equal">
                    <Apply FunctionId=
                        "urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only">
                        <AttributeDesignator MustBePresent="true" Category="threshold"
                            AttributeId="mismatch_threshold"
                            DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                    </Apply>
                    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                        >1</AttributeValue>
                </Apply>
            </Apply>
            <!-- closes and date parameters -->
        </Condition>
    </Rule>
</Policy>
```

Example of authorization request for the privacy rule.

```
<Request xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17
    http://docs.oasis-open.org/xacml/3.0/xacml-core-v3-schema-wd-17.xsd"
    ReturnPolicyIdList="false" CombinedDecision="false">

    <Attributes Category=
       "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
```

```
        <Attribute AttributeId="urn:oasis:names:tc:xacml:3.0:example:attribute:role"
            IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string"
                >researcher</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource">
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:simple-file-name"
            IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string"
                >urn:mpgen:file.gen</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="urn:oasis:names:tc:xacml:3.0:attribute-category:action">
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
            IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string"
                >GetDataDataset</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="urn:oasis:names:tc:xacml:3.0:date">
        <Attribute AttributeId="accessDate" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#date"
                >2018-04-10</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="dgidentifier">
        <Attribute AttributeId="dataset_group_ID" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >1</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="dsidentifier">
        <Attribute AttributeId="dataset_ID" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >1</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="type">
        <Attribute AttributeId="class_type" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >1</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="count">
        <Attribute AttributeId="read_count" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >1</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="urn:oasis:names:tc:xacml:3.0:unsignedInteger">
```



```
        <Attribute AttributeId="position" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >50</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="sequence">
        <Attribute AttributeId="reference_sequence" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >1</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="alignment">
        <Attribute AttributeId=
          "presence_of_multiple_alignments" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#boolean"
                >true</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="score">
        <Attribute AttributeId="alignment_score" IncludeInResult="true">
            <AttributeValue DataType=
             "http://www.w3.org/2001/XMLSchema#double">0.9</AttributeValue>
        </Attribute>
    </Attributes>

    <Attributes Category="threshold">
        <Attribute AttributeId="mismatch_threshold" IncludeInResult="true">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#integer"
                >1</AttributeValue>
        </Attribute>
    </Attributes>

</Request>
```

Bibliography

[1] W3C, "Extensible Markup Language (XML) 1.1 (Second Edition), 2016, Available: https://www.w3.org/TR/xml11/

[2] EBI, "Read domain XML 1.5 metadata format", Available: https://www.ebi.ac.uk/ena/submit/read-xml-format-1-5

[3] NCBI, "Preview BioSample types and attributes", Available: https://submit.ncbi.nlm.nih.gov/biosample/template/

[4] G. S. C. "MIxS GSC Project," 2016, Available: http://gensc.org/projects/mixs-gsc-project/

[5] EGA, European genome-phenome archive, Available: https://ega-archive.org/

[6] W3C, XML Encryption Syntax and Processing Version 1.1, 2013, Available: https://www.w3.org/TR/xmlenc-core1/

[7] OASIS, eXtensible Access Control Markup Language (XACML) Version 3.0, 2013, Available: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf

[8] W3C, XML Signature Syntax and Processing Version 1.1, 2013, Available: https://www.w3.org/TR/xmldsig-core1/

[9] ISO/IEC JTC 1/SC 29/WG 11, N17140 “Text of ISO/IEC 23092-1 CD, Coding of Genomic Information” Macao, October 2017

[10] SAM/BAM Format Specification Working Group, Sequence Alignment/Map format specification, 1 06 2017, Available: https://samtools.github.io/hts-specs/SAMv1.pdf

[11] L. Stein, "Generic Feature Format Version 3 (GFF3)", 2013, Available: https://github.com/The-Sequence-Ontology/Specifications/blob/master/gff3.md

[12] Global Alliance for Genomics and Health. A federated ecosystem for sharing genomic, clinical data. Science 352, 1278–1280, 2016, Available https://www.ga4gh.org/

