Standard Archive Format for Europe

Control Book
Volume 2

Recommendation for specialisations

Reference	PGSI-GSEG-EOPG-FS-05-0002		issue 1	revision 11
Author(s)	Mathias Moucha Stephane Mbaye	GAEL Consultant GAEL Consultant	date	
Reviewed by			date	
Approved by	Dario Romano	European Space Agency (ESA)	date	

European Space Agency
Agence spatiale européenne

ESRIN
Via Galileo Galilei - Casella Postale 64 - 00044 Frascati - Italy
Tel. (39) 06 941801 - Fax (39) 06 94180 280

Table of Contents

List of Tables

Preface

The present document provides the general recommendations for specializing SAFE specifications to a specific context, content or purpose. This recommendations have been developed by the European Space Agency in the framework of its Earth Observation ground segment activities.

SAFE product format has been designed to be an opened shell capable of holding as many type of Earth Observation data as reasonably feasible at the time of writing. SAFE inherits its main structure from XFDU packaging format and defines high level constraints and new rules coping with most of Earth Observation data products currently archived and exchanged across ESA and cooperating organizations. These high level definitions provides all SAFE with a common interface making easier their manipulation and exchange across various facilities, systems and environments. For defining more complete products with specific constraints, rules and specific types of data, it is, however, necessary to design a so called "SAFE Specialisation" and possibly to write a corresponding SAFE Specialisation Control Book and produce the accompanying XML Schema's and examples.

This book provides rules and guidelines for designing and implementing a SAFE Specialisation and intends to provide the necessary guidance to assure a consistent and homogeneous development of SAFE Specialisation across projects and organizations.

1. This Book Audience

This book is primarily dedicated to technical managers or developers involved in the design of a SAFE Specialisation. It is also recommended for SAFE implementers or users desiring to improve their skill about the format. In particular, this book provides explanations regarding technical solutions selected in the [SAFE-CORE] specifications, because this latter does not intend to act as a guideline document.

It is expected that most readers will have some familiarity with OAIS concepts as well as with XFDU and XML technologies.

However, although many information regarding [SAFE-CORE] specifications is provided, this book does not suffice as tutorial about SAFE standard. It is highly recommended to reach at least a basic skill with respect to SAFE from the [SAFE-CORE] document before reading this book.

2. Organisation of This Book

Chapter 1:	Provides an overview of a SAFE Specialisation.
Chapter 2:	Provides recommendations for a SAFE Product to be well structured.
Chapter 3:	Provides recommendations for manipulation of all Manifest Types.
Chapter 4:	Provides recommendations for the Representation Information of Components.
Chapter 5:	Provides recommendations for identification of a SAFE Product.
Chapter 6:	Provides recommendations for namespaces used in SAFE.
Chapter 7:	Provides recommendations to define filename of Components, name of different entities gathered by a SAFE Product.
Appendix A:	A complete example of ERS AMI SAR Level 0 Product SAFE Manifest document.

Appendix B: GNU Free Documentation License.

Appendix C: A standard form for proposing improvements to the present book.

Book Index: An alphabetical list of the specific information included in the book. It is prepared to help the reader find information quickly and easily.

3. SAFE Specification Volume Set

Although SAFE Recommendations for specialisations are fully defined in the present book, this latest is part of a consistent set of books defining *official* specialisations dedicated to various kind of data.

The following list references these related documents with their revision status at the time of writing. It, however, does not significate that other specialisations are not available or in the pipeline of the development of any other party.

[SAFE-CORE] *Standard Archive Format for Europe Control Book - Volume 1 - Core Specifications*-PGSI-GSEG-EOPG-FS-05-0001- Issue 1 , Revision 14-2010-11-12-Copyright © 2006,2007,2008 European Space Agency (ESA)GAEL Consultant-

4. Bibliography

[OAIS-RM] *Reference Model for an Open Archival Information System (OAIS)* -650.0-B-1-January 2002-Blue Book-Copyright © 2002 Consultative Committee for Space Data Systems (CCSDS) -

[SI] *The International System of Units (SI)* -1998-7th edition- Bureau International des Poids et Mesures -Copyright © 1998 Organisation Intergouvernementale de la Convention du Mètre -

[SI-SUP2000] *The International System of Units (SI)*- Supplement 2000: addenda and corrigenda of the 7th edition (1998) -1998-7th edition- Bureau International des Poids et Mesures -Copyright © 1998 Organisation Intergouvernementale de la Convention du Mètre -

[XFDU] *XML Formatted Data Unit (XFDU) - Structure and Construction Rules* -661.0-B-1-September 2008-Blue Book -Copyright © 2008 Consultative Committee for Space Data Systems (CCSDS) -

[XML] *eXtensible Markup Language (XML) 1.0 (Fifth Edition)* -W3C Recommendation-November 26, 2008-Version 1.0-Copyright © 2008 World Wide Web Consortium (W3C) -

[XML-SCHEMA] *XML Schema: Primer Second Edition* -W3C Recommendation-October 28, 2004-Version 1.0-Copyright © 2004 World Wide Web Consortium (W3C) -

[XML-SCHEMA-STRUCT] *XML Schema: Structures Second Edition* -W3C Recommendation-October 28, 2004-Version 1.0-Copyright © 2004 World Wide Web Consortium (W3C) -

[XML-SCHEMA-TYPES] *XML Schema: Data Types Second Edition* -W3C Recommendation-October 28, 2004-Version 1.0-Copyright © 2004 World Wide Web Consortium (W3C) -

5. Glossary of Terms

5.1. General Definitions

Additional abstract XML Schema	An additional abstract XML Schema is not the XML Schema which describes a Data Component or a Metadata Component, but is included by the XML Schema which describes a Data Component or a Metadata Component. An additional abstract XML Schema is a Component of a SAFE Product.
Association	Refers to a relationship between Components in a Collection, or other Metadata related to a Component or the Collection [XFDU].
Collection	Refers to Components that are gathered together along with a Manifest. This is analogous to files on a file system [XFDU].
Component	Refers to a file that can be grouped together to be part of a Collection, or XFDU Package [XFDU], or SAFE Product. A Component may be a Data Component, a Metadata Component, an XML Schema Component, etc.
Consumer	The role played by those persons, [...], who interact with OAIS services to find preserved information of interest and to access that information in detail. This can include other OAIS, as well as internal OAIS persons or systems [OAIS-RM].
Content Unit	XML Structure that contains pointers to Data Objects and associated Metadata Objects, and possibly other Content Units [XFDU].
Data	A reinterpretable representation of Information in a formalized manner suitable for communication, interpretation, or processing. Examples of data include a sequence of bits, a table of numbers, the characters on a page, the recording of sounds made by a person speaking, or a moon rock specimen [OAIS-RM].
Data Component	A Component holding data.
Data Object	“An object composed of a set of bit sequences”. This definition has been inherited from the [OAIS-RM] discarding the case of Physical Object (e.g. Books, Pen, etc.) that is out of the scope of the present specifications. The [XFDU] document completes this definition: “Contains some file content and any data required to allow the Information Consumer to reverse any transformations that have been performed on the object and restore it to the byte stream intended for the original Designated Community and described by the Representation Information in the Content Unit”.
dataObject	A `dataObject` element of a SAFE Manifest.
Datatype	In [XML-SCHEMA-TYPES] specification, a datatype is a 3-tuple, consisting of: 1. a set of distinct values, called its Value Space,

2. a set of lexical representations, called its Lexical Space,
3. a set of Facets that characterize properties of the Value Space, individual values or lexical items.

Designated Community	An identified group of potential Consumers who should be able to understand a particular set of Information. The Designated Community may be composed of multiple user communities [OAIS-RM].
Facet	A facet is a single defining aspect of a Value Space. Generally speaking, each facet characterizes a Value Space along independent axes or dimensions. The Datatype of a Datatype serves to distinguish those aspects of one Datatype which differ from other Datatypes. Rather than being defined solely in terms of a prose description the Datatypes in this specification are defined in terms of the synthesis of facet values which together determine the Value Space and properties of the Datatype [XML-SCHEMA-TYPES].
Fixity Information	The Information which documents the authentication mechanisms and provides authentication keys to ensure that the [...] object has not been altered in an undocumented manner. An example is a Cyclical Redundancy Check (CRC) code for a file [OAIS-RM].
Information	Any type of knowledge that can be exchanged. In an exchange, it is represented by data. An example is a string of bits (the data) accompanied by a description of how to interpret a string of bits as numbers representing temperature observations measured in degrees Celsius (the Representation Information) [OAIS-RM].
Lexical Space	A lexical space is the set of valid *literals* for a datatype [XML-SCHEMA-TYPES].
Manifest	A document containing Metadata about Components, and the Associations between them. This Information is stored as a Component, using an XML language designed just for this purpose [XFDU].
Manifest Type	Any type provided by SAFE or any SAFE Specialisation contained in a SAFE Manifest.
Metadata	Data about other Data [OAIS-RM].
Metadata Component	A Component holding Data about other Data.
Metadata Object	A SAFE Object that is a Metadata for another SAFE Object within the same SAFE Product, or XFDU Package.
metadataObject	A `metadataObject` element of a SAFE Manifest.
Package	A collection that is bundled together, or packaged, into one file using a defined packaging scheme. All Packages are Collections, but not all Collections have been packaged, so they are not all Packages [XFDU].

Information	Information which is necessary for adequate preservation of the Content Information and which can be categorized as Provenance, Reference, Fixity, and Context information [OAIS-RM].
Provenance Information	Information that documents the history of the Content Information. This information tells the origin or source of the Content Information, any changes that may have taken place since it was originated, and who has had custody of it since it was originated. Examples of Provenance Information are the principal investigator who recorded the data, and the information concerning its storage, handling, and migration [OAIS-RM].
Representation Information	The information that maps a Data Object into more meaningful concepts. An example is the ASCII definition that describes how a sequence of bits (i.e. a Data Object) is mapped into a symbol [OAIS-RM].
Package Interchange File	A collection of files that have been bundled together into a single container that also contains a manifest describing the contained files and the relationships among those files [XFDU].
Producer	The role played by those persons, or client systems, who provide the information to be preserved.
Referenced Metadata Object	A Metadata Object stored inside *and* outside the SAFE Manifest.
SAFE Manifest	A Manifest conforming the present specifications.
SAFE Object	Either a Data Object or a Metadata Object defined by SAFE.
SAFE Product	An XFDU Package specialised for Earth Observation data purposes. The term of *Product* has been selected for historical reason but matches exactly the definition of an XFDU Package introduced above.
SAFE Specialisation	A SAFE Specialisation is a restriction of the SAFE Core specifications for a more specific type of data. Examples of SAFE Specialisation include specialisations for ENVISAT or LANDSAT Products, for CCSDS Telemetry Data, or for SPOT Measurements...
SAFE Type	Type defined in the “http://www.esa.int/safe/1.3” namespace and part of SAFE.
Value Space	A value space is the set of values for a given Datatype. Each value in the value space of a Datatype is denoted by one or more literals in its Lexical Space [XML-SCHEMA-TYPES].
Wrapped Metadata Object	A Metadata Object wrapped inside the SAFE Manifest.
XFDU Package	A Package Interchange File that contains an XFDU Manifest and is conformant to the semantics specified in the XFDU Specifications. An XFDU Package is a specialization of Package Interchange File [XFDU].
XFDU Type	Type defined in the “urn:ccsds:schema:xfdu:1” namespace and part of SAFE.

Component	A Component holding part or the entire Representation Information of another Component.
XML Schema Object	A Metadata Object holding part or the entire Representation Information of another SAFE Object.

5.2. Specialisation Definitions

New Specific Type	Manifest Type defined by any SAFE Specialisation.
SAFE Abstract Specialisation	A SAFE abstract specialisation is a specialisation that does not define by itself a single product.
SAFE Auxiliary Specialisation	A restriction of the SAFE Core specifications for a specific auxiliary product. Examples of SAFE Specialisation include a specialisation for ENVISAT Predicted Orbit State Vectors Auxiliary Products, ENVISAT UTC Time Reference and Conversion Table Auxiliary Products, ERS General Headers Auxiliary Products, etc.
SAFE Product Specialisation	A restriction of the SAFE Core specifications for a specific product. Examples of SAFE Specialisation include a specialisation for ENVISAT ASAR APC Level 0 products, Landsat MSS Level 0 products, SeaStar SeaWiFS Level 1A products, etc.
SAFE Specialisation Control Book	A book which defines the SAFE Specialisation for one or more products.

5.3. Acronyms and Abbreviations

ASAR	Advanced Synthetic Aperture Radar (an instrument of ENVISAT platform).
AMI	Active Microwave Instrument (an instrument of ERS platforms).
CCSDS	Consultative Committe for Space Data Systems
ENVISAT	ENVIronment SATellite platform
ENVISAT - TIM Auxiliary Product	ENVIronment SATellite - UTC Time Reference and Conversion Table Auxiliary Product
ERS	European Remote Sensing Satellite(s)
ESA	European Space Agency
GOMOS	Global Ozone Measurement by the Occultation of Stars (an instrument of ENVISAT platform).
Landsat	A program of seven platforms
MSS	Multispectral Scanner (an instrument of Landsat 1 to 5 platforms).
OAIS	Reference Model for an Open Archival Information System
SAFE	Standard Archive Format for Europe (SAFE)
SAR	Synthetic Aperture Radar

SDF	Strutured Data File Markup Language
SeaStar	Spacecraft flying SeaWiFS
SeaWiFS	Sea-viewing Wide Field-of-view Sensor (an instrument of SeaStar platform).
UML	Unified Modeling Language
UTC	Universal Time Coordinated
XFDU	XML Fomatting Data Unit
XML	eXtensible Markup Language

Chapter 1. SAFE Specialisation Overview

1.1. What is a SAFE Specialisation?

The [SAFE-CORE] defines that:

> SAFE has been designed to “be” an instance of XFDU, restricting the generic areas of XFDU to the specific needs of Earth Observation ground segments and to provide semantic in the same domain for improving the interoperability between the ground segment facilities.

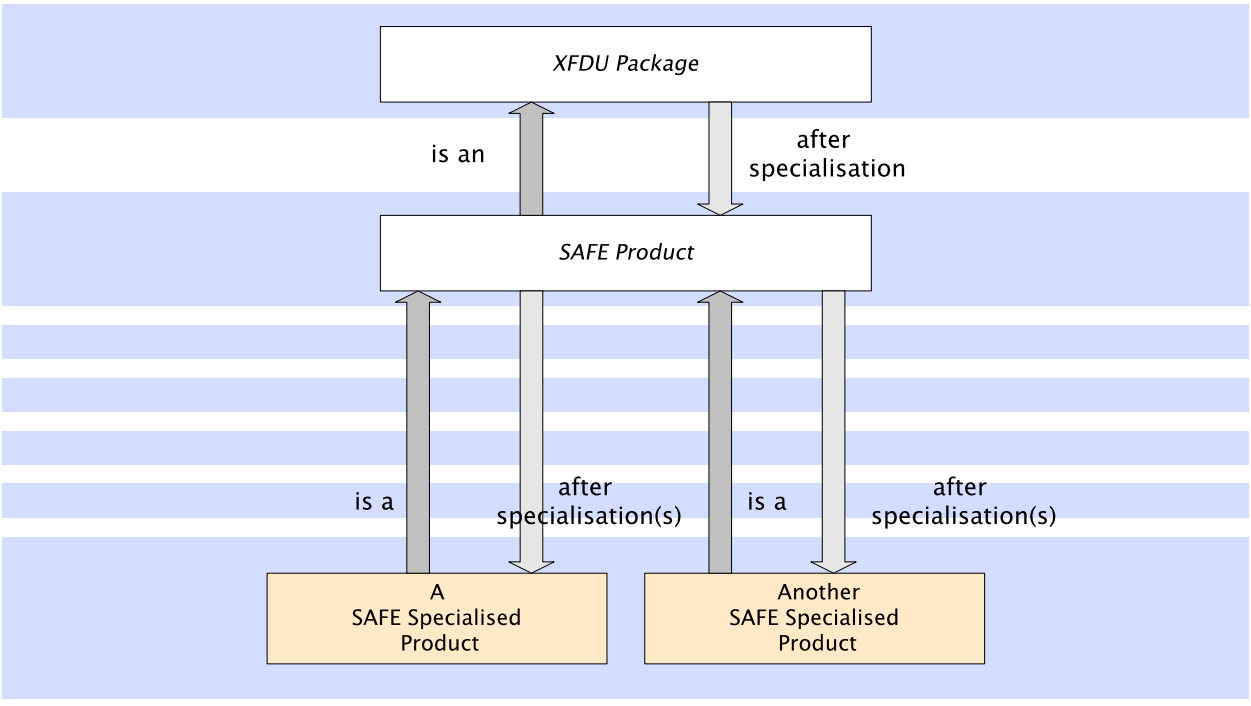
The same principle *must* apply for each specialisation of SAFE:

Any SAFE Specialisation *must* “be” an instance of SAFE.

The [SAFE-CORE] defines that:

> A critical characteristic of SAFE is that “any SAFE Product is an XFDU Package”.

In the same way, a critical characteristic of SAFE Specialisation rules is that “any SAFE Product defined by a SAFE Specialisation is a SAFE Product defined by the [SAFE-CORE]”.



SAFE Specialisation Definition

A SAFE Specialisation can be either a SAFE Abstract Specialisation, a SAFE Auxiliary Specialisation or a SAFE Product Specialisation.

To be complete, a SAFE Specialisation Specification shall respect the following requirements:

- The SAFE Product defined by a SAFE Specialisation shall be exhaustively defined in a SAFE Specialisation Control Book (SAFE Manifest and all Components content);
- Representation Information of every Component of the SAFE Product shall be described by an XML Schema following the rules defined in the [SAFE-CORE].
- The SAFE Manifest of the SAFE Product shall be validated by an xfdu.xsd XML Schema (strict *restriction* i.e. in the sense of XML Schema recommendation [XML-SCHEMA-STRUCT] of the XML Schema provided in the Appendix A of the [SAFE-CORE]);

The SAFE Manifest or the entire SAFE Product can be validated by any other method than the xfdu.xsd XML Schema validation, if the validation is more accurate.

1.2. SAFE Specialisation Control Books

A SAFE Specialisation Control Book is a book which defines the SAFE Specialisation(s) for one or more product(s). These products shall share a strong common point (for example products processed from the same platform, the same instrument, etc.).

1.2.1. Content of Specialisation Control Books

A SAFE Specialisation Control Book shall:

- Define the precise list of all components the SAFE Product(s) gather(s) (Binary, ASCII, XML, XML Schema, etc. Components);
- Define the precise list of all XML Schemas used for the validation of the SAFE Manifest of the SAFE Product(s);
- Provide in appendix the entire set of XML Schemas defined for the SAFE Specialisation(s);
- Provide the precise description of the SAFE Manifest content of the SAFE Product(s). This description includes XFDU Types, SAFE Types, New Specific Types, element occurrences, mandatory pattern values, namespaces, etc.

An example of SAFE Manifest for every SAFE Product should be provided in appendix.

The thorough description of Data and Metadata Component(s) of the SAFE Product(s) *can* be provided (since SAFE is an Archive Format, the SAFE Specialisation Control Book *does not* replace the previous product specifications).

An overview of the product(s), instrument(s), platform(s), etc. can be provided.

An example of precise description of the SAFE Manifest content of a SAFE Product (Landsat MSS Level 0 Product) follows:

ENTITY / ATTRIBUTE	VALUE	OCC.
`informationPackageMap`		

ENTITY / ATTRIBUTE	VALUE	OCC.
. xfdu:contentUnit		1
. . *@ID*	“packageUnit”	0..1
. . *@textInfo*		0..1
. . *@pdiID*	“processing”	1
. . *@dmdID*	In any order: “acquisitionPeriod” “platform”	1
. . xfdu:contentUnit		1
. . . *@ID*	“measurementUnit”	0..1
. . . *@textInfo*		0..1
. . . *@dmdID*	In any order: “measurementOrbitReference” (mandatory) “measurementIndex” (mandatory) “measurementQualityInformation” (non-mandatory)	1
. . . *@repID*	In any order: “measurementSchema” “landsatMeasurementSchema”	1
. . . dataObjectPointer		1
. . . . *@ID*		0..1
. . . . *@dataObjectID*	“measurementData”	1
. . xfdu:contentUnit		1
. . . *@ID*	“measurementIndexUnit”	0..1
. . . *@textInfo*		0..1
. . . *@repID*	“measurementIndexSchema”	1
. . . dataObjectPointer		1
. . . . *@ID*		0..1
. . . . *@dataObjectID*	“measurementIndexData”	1

Table 1.1. Landsat MSS Level 0 Specialisation informationPackageMap Entity

1.2.2. Development and Maintenance

SAFE is a long-term preservation format. Accuracy and stability in time of a SAFE Specialisation Control Book is a *critical* need.

For the creation of a SAFE Specialisation Control Book, it is *very strongly* recommended to use advanced technologies in order to avoid any integrity error in the definition of a SAFE Specialisation.

Exemple of technologies used to create the entire set of ESA official SAFE Specialisation Control Books as the [SAFE-CORE] are:

- DocBook;

- XSLT;
- XSL-FO;
- XQuery;
- Java;
- Maven;
- etc.

Chapter 2. Product Structure

2.1. Introduction

A SAFE Product contains information. Every information is a “Data”. Some “Data” are mesured by an instrument, some are calculated by a processor, some are added by a man. SAFE classifies information in two categories: *Data* and *Metadata*.

For a SAFE Product, Data is:

“All stored information, cause of the existence of the product”.

For a SAFE Product, Metadata is:

“All other stored information”.

Such different information as the platform identification, a Component index, a Component Representation Information are considered as Metadata.

2.2. Data

SAFE is designed to manage any Data. A SAFE Product can gather only one data file or multiple data files. Cause of multiple files can be:

- there are “meaning-different” data files
- a single file has been split into several ones.

Although the SAFE primary goal is to handle product levels close to the usually called “level 0”, no particular limitation exists regarding the handling of higher level products.

Even if each SAFE Specialisation shall be different, the following recommendation can be provided.

- For Level 0 Products, there shall be only one Data Component.
- For higher than Level 0 Products, there shall be as many Data Components as required.

For higher than Level 0 Products, number of Data Components shall be defined for each SAFE Product depending on the nature of the Data.

A little file size for a Data Component should not be a reason for wrapping the Data instead of reference it (to wrap Data is anyway not allowed by SAFE, only Metadata can be wrapped).

2.3. Metadata

2.3.1. Mandatory Metadata

As defined in the [SAFE-CORE], any SAFE Product must contain at least two Metadata Objects (dedicated to the processing history and to platform information).

`safe:platform` element definition shall be as restricted as possible. An example of complete restriction follows:

```
[...]

   <xs:complexType name="platformType">
      <xs:complexContent>
         <xs:restriction base="safe:platformType">
            <xs:sequence>
               <xs:element name="nssdcIdentifier"
                           type="safe:nssdcIdentifierType"/>
               <xs:element name="familyName"
                           type="safe:platformFamilyNameType"/>
               <xs:element name="number"
                           type="safe:platformNumberType"/>
               <!-- instrument element mandatory -->
               <xs:element name="instrument"
                           type="safe:instrumentType"/>
               <!-- Wild Card not allowed -->
            </xs:sequence>
         </xs:restriction>
      </xs:complexContent>
   </xs:complexType>

   <xs:simpleType name="platformFamilyNameType">
      <xs:restriction base="safe:platformFamilyNameType">
         <xs:enumeration value="Landsat"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="nssdcIdentifierType">
      <xs:restriction base="safe:nssdcIdentifierType">
         <xs:enumeration value="1972-058A"/>
         <xs:enumeration value="1975-004A"/>
         <xs:enumeration value="1978-026A"/>
         <xs:enumeration value="1982-072A"/>
         <xs:enumeration value="1984-021A"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="platformNumberType">
      <xs:restriction base="safe:platformNumberType">
         <xs:enumeration value="1"/>
         <xs:enumeration value="2"/>
         <xs:enumeration value="3"/>
         <xs:enumeration value="4"/>
         <xs:enumeration value="5"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="completeInstrumentFamilyNameType">
      <xs:restriction base="safe:completeInstrumentFamilyNameType">
         <xs:enumeration value="Multispectral Scanner"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="abbreviationType">
      <xs:restriction base="safe:abbreviationType">
         <xs:enumeration value="MSS"/>
      </xs:restriction>
   </xs:simpleType>

[...]
```

Example 2.1. Restriction of safe:platform for Landsat MSS Level 0 Products

2.3.2. Wrapped Metadata or Metadata Component?

SAFE allows to wrap Metadata and to reference any number of Metadata Components. The [SAFE-CORE] provides several type definitions. Regarding Metadata which doesn't match the definition of any SAFE Type, the following recommendation can be provided.

- Metadata repeated such as records shall be stored as Metadata Components and referenced by the SAFE Manifest.
- Any other Metadata shall be wrapped into the SAFE Manifest.

A little file size for a Metadata Component should not be a reason for wrapping the Metadata instead of reference it.

2.3.3. Data Index

Each Data Component can be completed by an index for access efficiency. Nevertheless it is recommended to be very careful to the Data Component file size before adding an index to a Data Component. SAFE is a long-term preservation format; a today big file size could become a tomorrow litte file size.

2.3.4. Representation Information

SAFE defines that every Component (except XML Schema Component) must be accompanied by a Representation Information (standard XML Schemas). There can be more than one XML Schema for the Representation Information of a Component. Each XML Schema shall be an XML Schema Component.

2.4. Case of Auxiliary Products

Although the SAFE primary goal is to handle product levels close to the usually called “level 0”, no particular limitation exists regarding the handling of Auxiliary Products.

Auxiliary Products contain Data which are Metadata for other products. In fact, auxiliary products “are Metadata products for products”. Data stored in an auxiliary product are sometimes present in another product as Metadata.

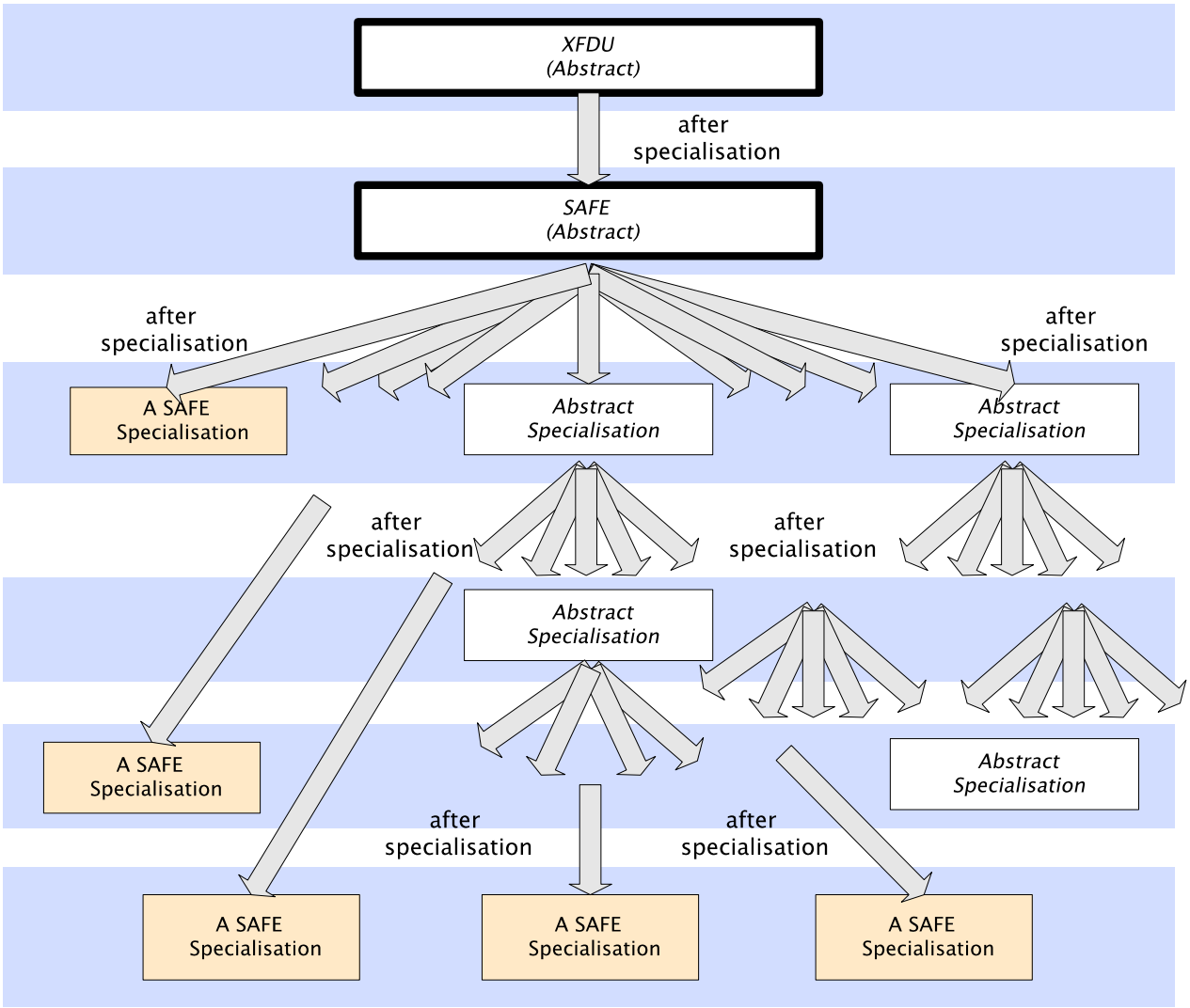
Even if there can be a confusion between the nature of Data of an auxiliary data (Data/Metadata), the following recommendation can be provided.

For a product (auxiliary or not), Data is “all stored information, cause of the existence of the product”.

For example, for an ENVISAT - TIM Auxiliary Product, Data is the UTC Time Reference, the satellite binary clock and the clockstep (for a given time validity). Even if these Data are considered “Metadata” for a Telemetry Product, for a ENVISAT - TIM Auxiliary Product it's Data.

Chapter 3. Manifest Types

3.1. Abstract Specialisations



Abstract Specialisations

As said in Section 1.2, some SAFE Products share some strong common points. As an example, for ENVISAT platform there are 25 different Level 0 Products.

Some Manifest Types do not need to be redefined and restricted as many times as number of products. As an example, the following redefinition of `platformFamilyNameType` shall restrict the authorised `xs:string` to value “ENVISAT” once for all ENVISAT products:

```
<xs:simpleType name="platformFamilyNameType">
   <xs:restriction base="safe:platformFamilyNameType">
```

```
        <xs:enumeration value="ENVISAT"/>
      </xs:restriction>
    </xs:simpleType>
```

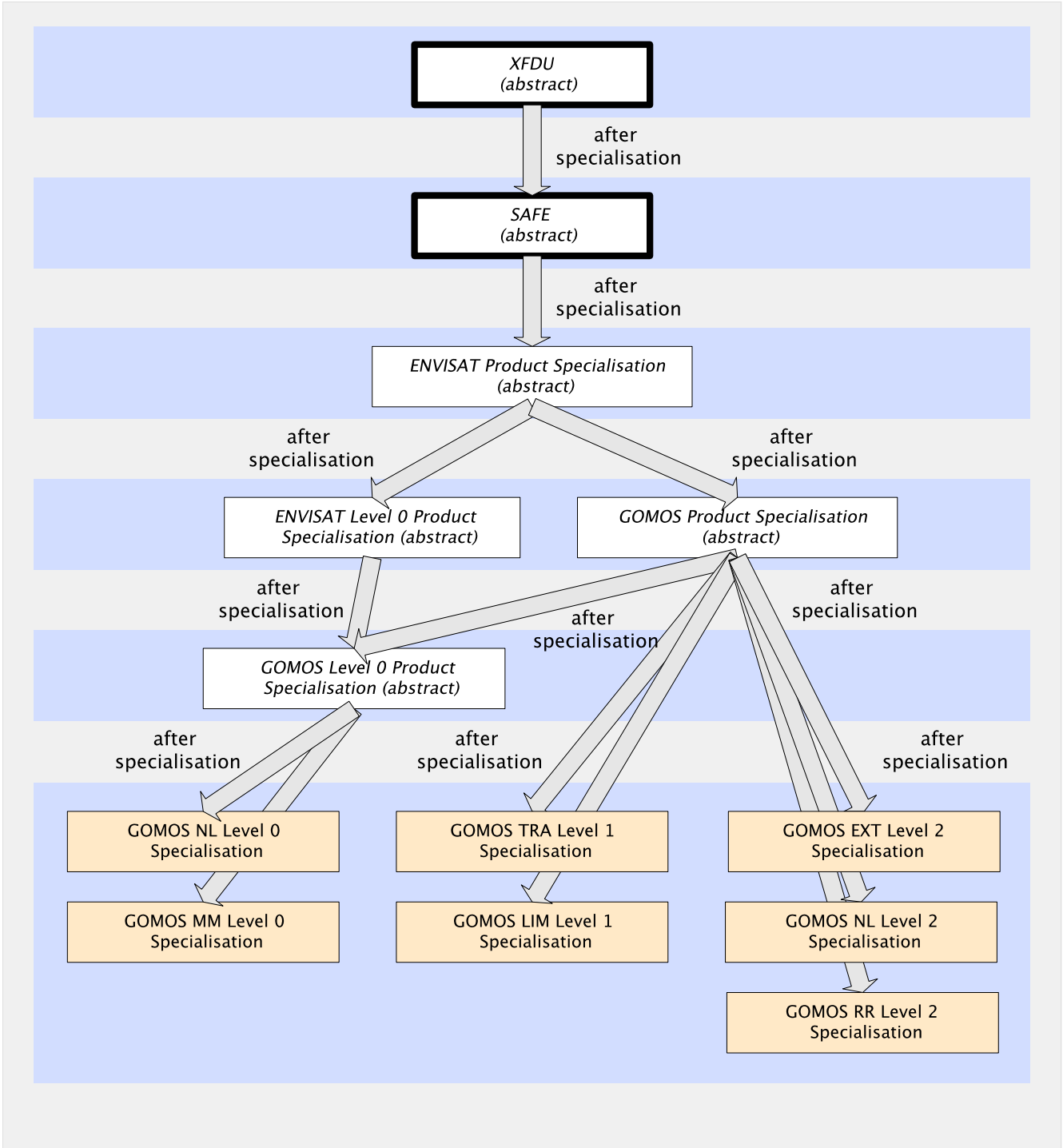
Example 3.2. Restriction of safe:platformFamilyNameType for ENVISAT Products

Redefinition and restriction of Manifest Types:

- shall be done at the higher possible level (possibly by a SAFE Abstract Specialisation);
- can be inherited from a SAFE Abstract Specialisation by one or more SAFE Specialisation(s) (SAFE Abstract Specialisation, SAFE Auxiliary Specialisation or SAFE Product Specialisation).

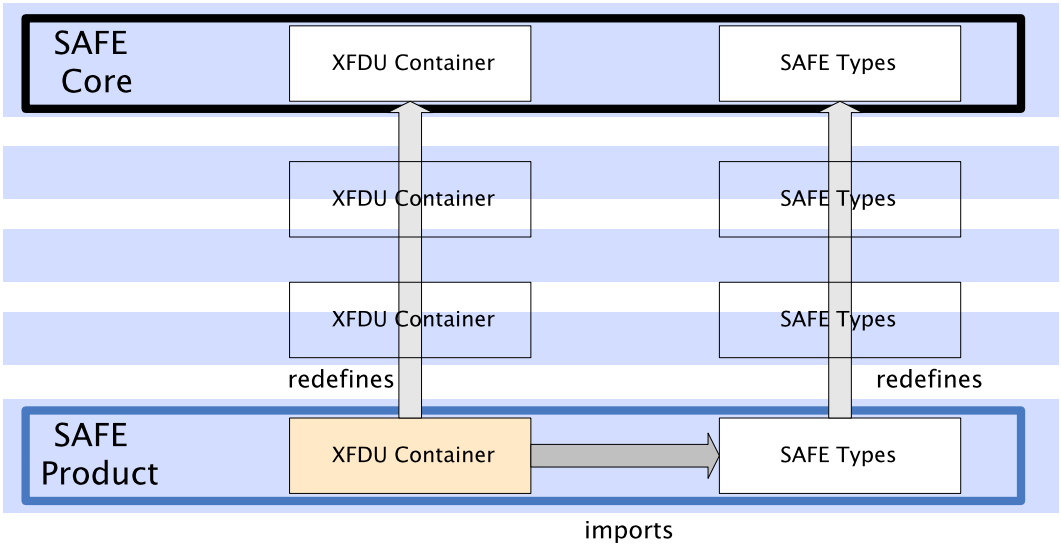
A SAFE Specialisation can inherit types from one or more SAFE Abstract Specialisations.

The following example describes the SAFE Specialisation for ENVISAT GOMOS tree:



Example 3.3. SAFE Specialisations for ENVISAT GOMOS Products

3.2. XML Schema Mechanisms



Redefinition and Import of Manifest Types

```
<xs:schema xmlns:xs       ="http://www.w3.org/2001/XMLSchema"
           xmlns:safe     ="http://www.esa.int/safe/1.3"
           xmlns:xfdu     ="urn:ccsds:schema:xfdu:1"
           targetNamespace="urn:ccsds:schema:xfdu:1"

           elementFormDefault ="qualified"
           attributeFormDefault="unqualified">

   <xs:import namespace="http://www.esa.int/safe/1.3"
              schemaLocation="safe.xsd"/>

   <xs:redefine schemaLocation="xfdu.xsd">

   <xs:complexType name="xmlDataType">
      <xs:complexContent>
         <xs:restriction base="xfdu:xmlDataType">
            <xs:choice>
               <xs:element ref="safe:processing"
                           minOccurs="1" maxOccurs="unbounded"/>
               <xs:element ref="safe:platform"/>
               <xs:element ref="safe:acquisitionPeriod"/>
               <xs:element ref="safe:orbitReference"/>
               <xs:element ref="safe:gridReference"/>
               <xs:element ref="safe:qualityInformation"/>
            </xs:choice>
         </xs:restriction>
      </xs:complexContent>
   </xs:complexType>

   </xs:redefine>

</xs:schema>
```

Example 3.4. Redefinition and Import of Manifest Types

3.2.1. Use of xs:redefine

SAFE is defined by:

- an xfdu.xsd XML Schema, provided in the Appendix A of the [SAFE-CORE] .
- a safe.xsd XML Schema, provided in the Appendix B of the [SAFE-CORE] .
- an active-sensor-types.xsd XML Schema, provided in the Appendix C of the [SAFE-CORE] .
- an index.xsd XML Schema, provided in the Appendix E of the [SAFE-CORE] .

As said in Chapter 1, any SAFE Specialisation *must* "be" an instance of SAFE.

To restrict SAFE (and its XML Schemas), it is recommended to use the `xs:redefine` mechanism.

This mechanism is intended to provide a declarative and modular approach to schema modification.

The definitions within the `xs:redefine` element itself are restricted to be redefinitions of types from the redefined XML schema document, in terms of themselves. Type definitions must use themselves as their base type definition.

XML Schema recommendations allow type redefinition either by restriction or extension (`xs:restriction` or `xs:extension`). According to SAFE specialisation rules, it is *very strongly* recommended:

Redefinition of Manifest Types shall be done by restriction (`xs:restriction`). Use of `xs:extension` can *not* guarantee a SAFE Specialisation to "be" an instance of SAFE.

`xs:any` Wild Cards have been added to SAFE Types. These Wild Cards allow addition of New Specific Types.

3.2.2. Use of xs:import

SAFE uses several namespaces. In particular, the "container" XFDU Types and the "contained" SAFE Types have two different namespaces.

The SAFE Manifest of a SAFE Product is an "XFDU container"; its namespace is `urn:ccsds:schema:xfdu:1`. According to the XML Schema recommendations, the only method to allow SAFE Types (which are qualified) inside XFDU Types is to use `xs:import`.

The only method to allow New Specific Types (defined and provided by a SAFE Specialisation) as part of XFDU Types and/or SAFE Types is to use `xs:import` too.

3.2.3. Use of xs:include

Use of `xs:include` for specialising SAFE Types and XFDU Types is neither recommended, nor forbidden. From our experience (specialisation of 49 SAFE Products), `xs:include` mechanism is simply not interesting.

`xs:include` mechanism *can* be useful for New Specific Types (management of XML schemas), for types qualified with the same namespace.

3.2.4. Global or Local Types?

XML Schema recommendations allow to declare types and elements. Declared types are global types, i.e. modifiables. Declared elements can be local typed, i.e. not modifiables. The following examples illustrate the two methods:

```
<xs:element name="familyName"
            type="safe:platformFamilyNameType"

<xs:simpleType name="platformFamilyNameType">
   <xs:restriction base="xs:string"/>
</xs:simpleType>
```

Example 3.5. Declaration of a global type

```
<xsd:element name="familyName" type="xs:string"/>
```

Example 3.6. Declaration of an element of local type

For Manifest Types, it is *strongly* recommended to declare global types and use these to declare elements (as declared within the Example 3.5).

This method allows redefinition by restriction of types and, consequently, of elements of those latter.

3.2.5. XML Schema Validation Limit

XML Schema role is mainly to control and validate the *structure* of an XML file and *not* its content. Even if XML Schema can also be used to control and validate *some* fields content (by use of `xs:enumeration` or `xs:pattern`), for very open XML Schemas like safe.xsd and xfdu.xsd this validation is incomplete.

For a fully complete and guaranteed validation, use of specific tools is needed (Schematron, XQuery, etc.).

3.3. XFDU Types

3.3.1. Restriction

SAFE provides XFDU Types. Even if a SAFE Product is able to contain many information (for example an unbounded number of Data Objects, an unbounded number of Metadata Objects, etc.), every product contains a fixed quantity of Data and Metadata.

```
An xfdu:dataObjectSectionType allows from one to unbounded number of
dataObjects:

dataObject (from 1 to n);

For an ENVISAT GOMOS NL level 0 product, there is a single Data Object
holding recorded measurements:

dataObject (1);

ENVISAT GOMOS NL level 0 Specialisation shall restrict the
xfdu:dataObjectSectionType in order to avoid any case of validating
an ENVISAT GOMOS NL level 0 product holding more than a single Data Object:

<xs:complexType name="dataObjectSectionType">
   <xs:complexContent>
      <xs:restriction base="xfdu:dataObjectSectionType">
         <xs:sequence>
            <xs:element name="dataObject" type="xfdu:dataObjectType"
                        form="unqualified"
                        minOccurs="1" maxOccurs="1"/>
         </xs:sequence>
      </xs:restriction>
   </xs:complexContent>
</xs:complexType>

The ENVISAT GOMOS NL level 0 product single Data Object holds recorded
measurements:

Data Object = Measurements

ENVISAT GOMOS NL level 0 Specialisation shall restrict the
xfdu:dataObjectSectionType in order to:
- avoid any case of validating an ENVISAT GOMOS NL level 0 product
holding other Data Object than measurements;
- avoid any case of validating an ENVISAT GOMOS NL level 0 product
holding a Data Object without a Representation Information:

<xs:complexType name="dataObjectType">
   <xs:complexContent>
      <xs:restriction base="xfdu:dataObjectType">
         <xs:sequence>
            <xs:element name="byteStream" type="xfdu:byteStreamType"
                        form="unqualified"/>
         </xs:sequence>
         <xs:attribute name="ID" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:ID">
                  <xs:enumeration value="measurementData"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="repID" type="xs:IDREFS" use="required"/>
      </xs:restriction>
   </xs:complexContent>
```

```
    </xs:complexType>
```

Example 3.7. Why Restrict XFDU Types?

The following recommendation can be provided:

If possible, value and occurrences shall always be restricted.

3.3.2. Use of Content Units

The Information Package Map (`xfdu:informationPackageMapType`) outlines a hierarchical structure for the original object being encoded, by a series of nested contentUnit elements [XFDU].

SAFE provides an `informationPackageMap` that can contain only one `contentUnit`: this single `contentUnit` is *a view* of the SAFE Product.

SAFE provides a `contentUnit` that can contain none or one `dataObjectPointer`, and none, one or more `contentUnit`(s):

- each `contentUnit` (except the "Root" `contentUnit` i.e. the unique sub-element of `informationPackageMap`) shall be *a view* of a SAFE Object. This *does not* mean that "each SAFE Object has a dedicated `contentUnit`".
- for each Data Component, there shall be a `contentUnit`;
- for each Metadata Component (except XML Schema Component), there shall be a `contentUnit`;
- for each `contentUnit`, *view* of a SAFE Object composed of a Data Component or a Metadata Component (except XML Schema Component):
  - the `contentUnit` shall have a `dataObjectPointer` sub-element pointing on a `dataObject` referencing the Data Component or the Metadata Component;
  - as each Data Component and Metadata Component (except XML Schema Component) *must* have a Representation Information, the `contentUnit` shall hold a `repID` attribute pointing on one or more `metadataObject`(s) referencing an XML Schema Component;

SAFE provides an `ID` attribute to `contentUnit`. Its use is recommended, even if not mandatory. `contentUnit` element has also a `unitType` attribute: its use is not recommanded, since it is not a perfect identifier (even if not prohibited).

As an example of use of Content Units, a SAFE Manifest for ERS AMI SAR Level 0 Product is provided in Appendix A.

3.3.3. Relationship between metadataObject and dataObject

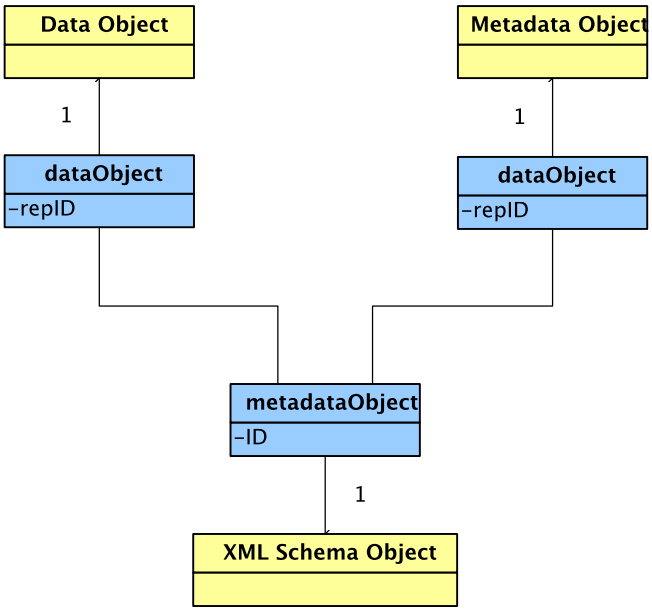
As introduced in the Section 2.1, classification of information as Data or Metadata is delicate. Classifying two information, one as Data, the other as Metadata, it's applying a *view* on both information.

XFDU (and so SAFE) provides two kinds of "objects-types": `dataObjectType` and `metadataObjectType`. As defined in the [SAFE-CORE] :

- `metadataObject` is always part of a Metadata Object (never part of a Data Object);
- `dataObject` is always part of a Data Object *or can be part of a Metadata Object*.

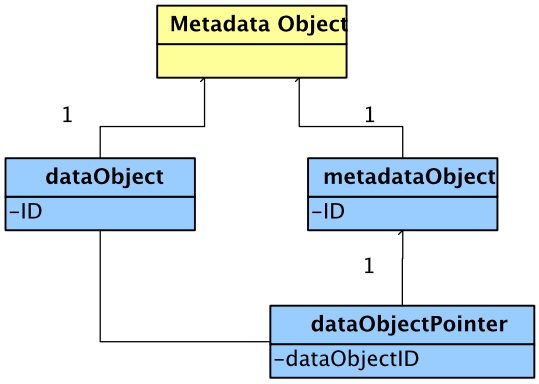
A dataObject and a metadataObject shall be linked together in the following cases:

- for the `metadataObject`, part of an XML Schema Object, `ID` attribute shall be pointed by the `repID` attribute hold by the `dataObject` part of the associated Data Object or Metadata Object;



metadataObject and dataObject Linking

- for the `dataObject`, part of a Metadata Object, `ID` attribute shall be pointed by the `dataObjectID` attribute of `dataObjectPointer` sub-element of the `metadataObject`.



metadataObject and dataObject Linking

3.4. SAFE Types

3.4.1. Restriction

SAFE provides SAFE Types. Even if a SAFE Type is able to contain many information, for some products some of the information are not available.

```
For safe:orbitReferenceType, the following sub-elements are provided:

Orbit Number (from 0 to 2);
Relative Orbit Number (from 0 to 2);
Cycle Number (from 0 to 1);
Phase Identifier (from 0 to 1);
Start Track (from 0 to 1);
Stop Track (from 0 to 1);
Any Information (Wild Card) (from 0 to unbounded).

If a product contains information about:
- Orbit Number (1)
- Relative Orbit Number (1)
- Cycle Number (1)
- Phase Identifier (1)
and does not contain any other orbit information, the
safe:orbitReferenceType shall be restricted in order to avoid any
case of misinformation:

<xs:complexType name="orbitReferenceType">
   <xs:complexContent>
      <xs:restriction base="safe:orbitReferenceType">
         <xs:sequence>
            <xs:element name="orbitNumber"
                        type="safe:orbitNumberType"
                        minOccurs="1" maxOccurs="1"/>
            <xs:element name="relativeOrbitNumber"
                        type="safe:relativeOrbitNumberType"
                        minOccurs="1" maxOccurs="1"/>
            <xs:element name="cycleNumber"
                        type="safe:cycleNumberType"
                        minOccurs="1" maxOccurs="1"/>
            <xs:element name="phaseIdentifier"
                        type="safe:phaseIdentifierType"
                        minOccurs="1" maxOccurs="1"/>
         </xs:sequence>
      </xs:restriction>
   </xs:complexContent>
</xs:complexType>
```

Example 3.8. Why Restrict SAFE Types?

The following recommendation can be provided:

If possible, value and occurrences shall always be restricted.

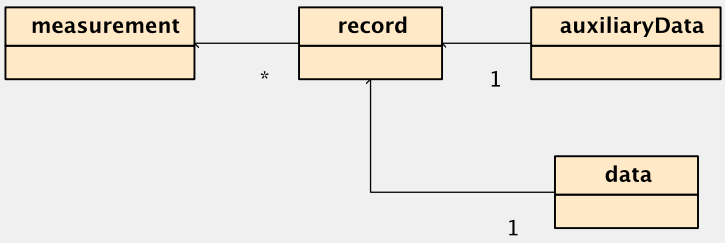
3.4.2. Use of safe:qualityInformationType

The [SAFE-CORE] provides the definition of `safe:qualityInformationType`.

`safe:qualityInformationType` has been designed in order to list all missing and corrupted units detected in a product. Role of a Quality Information Metadata Object is also to list all the parts of a product checked without detected error.

SAFE provides many elements and attributes for the listing of missing and corrupted units (`safe:location/safe:path`, `safe:location/safe:time`, elements, `following`, `after`, `preceding` and `before` attributes...). Even if it is possible to freely use `safe:missingElementsType` and `safe:corruptedElementsType`, some recommendations can be provided with plenty of examples.

An example of a product is provided in order to illustrate following recommendations. The product (with a `measurement` root node) is a concatenation of `records`. Each `record` has two mandatory sub-elements: `auxiliaryData` and `data` (each one is a single unsignedByte element).



Example 3.9. Logical view of a Component

```
<xs:element name="measurement">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="record" minOccurs="0" maxOccurs="unbounded">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="auxiliaryData" type="xs:unsignedByte"/>
                  <xs:element name="data"          type="xs:unsignedByte"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
</xs:element>
```

Example 3.10. XML Schema, Representation Information of a Component

3.4.2.1. Corrupted and Missing Units

One thing important to understand is that:

- the corrupted units are present in the product but their value are wrong;
- the missing units are *not* present in the product while they should.

This leads to the possibility for corrupted units to locate them using only an XPath, while it's impossible to locate the missing units using only an XPath since the missing units do not exist in the product. For missing units use of safe:path attributes and/or safe:count element is mandatory.

For example, if a physical product contains 10 records and it has been detected that there are 5 missing records, it means that the product should contain 15 records. safe:path attributes and/or safe:count will allow to locate the 5 “non-present” records.

3.4.2.2. Use of safe:missingElementsType

3.4.2.2.1. Cases of Unknown Location

3.4.2.2.1.1. Missing Units - Unknown Location

If missing units have been detected, but their location are completly unknown, a `safe:missingElements` element shall be created using XPath of `safe:location/safe:path` element and `safe:count` element.

```
<safe:missingElements>
   <safe:location>
      <safe:path>
         measurement/record
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:missingElements>
```

Example 3.11. 3 Detected Missing Records - Unknown Location

3.4.2.2.1.2. Missing Units - Unknown Location (but beginning of the product is known and correct)

If missing units have been detected, but their location are partially unknown (the beginning of the product is known and correct), a `safe:missingElements` element shall be created using XPath and `after` attribute of `safe:location/safe:path` element, and `safe:count` element.

```
<safe:missingElements>
   <safe:location>
      <safe:path after="100">
         measurement/record
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:missingElements>
```

Example 3.12. 3 Detected Missing Records - Unknown Location (but we know that the records #1 to #100 are present)

3.4.2.2.1.3. Missing Units - Unknown Location (but end of the product is known and correct)

If missing units have been detected, but their location are partially unknown (the end of the product is known and correct), a `safe:missingElements` element shall be created using XPath and `before` attribute of `safe:location/safe:path` element, and `safe:count` element.

```
<safe:missingElements>
   <safe:location>
      <safe:path before="100">
         measurement/record
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:missingElements>
```

Example 3.13. 3 Detected Missing Records - Unknown Location (but we know that after the record #100, all records are present)

3.4.2.2.1.4. Missing Units in a Segment

If missing units have been detected in a segment of units, a `safe:missingElements` element shall be created using XPath, `after` and `before` attributes of `safe:location/safe:path` element, and `safe:count` element.

```
<safe:missingElements>
```

```
      <safe:location>
         <safe:path after="200" before="300">
            measurement/record
         </safe:path>
      </safe:location>
      <safe:count value="3"/>
   </safe:missingElements>
```

Example 3.14. 3 Detected Missing Records (located between records #200 to #300)

3.4.2.2.2. Cases of Known Location

3.4.2.2.2.1. Missing Unit(s) (neither first nor last elements of a sequence)

If missing units have been detected and located (neither first nor last elements of a sequence), a `safe:missingElements` element shall be created using XPath, `following` *or/and* `preceding` attributes of `safe:location/safe:path` element, and `safe:count` element.

```
   <safe:missingElements>
      <safe:location>
         <safe:path following="3">
            measurement/record
         </safe:path>
      </safe:location>
      <safe:count value="3"/>
   </safe:missingElements>
```

Example 3.15. 3 Detected Missing Records (records #4, #5, #6): first case

```
   <safe:missingElements>
      <safe:location>
         <safe:path preceding="4">
            measurement/record
         </safe:path>
      </safe:location>
      <safe:count value="3"/>
   </safe:missingElements>
```

Example 3.16. 3 Detected Missing Records (records #4, #5, #6): second case

```
<safe:missingElements>
   <safe:location>
      <safe:path following="3" preceding="4">
         measurement/record
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:missingElements>
```

Example 3.17. 3 Detected Missing Records (records #4, #5, #6): third case

3.4.2.2.2.2. Missing Unit(s) (first elements of a sequence)

If missing units (first elements of a sequence) have been detected, a `safe:missingElements` element shall be created using XPath, `preceding` attribute of `safe:location/safe:path` element, and `safe:count` element.

```
<safe:missingElements>
   <safe:location>
      <safe:path preceding="1">
         measurement/record
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:missingElements>
```

Example 3.18. 3 Detected Missing Records (records #1, #2, #3)

3.4.2.2.2.3. Missing Unit(s) (last elements of a sequence)

If missing units (last elements of a sequence) have been detected, a `safe:missingElements` element shall be created using XPath, `following` attribute of `safe:location/safe:path` element, and `safe:count` element.

```
<safe:missingElements>
   <safe:location>
      <safe:path following="99">
         measurement/record
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:missingElements>
```

Example 3.19. 3 Detected Missing Records (records #100, #101, #102 should be present - the product last record is the #99)

3.4.2.2.2.4. Time Located Missing Unit(s)

If missing units have been detected and time located, a `safe:missingElements` element shall be created using `safe:location/safe:time/safe:start` and `safe:location/safe:time/safe:stop` elements (and `safe:location/safe:path` element since the product can gather more than one Data Object).

```
<safe:missingElements>
   <safe:location>
      <safe:time>
         <safe:start>2004-12-16T23:55:53.000000Z</safe:start>
         <safe:stop>2004-12-16T23:55:57.55555Z</safe:stop>
      </safe:time>
      <safe:path>
         measurement/record
      </safe:path>
   </safe:location>
</safe:missingElements>
```

Example 3.20. Detected Missing Records (between 2004-12-16T23:55:53.000000Z and 2004-12-16T23:55:57.55555Z)

3.4.2.3. Use of safe:corruptedElementsType

3.4.2.3.1. Cases of Unknown Location

3.4.2.3.1.1. Corrupted Units - Unknown Location

If corrupted units have been detected, but their location are completly unknown, a

`safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and `safe:count` element.

```
<safe:corruptedElements>
   <safe:location>
      <safe:path>
         measurement/record/data
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:corruptedElements>
```

Example 3.21. 3 Detected Corrupted data elements - Unknown Location

3.4.2.3.1.2. Corrupted Units - Unknown Location (but beginning of the product is known and correct)

If corrupted units have been detected, but their location are partially unknown (the beginning of the product is known and correct), a `safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and `safe:count` element.

```
<safe:corruptedElements>
   <safe:location>
      <safe:path>
         <![CDATA[measurement/record[fn:position() > 100]/data]]>
      </safe:path>
   </safe:location>
   <safe:count value="3"/>
</safe:corruptedElements>
```

Example 3.22. 3 Detected Corrupted data elements - Unknown Location (but we know that the records #1 to #100 are correct)

3.4.2.3.1.3. Corrupted Units - Unknown Location (but end of the product is known and correct)

If corrupted units have been detected, but their location are partially unknown (the end of the product is known and correct), a `safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and `safe:count` element.

```
<safe:corruptedElements>
   <safe:location>
      <safe:path>
```

```
            <![CDATA[measurement/record[fn:position() < 101]/data]]>
         </safe:path>
      </safe:location>
      <safe:count value="3"/>
   </safe:corruptedElements>
```

Example 3.23. 3 Detected Corrupted data elements - Unknown Location (but we know that the records #101 to the end of the product are correct)

3.4.2.3.1.4. Corrupted Units in a Segment

If corrupted units have been detected in a segment of units, a `safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and `safe:count` element.

```
   <safe:corruptedElements>
      <safe:location>
         <safe:path>
             <![CDATA[measurement/record[fn:position() > 200
                and fn:position() < 300]/data]]>
         </safe:path>
      </safe:location>
      <safe:count value="3"/>
   </safe:corruptedElements>
```

Example 3.24. 3 Detected Corrupted data elements (located between records #200 to #300)

3.4.2.3.2. Cases of Known Location

3.4.2.3.2.1. Corrupted Unit(s) (neither first nor last elements of a sequence)

If corrupted units have been detected and located (neither first nor last elements of a sequence), a `safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and *non-mandatory* `safe:count` element.

```
   <safe:corruptedElements>
      <safe:location>
         <safe:path>
             <![CDATA[measurement/record[fn:position() > 3
                and fn:position() < 7]/data]]>
         </safe:path>
      </safe:location>
      <safe:count value="3"/> non mandatory
   </safe:corruptedElements>
```

Example 3.25. 3 Detected Corrupted data elements (data of records #4, #5, #6)

3.4.2.3.2.2. Corrupted Unit(s) (first elements of a sequence)

If corrupted units (first elements of a sequence) have been detected, a `safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and *non-mandatory* `safe:count` element.

```
<safe:corruptedElements>
   <safe:location>
      <safe:path>
          <![CDATA[measurement/record[fn:position() < 4]/data]]>
      </safe:path>
   </safe:location>
   <safe:count value="3"/> non mandatory
</safe:corruptedElements>
```

Example 3.26. 3 Detected Corrupted data elements (data of records #1, #2, #3)

3.4.2.3.2.3. Corrupted Unit(s) (last elements of a sequence)

If corrupted units (last elements of a sequence) have been detected, a `safe:corruptedElements` element shall be created using XPath of `safe:location/safe:path` element and *non-mandatory* `safe:count` element.

```
<safe:corruptedElements>
   <safe:location>
      <safe:path>
          <![CDATA[measurement/record[fn:position() > 99]/data]]>
      </safe:path>
   </safe:location>
   <safe:count value="3"/> non mandatory
</safe:corruptedElements>
```

Example 3.27. 3 Detected Corrupted data elements (data of records #100, #101, #102 - the product last record is the #102)

3.4.2.3.2.4. Time Located Corrupted Unit(s)

If corrupted units have been detected and time located, a `safe:corruptedElements` element shall be created using `safe:location/safe:time/safe:start` and `safe:location/safe:time/safe:stop` elements (and `safe:location/safe:path` element since the product can gather more than one Data Object).

```
<safe:corruptedElements>
   <safe:location>
      <safe:time>
         <safe:start>2004-12-16T23:55:53.000000Z</safe:start>
         <safe:stop>2004-12-16T23:55:57.55555Z</safe:stop>
      </safe:time>
      <safe:path>
         measurement/record
      </safe:path>
   </safe:location>
</safe:corruptedElements>
```

Example 3.28. Detected Corrupted Records (between 2004-12-16T23:55:53.000000Z and 2004-12-16T23:55:57.55555Z)

3.4.3. Use of Wild Cards

Some SAFE Types (`safe:acquisitionPeriodType`, `safe:instrumentType`...) allow, thanks to an `xs:any` Wild Card, elements which *are not* SAFE Types.

Any element “using” a Wild Card to be part of a SAFE Type:

- *shall* be qualified with a namespace different from `http://www.esa.int/safe/1.3`;
- *shall* have its type (a New Specific Type) fully defined by a SAFE Specialisation.

Wild Cards have been introduced in SAFE to prevent from using `xs:extension` mechanism. For the same goal, a SAFE Abstract Specialisation can define New Specific Types which allow Wild Cards (for further SAFE Specialisation(s)).

Wild Cards can be used according to the *critical* definition provided in Chapter 1.

Exemple of Use of the Platform Wilcard (ENVISAT ASAR APC Level 0 Specialisation) follows:

```
[...]
<metadataObject ID="platform"
                classification="DESCRIPTION" category="DMD">
   <metadataWrap textInfo="Platform Description"
                 vocabularyName="SAFE" mimeType="text/xml">
      <xmlData>
         <safe:platform>
            <safe:nssdcIdentifier>2002-009A</safe:nssdcIdentifier>
```

```
            <safe:familyName>ENVISAT</safe:familyName>
            <safe:number>1</safe:number>
            <safe:instrument>
               <safe:familyName abbreviation="ASAR"
               >Advanced Synthetic Aperture Radar</safe:familyName>
               <asar:txRxPolar>HV/HV</asar:txRxPolar>
               <asar:swath>IS3</asar:swath>
            </safe:instrument>
            <safe:timeReference>
               <safe:utc>2004-11-19T21:25:46.600940Z</safe:utc>
               <safe:clock>535089664</safe:clock>
               <safe:clockStep>3906249803</safe:clockStep>
            </safe:timeReference>
         </safe:platform>
      </xmlData>
   </metadataWrap>
</metadataObject>

[...]
```

Example 3.29. Manifest File for ENVISAT ASAR APC Level 0 Product

```
[...]

<xs:schema xmlns:xs  ="http://www.w3.org/2001/XMLSchema"
           xmlns:asar="http://www.esa.int/safe/1.3/envisat/asar"
           targetNamespace="http://www.esa.int/safe/1.3/envisat/asar"
           elementFormDefault="qualified">

   <xs:element name="txRxPolar" type="asar:txRxPolarType"/>
   <xs:element name="swath" type="asar:swathType"/>

   <xs:simpleType name="txRxPolarType">
      <xs:restriction base="xs:string"/>
   </xs:simpleType>

   <xs:simpleType name="swathType">
      <xs:restriction base="xs:string"/>
   </xs:simpleType>

</xs:schema>

[...]
```

Example 3.30. XML Schema Defining New Specific Types for ENVISAT ASAR Level 0 Products

```
[...]

<xs:complexType name="instrumentType">
   <xs:complexContent>
      <xs:restriction base="safe:instrumentType">
         <xs:sequence>
            <xs:element name="familyName"
                        type="safe:instrumentFamilyNameType"/>
            <xs:element name="number"
                        type="safe:instrumentNumberType"
                        minOccurs="0"/>
            <xs:element name="sideLookingAngle"
                        type="safe:sideLookingAngleType"
                        minOccurs="0"/>
            <xs:element name="fieldOfView"
                        type="safe:fieldOfViewType"
                        minOccurs="0"/>
            <xs:element ref="asar:txRxPolar"/>
            <xs:element ref="asar:swath"/>
         </xs:sequence>
      </xs:restriction>
   </xs:complexContent>
</xs:complexType>

[...]
```

Example 3.31. XML Schema Redefining instrumentType for ENVISAT ASAR Level 0 Products

3.5. Specialisation Types

SAFE provides SAFE Types for basic data of an EO product. For some products, some data cannot match the definition of any SAFE Type.

If needed, a SAFE Specialisation (possibly a SAFE Abstract Specialisation) shall provide New Specific Types. These types shall be fully defined by the SAFE Specialisation, qualified with a new namespace. Elements from New Specific Types shall become part of XFDU Types and/or SAFE Types.

If possible, value and occurrences shall be restricted.

Chapter 4. Representation Information of Data and Metadata Components

4.1. The Package and the XML Schema Components

SAFE *do not* impose any place within the SAFE Product for the XML Schema Components. These XML Schema Components can be placed in any subfolder within the SAFE Product.

Example of multi-folders within the SAFE Product:

```
My-SAFE-Product/manifest.safe
               /measurement-one.dat
               /measurement-two.dat
               /schemas/shared/types.xsd
               /schemas/measurement-one.xsd
               /schemas/measurement-two.xsd
```

Example 4.32. Example of Multi-Folders

Even if it is up to the creator of the specialisation to provide the location of the XML Schema Components, the following recommendation can be provided:

All XML Schema Components shall be put in a single folder named *rep-info*.

Example of single "rep-info" folder within the SAFE Product:

```
My-SAFE-Product/manifest.safe
               /measurement-one.dat
               /measurement-two.dat
               /rep-info/types.xsd
               /rep-info/measurement-one.xsd
               /rep-info/measurement-two.xsd
```

Example 4.33. Example of single "rep-info" folder

4.2. XML Schema Mechanisms

4.2.1. Use of xs:redefine

No recommendation is provided about use of `xs:redefine`.

4.2.2. Use of xs:import

No recommendation is provided about use of `xs:import`.

4.2.3. Use of xs:include

Use of `xs:include` for "calling" Additional abstract XML Schemas is recommended. This mechanism is simpler than other mechanisms provided by XML. All elements and types used by an XML Schema Object, if qualified, shall share the same namespace.

4.2.4. Global or Local Types?

As defined in the Section 3.2.4, XML Schema recommendations allow to declare types and elements. Declared types are global types, i.e. modifiables. Declared elements can be local typed, i.e. not modifiables.

For elements and types provided by an XML Schema Component, the following recommendations can be provided:

- if the type can be used inside another XML Schema Component, then it can be provided by an Additional abstract XML Schema as a global type;
- if the type is used only inside a single XML Schema Component, then it can be provided as a local type.

4.3. Accuracy Depth

SAFE is a *Standard Archive Format*. Long-term preservation of any data is pretty useless if after years noone knows what represent these data. So, the following recommendation can be provided:

Representation Information of any data hold by a SAFE Product shall be as accurate as possible.

These information can also be available in the SAFE Specialisation Control Book or in an existing specification referenced by the SAFE Specialisation Control Book.

Chapter 5. Product Identification

A SAFE Product is composed of multiple files (a SAFE Manifest and a collection of Components). As well the SAFE Manifest as the Components names shall not have a reference to the product.

For example a Manifest file shall almost always be named “manifest.safe” or “MANIFEST.SAFE”.

The name of the product shall be the name of the *directory* which gathers all the files of the SAFE Product (with the exception the SAFE Product is only composed of a SAFE Manifest).

Inside the SAFE Manifest of a product, there is a mandatory `version` attribute, which role is to identify the SAFE Product Specialisation or the SAFE Auxiliary Specialisation (and its major/minor version) defining content of the product.

Value of the `version` attribute shall *always* be defined by the SAFE Product Specialisation or the SAFE Auxiliary Specialisation.

The `version` attribute value *must* be a perfect identifier.

Examples of `version` value follow:

```
For an ENVISAT ASAR APC Level 0 Product:
   version="esa/safe/1.3/envisat/asar/apc/level-0"

For a Landsat MSS Level 0 Product:
   version="esa/safe/1.3/landsat/mss/level-0"

For a MODIS (TERRA or AQUA platform) Level 0 Product:
   version="esa/safe/1.3/modis/level-0"
```

Example 5.34. `version` Value

Chapter 6. Namespaces and Prefixes

6.1. Manifest File Representation Information

A SAFE Manifest contains elements and types qualified with several namespaces. Principal namespaces are:

- the SAFE namespace `http://www.esa.int/safe/1.3`;
- the SAFE - Active Sensor namespace `http://www.esa.int/safe/1.3/active-sensor`;
- the XFDU namespace `urn:ccsds:schema:xfdu:1`;
- the GML namespace `http://www.opengis.net/gml`;

If using a prefix inside the SAFE Manifest for anyone of these namespaces, it is recommended to create a prefix as explicit as possible:

- for `http://www.esa.int/safe/1.3`: “safe”;
- for `http://www.esa.int/safe/1.3/active-sensor`: “safe-as”;
- for `urn:ccsds:schema:xfdu:1`: “xfdu”;
- for `http://www.opengis.net/gml`: “gml”;

A New Specific Type, if provided by a SAFE Specialisation *must* be qualified with a new namespace. The following recommendation can be provided:

A namespace defined by a SAFE Specialisation *must* be a perfect identifier.

If using a prefix inside the SAFE Manifest for any new namespace, it is recommended to create a prefix as explicit as possible.

Examples of new namespace and associated prefix follow:

```
"http://www.esa.int/safe/1.3/envisat/gomos" (used prefix "gomos")
"http://www.esa.int/safe/1.3/seastar"       (used prefix "seastar")
"http://www.esa.int/safe/1.3/modis"         (used prefix "modis")
```

Example 6.35. Namespace and Prefix Defined for a New Specific Type

6.2. Component Representation Information

An XML Schema Component gathers elements and types qualified with several namespaces. Two namespaces are mandatory:

- the XML Schema namespace `http://www.w3.org/2001/XMLSchema` (current version);
- the SDF namespace `http://www.gael.fr/2005/04/drb/sdf` (current version).

If using a prefix inside the XML Schema Component for anyone of these namespaces, it is recommended to create a prefix as explicit as possible:

- for `http://www.w3.org/2001/XMLSchema`: “xs”;
- for `http://www.gael.fr/2005/04/drb/sdf`: “sdf”.

Elements and types defined by an XML Schema Component shall be qualified with a namespace. The following recommendations can be provided:

- a namespace defined for an element or a type *must* be a perfect identifier.
- regarding the XML Schemas tree (with included or imported XML Schemas), choice of a namespace shall be done *very carefully*.

If using a prefix inside an XML Schema Component for any new namespace, it is recommended to create a prefix as explicit as possible.

Examples of new namespace and associated prefix follow:

```
"http://www.esa.int/safe/1.3/envisat" (used prefix "envisat")
"http://www.esa.int/safe/1.3/seastar"  (used prefix "seastar")
"http://www.esa.int/safe/1.3/modis"    (used prefix "modis")
```

Example 6.36. Namespace and Prefix Defined for Elements and Types

Chapter 7. Naming Recommendations

7.1. File Naming Recommendations

7.1.1. SAFE Package

According to the [SAFE-CORE], SAFE Products are usually composed of a SAFE Manifest and one or more Component(s). The SAFE Product shall be packaged either in a directory or in another type of packaging (zip, tar, etc.). For the name of the directory or package file, the following recommendation can be provided:

The directory or package file shall be named:

“MMNN_PPPPPPPPPP_ttttttttttttttt_TTTTTTTTTTTTTTT_FFF_OOOOOO_XXXX.EEEE”

where:

- “MM”: platform name: 2 uppercase letters;
- “NN”: mission number: 2 digits;
- “PPPPPPPPPP”: product type: 10 uppercase letters, digits or underscores “_”;
- “ttttttttttttttt”: product start time, where:
  - “ttttttttttttttt” = “YYYY” (year: 4 digits) + “MM” (month: 2 digits) + “DD” (day: 2 digits) + “T” (constant) + “HH” (hours: 2 digits) + “MM” (minutes: 2 digits) + “SS” (seconds: 2 digits).
- “TTTTTTTTTTTTTTT”: product stop time, where:
  - “TTTTTTTTTTTTTTT” = “YYYY” (year: 4 digits) + “MM” (month: 2 digits) + “DD” (day: 2 digits) + “T” (constant) + “HH” (hours: 2 digits) + “MM” (minutes: 2 digits) + “SS” (seconds: 2 digits).
- “FFF”: originating facility which has generated the product: 3 characters;
- “OOOOOO”: platform absolute orbit: up to 6 characters (no leading zeroes);
- “XXXX”: CRC-16 computed on the SAFE Manifest: 4 hexadecimal characters;
- “.EEEE”: the SAFE extension: “SAFE”.

Examples of SAFE Product composed only of a SAFE Manifest follow:

```
"EN01_ASA_WS__0P_20050130T080000_20050130T081000_ESR_15263_1FA5.SAFE"
"ER01_AT1_ATS_0P_19920531T234020_19920601T012340_GAT_4584_1FA5.SAFE"
```

Example 7.37. SAFE Manifest Names (products composed only of a SAFE Manifest)

7.1.2. Manifest File

According to the [SAFE-CORE], for SAFE Products which are composed of a SAFE Manifest and one or more Component(s), the SAFE Manifest File *must* be named “manifest.safe” or “MANIFEST.SAFE”.

For SAFE Products which are *only* composed of a SAFE Manifest, the following recommendation can be provided:

The SAFE Manifest shall be named:

“MMNN_PPPPPPPPPP_tttttttttttttt_TTTTTTTTTTTTTTT_FFF_OOOOOO_XXXX.EEEE”

where:

- “MM”: platform name: 2 uppercase letters;
- “NN”: mission number: 2 digits;
- “PPPPPPPPPP”: product type: 10 uppercase letters, digits or underscores “_”;
- “tttttttttttttt”: product start time, where:
  - “tttttttttttttt” = “YYYY” (year: 4 digits) + “MM” (month: 2 digits) + “DD” (day: 2 digits) + “T” (constant) + “HH” (hours: 2 digits) + “MM” (minutes: 2 digits) + “SS” (seconds: 2 digits).
- “TTTTTTTTTTTTTTT”: product stop time, where:
  - “TTTTTTTTTTTTTTT” = “YYYY” (year: 4 digits) + “MM” (month: 2 digits) + “DD” (day: 2 digits) + “T” (constant) + “HH” (hours: 2 digits) + “MM” (minutes: 2 digits) + “SS” (seconds: 2 digits).
- “FFF”: originating facility which has generated the product: 3 characters;
- “OOOOOO”: platform absolute orbit: up to 6 characters (no leading zeroes);
- “XXXX”: CRC-16 computed on the SAFE Manifest: 4 hexadecimal characters;
- “.EEEE”: the SAFE extension: either “safe” or “SAFE”.

Examples of SAFE Product composed only of a SAFE Manifest follow:

```
"EN01_ASA_WS__0P_20050130T080000_20050130T081000_ESR_15263_1FA5.safe"
"ER01_AT1_ATS_0P_19920531T234020_19920601T012340_GAT_4584_1FA5.safe"
```

Example 7.38. SAFE Manifest Names (products composed only of a SAFE Manifest)

7.1.3. Data Components

The [SAFE-CORE] defines that:

Binary Data Components shall always be named:

```
filename = “[a-z,0-9,-].dat”
```

ASCII Data Components shall always be named:

```
filename = “[a-z,0-9,-].txt”
```

XML Data Components shall always be named:

```
filename = “[a-z,0-9,-].xml”
```

Even if the part of the name *before* the extension “.dat”, “.txt” or “.xml” is free, the following recommendation can be provided.

The name of a Data Component shall be as explicit as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).

```
A Data Component holding measurement data could be named:
      “measurement.dat”
```

Example 7.39. Name of a Data Component

```
A Data Component holding the ocean wave spectrum of a small, high-resolution,
complex image could be named:
      “ocean-wave-spectra.dat”
```

Example 7.40. Name of a Data Component

```
A Data Component holding information describing the pressure, temperature
and height correction profiles could be named:

      “pressure-temperature-height.dat”
```

Example 7.41. Name of a Data Component

7.1.4. Metadata Components

The [SAFE-CORE] defines that:

Binary Metadata Components shall always be named:

```
filename = “[a-z,0-9,-].dat”
```

ASCII Metadata Components shall always be named:

```
filename = “[a-z,0-9,-].txt”
```

XML Metadata Components shall always be named:

```
filename = “[a-z,0-9,-].xml”
```

Even if the part of the name *before* the extension “.dat”, “.txt” or “.xml” is free, the following recommendation can be provided.

The name of a Metadata Component shall be as explicit as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).

```
A Metadata Component holding an index for a measurement Data Component could
be named:

      “measurement-index.dat”
```

Example 7.42. Name of a Metadata Component

```
A Metadata Component holding the reference star spectrum measurement
corresponding to a measurement Data Component could be named:

      “reference-star-spectrum.dat”
```

Example 7.43. Name of a Metadata Component

```
A Metadata Component holding solar angles (nadir view) could be named:

      “nadir-view-solar-angles.dat”
```

Example 7.44. Name of a Metadata Component

7.1.5. XML Schema Components

7.1.5.1. Manifest File Representation Information

7.1.5.1.1. XFDU Types and SAFE Types

A SAFE Manifest gathers elements and types defined by:

- the xfdu.xsd XML Schema provided in the Appendix A of the [SAFE-CORE] (for XFDU Types).
- the safe.xsd XML Schema provided in the Appendix B of the [SAFE-CORE] (for SAFE Types).

As defined in the Chapter 3, there can be one or more SAFE Abstract Specialisation(s) involved in a SAFE Product Specialisation or in a SAFE Auxiliary Specialisation (i. e. a SAFE Specialisation provides one or more redefinition(s) of XFDU Types and one or more redefinition(s) of SAFE Types).

The following recommendations can be provided.

- Every XML Schema which redefines XFDU Types shall be named “xfdu.xsd”;
- Every XML Schema which redefines SAFE Types shall be named “safe.xsd”.

Multiple xfdu.xsd and multiple safe.xsd shall be distinct because placed in appropriate directories.

If a single directory gathers more than one xfdu.xsd XML Schema or more than one safe.xsd XML Schema, the following recommendations can be provided.

- The XML Schema which redefines XFDU Types shall be named “xfdu-[a-z,-].xsd”;
- The XML Schema which redefines SAFE Types shall be named “safe-[a-z,-].xsd”.

The name of the XML Schema shall be defined freely but as explicitly as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).

7.1.5.1.2. New Specific Types

Several SAFE Types allow Wild Cards, by use of `xs:any`. For some SAFE Specialisations, new New Specific Types defined by the SAFE Specialisation shall be required inside SAFE Types.

SAFE allows also New Specific Types as Wrapped Metadata Objects. The following example can be part of a SAFE Manifest of a product defined by a SAFE Specialisation.

```
[...]

   <metadataObject ID="occultationInformation"
                   classification="DESCRIPTION" category="DMD">
      <metadataWrap textInfo="Occultation Information"
                    vocabularyName="SAFE" mimeType="text/xml">
         <xmlData>
            <gomos:occultation>
               <gomos:duration>7050</gomos:duration>
               <gomos:number>14</gomos:number>
               <gomos:samplingDuration>500</gomos:samplingDuration>
               <gomos:measurementNumber>141</gomos:measurementNumber>
            </gomos:occultation>
         </xmlData>
      </metadataWrap>
   </metadataObject>

[...]
```

Example 7.45. New Specific Type as a Wrapped Medatada Object

These new types shall be defined in one or more XML Schema(s).

The following recommendations can be provided.

The XML Schema shall be named “[a-z,-]-types.xsd”.

The name of the XML Schema shall be defined freely but as explicitly as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).

7.1.5.2. Component Representation Information

The [SAFE-CORE] defines that each Component *must* have a Representation Information.

Each Component (Data Component or Metadata Component; ".dat", ".txt" or ".xml") shall be accompanied with an XML Schema Component describing its content.

The [SAFE-CORE] defines that:

- XML schemas which describe binary, XML or ASCII Data Components or Metadata Components shall always be named:

```
filename = "[a-z,0-9,-].xsd"
```

- Additional abstract XML Schemas shall always be named:

```
filename = "[a-z,-]-object-types.xsd"
```

Even if the part of the name *before* the extension ".xsd" (for Additional abstract XML Schemas, *before* "-object-types.xsd") is free, the following recommendations can be provided.

- The name of an XML Schema Component, unique or part of the Representation Information of a Data Component or a Metadata Component shall be as explicit as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).
- The name of an XML Schema Component which describes a Data Component or a Metadata Component shall match the name of the Data Component or the Metadata Component (except extensions).

```
An XML Schema Component which describes a Metadata Component named:
      "nadir-view-solar-angles.dat"
shall be named:
      "nadir-view-solar-angles.xsd"
```

Example 7.46. Name of an XML Schema Component

- The name of an XML Schema Component which is an Additional abstract XML Schema shall be defined freely but as explicitly as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).

```
An Additional Abstract XML Schema referenced by all Level 0 Products of a
platform named:

      “ETERNITY”

could be named:

      “eternity-level-0-object-types.xsd”
```

Example 7.47. Name of an Additional Abstract XML Schema

7.2. Entities Naming

A SAFE Specialisation shall use types (SAFE Types and XFDU Types) provided by SAFE.

But a SAFE Specialisation defines new types, specific to the Specialisation. It *can* define New Specific Types for the SAFE Manifest, and *shall* define types and elements for the Representation Information of Components.

Whether it is for the SAFE Manifest or for the Representation Information of Components, types and elements defined shall be named following these recommendations:

Entities name shall use only [a-z,A-Z,0-9,-] characters.

Entities name shall be lowerCamelCase (mixed case with a lowercase first letter, internal words start with capital letters).

Name of an entity shall be as explicit as possible (avoid acronyms and abbreviations, unless the abbreviation is much more widely used than the long form, such as URL or XML).

Examples of recommended element and type name follow:

```
measurementRecord

instrumentStatus

instrumentStatusType

packetFieldHeaderType
```

Example 7.48. Recommended Entities Field Naming

Examples of not recommended element and type name follow:

```
measurement-record
measurement_record
measRec
MEAS-REC
```

Example 7.49. Not Recommended Entities Field Naming

7.3. Manifest File Field Values

Many XFDU Types hold one or more attributes. These attributes can be classified in two categories:

- attributes holding an information (`mimeType`, `unitType`...);
- attributes permitting the internal linking (`ID`, `repID`...).

For attributes of first category, either the [SAFE-CORE] provides enumerations or allow any value regarding the XML Type (`xs:string`, `xs:double`...).

For attributes of second category, either the [SAFE-CORE] provides mandatory patterns (for `metadataObject/@ID` only), or allow any value regarding the XML Type (`xs:NCName`).

The [SAFE-CORE] defines relationships between Content Units, dataObjects and metadataObjects. If these relationships are defined for basic cases (an index, a Data Component...), value of attributes permitting the internal linking is not mentioned.

7.3.1. Root Content Unit

While recommended at Section 3.3.2, the `ID` attribute of the root `contentUnit` should be used.

Value of this `ID` attribute is free but the following recommendation can be provided.

Value of the `ID` attribute of the root `contentUnit` shall always be “packageUnit”.

```
<informationPackageMap>
   <xfdu:contentUnit ID="packageUnit"
                     textInfo="ERS AMI SAR Level 0"
                     pdiID="processing"
                     dmdID="acquisitionPeriod platform">
      [...]
   </xfdu:contentUnit>
```

```
    </informationPackageMap>
```

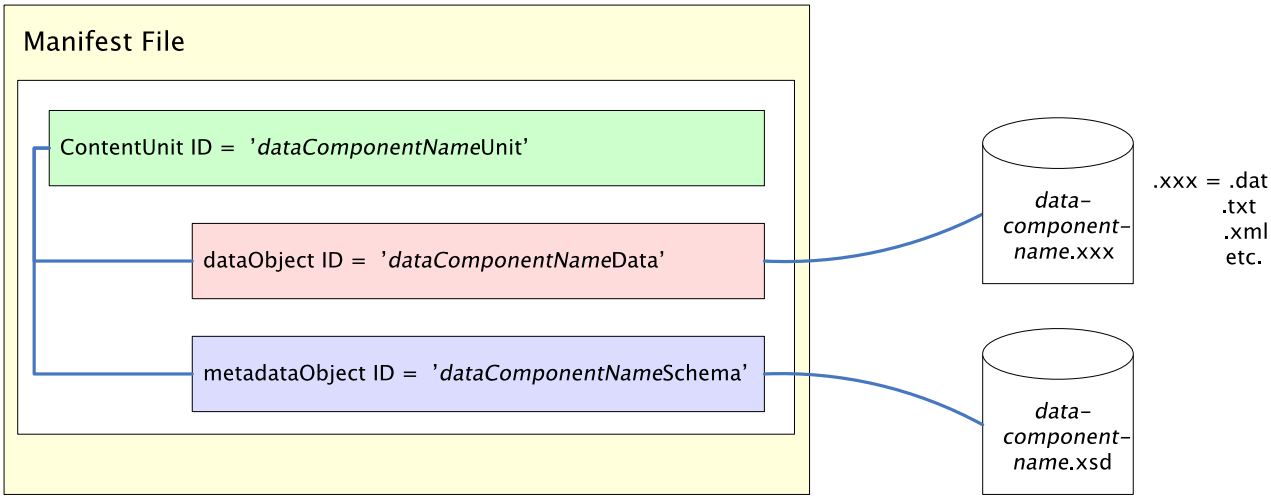
Example 7.50. Manifest File Field Values: The ID attribute of the Root Content Unit

7.3.2. Case of Data Object and its Representation Information (and their common Content Unit)

The [SAFE-CORE] defines:

- a Data Object is constituted of a `dataObject` element and a Data Component;
- an XML Schema Object is constituted of a `metadataObject` element and an XML Schema Component.

ID value of the `dataObject` element, ID value of the `metadataObject` element, ID value of the `contentUnit` pointing to these objects, filename of the Data Component and filename of the XML Schema Component shall be as close as possible (except for suffixes “Data”, “Schema” or “Unit”, extensions “.dat” / “.xml” / “.txt” or “.xsd”, and regarding the recommendations provided in Section 7.1.3, Section 7.1.5.2 and Section 7.2). The following figure and example illustrate this recommendation:



Manifest File Field Values: Data Object and its Representation Information

```
[...]
```

```
   <xfdu:contentUnit ID="aerosolsUnit"
                     repID="aerosolsSchema
                            envisatSchema">
      <dataObjectPointer dataObjectID="aerosolsData"/>
   </xfdu:contentUnit>

[...]

   <metadataObject ID="aerosolsSchema"
                   classification="SYNTAX" category="REP">
      <metadataReference locatorType="URL" href="aerosols.xsd"
                         vocabularyName="SDF" mimeType="text/xml"/>
   </metadataObject>

[...]

   <dataObject ID="aerosolsData"
               repID="aerosolsSchema
                      envisatSchema">
      <byteStream mimeType="application/octet-stream">
         <fileLocation locatorType="URL" href="aerosols.dat"/>
         <checksum checksumName="MD5">e94f8303ad92d056c3b2d0d5d250111b</checksum>
      </byteStream>
   </dataObject>

[...]
```

Example 7.51. Manifest File Field Values: Data Object and its Representation Information, and their common Content Unit

7.3.3. Case of Referenced Metadata Object

7.3.3.1. Case of XML Schema Object

7.3.3.1.1. Case of XML Schema Describing a Component

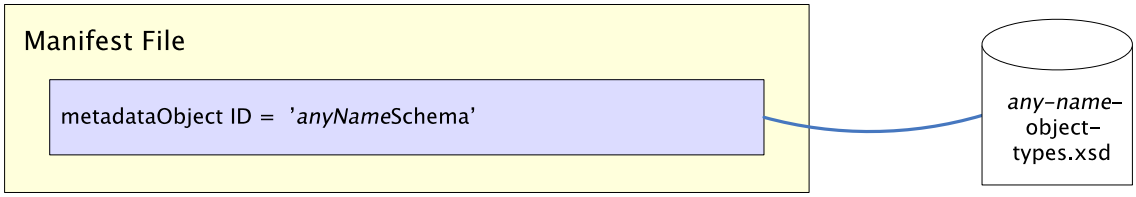
This case is treated in Section 7.3.2.

7.3.3.1.2. Case of Additional Abstract XML Schema

The [SAFE-CORE] defines:

- an Additional abstract XML Schema (a particular case of XML Schema Object) is constituted of a `metadataObject` element and a XML Schema Component.

ID value of the `metadataObject` element and filename of the XML Schema Component shall be free, but as close as possible (regarding the recommendations provided in Section 7.1.5.2 and Section 7.2). The following figure and example illustrate this recommendation:



Manifest File Field Values: Case of Additional Abstract XML Schema

```
[...]
   <metadataObject ID="envisatLevel0MeasurementSchema"
                   classification="SYNTAX" category="REP">
      <metadataReference locatorType="URL"
                         href="envisat-level-0-object-types.xsd"
                         vocabularyName="SDF" mimeType="text/xml"/>
   </metadataObject>
[...]
```

Example 7.52. Manifest File Field Values: Case of Additional Abstract XML Schema

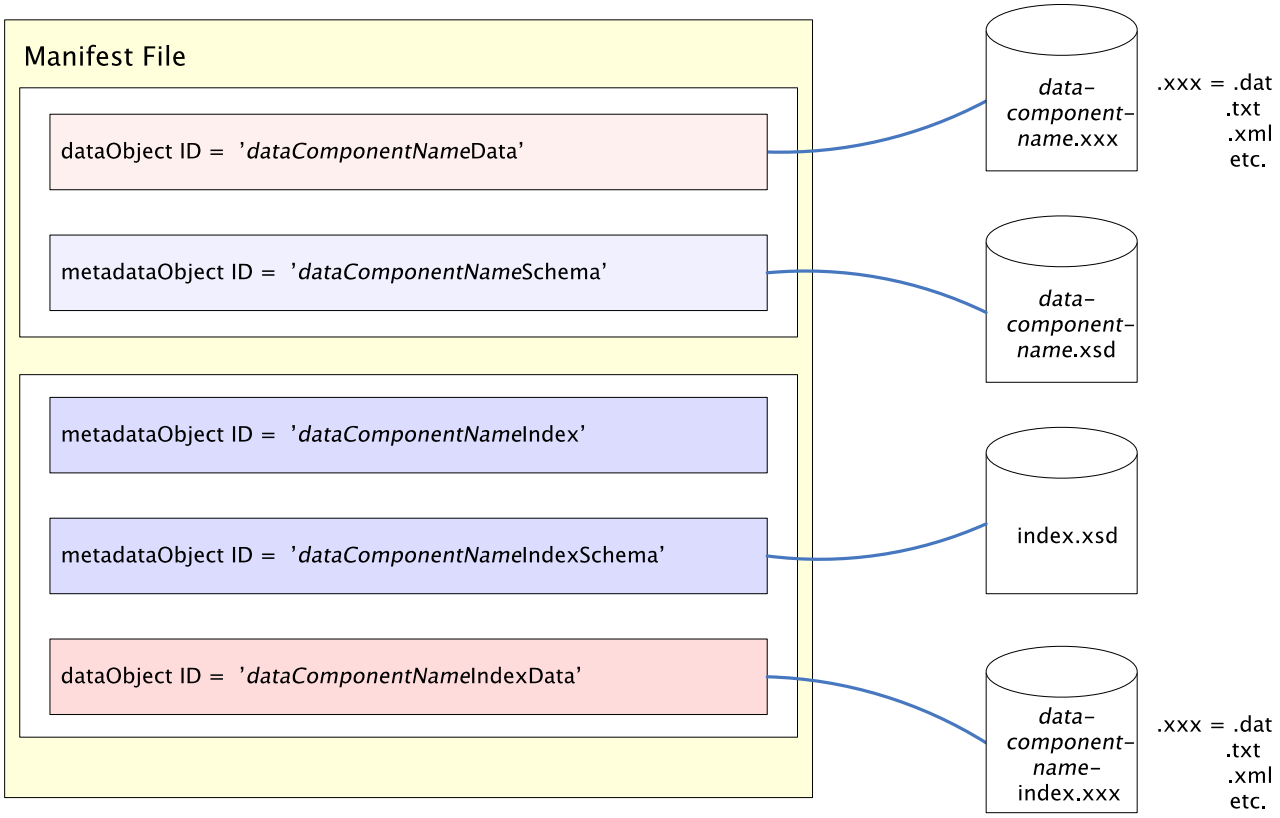
7.3.3.2. Case of Metadata Object

7.3.3.2.1. Case of Data Index Object and its Representation Information

The [SAFE-CORE] defines:

- a Metadata Object holding an index is constituted of a `metadataObject` element, a `dataObject` element and a Metadata Component;
- an XML Schema Object is constituted of a `metadataObject` element and a XML Schema Component;
- for a Data Object or a Metadata Object, there shall be only one index file; for an index file, there shall be only one Data Object or Metadata Object.

ID value of the two `dataObject` elements (both for data file and its index), ID value of the three `metadataObject` elements (both for data file and its index), filename of the Data Component and filename of the Metadata Components shall be as close as possible (except for suffixes “Data”, “Schema”, “Index”, “IndexData” or “IndexSchema”, extensions “.dat” / “.xml” / “.txt” or “.xsd”, and regarding the recommendations provided in Section 7.1.3, Section 7.1.4, Section 7.1.5.2 and Section 7.2). Note that for any index, the Representation Information is the index.xsd XML Schema provided in the Appendix E of the [SAFE-CORE] . The following figure and example illustrate this recommendation:



Manifest File Field Values: Case of Data Index Object and its Representation Information

```
[...]

   <dataObject ID="measurementData"
               repID="measurementSchema">
      <byteStream mimeType="application/octet-stream">
         <fileLocation locatorType="URL" href="measurement.dat"
                  textInfo="Measurement Data Set File"/>
            <checksum checksumName="MD5">
               e94f8303ad92d056c3b2d0d5d250111b</checksum>
      </byteStream>
   </dataObject>

[...]

   <metadataObject ID="measurementSchema"
                   classification="SYNTAX" category="REP">
      <metadataReference locatorType="URL" href="measurement.xsd"
                         vocabularyName="SDF" mimeType="text/xml"/>
   </metadataObject>

[...]

   <metadataObject ID="measurementIndex"
                   classification="DESCRIPTION" category="DMD">
```

```
      <dataObjectPointer
                    dataObjectID="measurementIndexData"/>
   </metadataObject>

[...]

   <metadataObject ID="measurementIndexSchema"
                    classification="SYNTAX" category="REP">
      <metadataReference locatorType="URL"
                           href="index.xsd"
                           vocabularyName="SDF" mimeType="text/xml"/>
   </metadataObject>

[...]

   <dataObject ID="measurementIndexData"
               repID="measurementIndexSchema">
      <byteStream mimeType="application/octet-stream">
         <fileLocation locatorType="URL" href="measurement-index.dat"
                 textInfo="Measurement Data Set Index File"/>
            <checksum checksumName="MD5">
               e94f8303ad92d056c3b2d0d5d250111b</checksum>
      </byteStream>
   </dataObject>

[...]
```

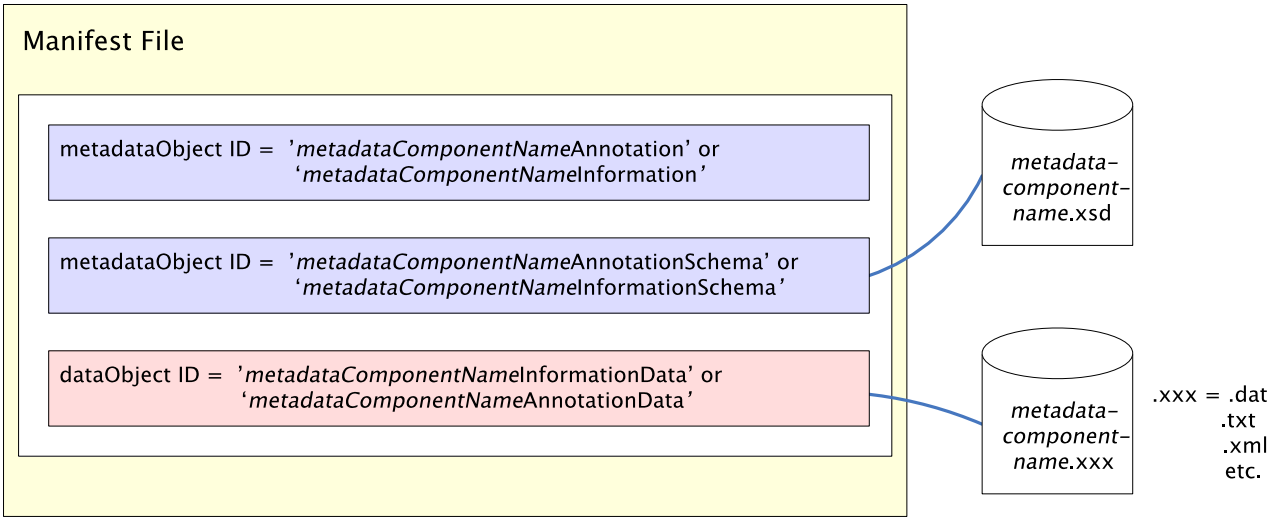
Example 7.53. Manifest File Field Values: Case of Data Index Object and its Representation Information

7.3.3.2.2. Case of Metadata Object (not an index)

The [SAFE-CORE] defines:

- a Metadata Object is constituted of a `metadataObject` element, a `dataObject` element and a Metadata Component;
- an XML Schema Object is constituted of a `metadataObject` element and a XML Schema Component.

ID value of the `dataObject` element, ID value of the two `metadataObject` elements, filename of the Metadata Component and filename of the XML Schema Component shall be as close as possible (except for suffixes “Annotation” or “Information”, “AnnotationData” or “InformationData”, “AnnotationSchema” or “InformationSchema”, extensions “.dat” / “.xml” / “.txt” or “.xsd”, and regarding the recommendations provided in Section 7.1.4, Section 7.1.5.2 and Section 7.2). The following figure and example illustrate this recommendation:



Manifest File Field Values: Case of Metadata Object (not an index)

```
[...]

   <metadataObject ID="landSeaMaskInformation"
                   classification="DESCRIPTION" category="DMD">
      <dataObjectPointer
                   dataObjectID="landSeaMaskInformationData"/>
   </metadataObject>

[...]

   <metadataObject ID="landSeaMaskInformationSchema"
                   classification="SYNTAX" category="REP">
      <metadataReference locatorType="URL"
                         href="land-sea-mask.xsd"
                         vocabularyName="SDF" mimeType="text/xml"/>
   </metadataObject>

[...]

   <dataObject ID="landSeaMaskInformationData"
               repID="landSeaMaskInformationSchema">
      <byteStream mimeType="application/octet-stream">
         <fileLocation locatorType="URL" href="land-sea-mask.dat"
                 textInfo="Measurement Data Set Index File"/>
            <checksum checksumName="MD5">
               e94f8303ad92d056c3b2d0d5d250111b</checksum>
      </byteStream>
   </dataObject>

[...]
```

Example 7.54. Manifest File Field Values: Case of Metadata Object (not an index)

Appendix A. ERS AMI SAR Level 0 Specialisation: Example of SAFE Manifest document (informative)

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
   SAFE - Standard Archive Format for Europe
   Copyright (C) 2004,2005,2006,2007,2008,2009,2010 European Space Agency (ESA)
   Copyright (C) 2004,2005,2006,2007,2008,2009,2010 GAEL Consultant
   GNU Lesser General Public License (LGPL)

   This file is part of SAFE

   SAFE is free software: you can redistribute it and/or modify
   it under the terms of the GNU Lesser General Public License as published by
   the Free Software Foundation, either version 3 of the License, or
   (at your option) any later version.

   SAFE is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public License
   along with this program.  If not, see <http://www.gnu.org/licenses/>.
-->


<!-- ERS AMI SAR Level 0 archive.
     This document is a sample manifest file of a SAFE archive. -->

<xfdu:XFDU xmlns:xfdu="urn:ccsds:schema:xfdu:1"
           xmlns:safe="http://www.esa.int/safe/1.3"
           xmlns:gml ="http://www.opengis.net/gml"
           version   ="esa/safe/1.3/ers/ami/sar/level-0">

   <!-- ====================================================================
        INFORMATION PACKAGE MAP SECTION
        ==================================================================== -->
   <informationPackageMap>

      <xfdu:contentUnit ID="packageUnit"
                        textInfo="ERS AMI SAR Level 0"
                        pdiID="processing"
                        dmdID="acquisitionPeriod platform">

         <!-- The single ERS AMI SAR Level 0 measurement data set -->
         <xfdu:contentUnit ID="measurementUnit"
                           textInfo="Measurement Data Unit"
                           dmdID="measurementQualityInformation
                                  measurementOrbitReference
                                  measurementFrameSet
                                  measurementIndex"
                           repID="measurementSchema">

            <dataObjectPointer dataObjectID="measurementData"/>

         </xfdu:contentUnit>

         <!-- An index to the measurement data set for access efficiency -->
         <xfdu:contentUnit ID="measurementIndexUnit"
```

```
                        textInfo="Measurement Data Index"
                        repID="measurementIndexSchema">

            <dataObjectPointer dataObjectID="measurementIndexData"/>

         </xfdu:contentUnit>

      </xfdu:contentUnit>

   </informationPackageMap>

   <!-- Metadata Section -->
   <metadataSection>

      <!-- Processing -->
      <metadataObject ID="processing"
                      classification="PROVENANCE" category="PDI">
         <metadataWrap mimeType="text/xml" vocabularyName="SAFE"
                       textInfo="Processing">
            <xmlData>
               <safe:processing name="Conversion from MDPS to SAFE format"
                                start="2006-01-12T15:27:50.975000Z">
                  <safe:facility country="France" name="ACS"
                                 organisation="ACS"
                                 site="Roma">
                     <safe:software name="ACS ERS SAR Transformer"
                                    version="1.1"/>
                  </safe:facility>
                  <safe:resource name=
                  "AMI_SAR__0P_20021231T042805_20021231T061130_PS_40239.E2"
                                 role="Input data">
                     <safe:processing name="Downlink">
                        <safe:facility country="Sweden" name="Kiruna"
                                       organisation="ESA" site="Kiruna"/>
                     </safe:processing>
                  </safe:resource>
               </safe:processing>
            </xmlData>
         </metadataWrap>
      </metadataObject>

      <!-- Acquisition Period -->
      <metadataObject ID="acquisitionPeriod"
                      classification="DESCRIPTION" category="DMD">
         <metadataWrap textInfo="Acquisition Period"
                       vocabularyName="SAFE" mimeType="text/xml">
            <xmlData>
               <safe:acquisitionPeriod>
                  <safe:startTime>2004-05-18T00:30:55.134207Z</safe:startTime>
                  <safe:stopTime>2004-05-18T00:30:57.134207Z</safe:stopTime>
               </safe:acquisitionPeriod>
            </xmlData>
         </metadataWrap>
      </metadataObject>

      <!-- Platform description -->
      <metadataObject ID="platform"
                      classification="DESCRIPTION" category="DMD">
         <metadataWrap textInfo="Platform Description"
                       vocabularyName="SAFE" mimeType="text/xml">
            <xmlData>
               <safe:platform>
                  <!-- Platform identification -->
```

```
                <safe:nssdcIdentifier>1995-021A</safe:nssdcIdentifier>
                <safe:familyName>ERS</safe:familyName>
                <safe:number>2</safe:number>
                <!-- Instrument identification -->
                <safe:instrument>
                   <safe:familyName abbreviation="AMI"
                   >Active Microwave Instrument</safe:familyName>
                   <safe:mode identifier="IM">Image Mode</safe:mode>
                </safe:instrument>
                <safe:timeReference>
                   <safe:utc>2004-03-12T09:38:12.354666</safe:utc>
                   <safe:clock>2521531000</safe:clock>
                   <safe:clockStep>23524000</safe:clockStep>
                </safe:timeReference>
                <safe:timeReference>
                   <safe:utc>2005-03-12T09:38:12.354666</safe:utc>
                   <safe:clock>2521531000</safe:clock>
                   <safe:clockStep>23524000</safe:clockStep>
                </safe:timeReference>
             </safe:platform>
          </xmlData>
       </metadataWrap>
    </metadataObject>

    <!-- Quality information of the measurements -->
    <metadataObject ID="measurementQualityInformation"
                    classification="DESCRIPTION" category="DMD">
       <metadataWrap textInfo="Quality Information"
                     vocabularyName="SAFE" mimeType="text/xml">
          <xmlData>
             <safe:qualityInformation>
                <!-- RangeLine 23 is missing -->
                <safe:missingElements>
                   <safe:location>
                      <safe:path following="22">
                         measurements/rangeLine
                      </safe:path>
                   </safe:location>
                   <safe:count value="1"/>
                </safe:missingElements>
                <!-- 5 RangeLines are missing
                     We only know that there are before the 54th record -->
                <safe:missingElements>
                   <safe:location>
                      <safe:path before="54">
                         measurements/rangeLine
                      </safe:path>
                   </safe:location>
                   <safe:count value="5"/>
                </safe:missingElements>
                <!-- 4 CORRUPTED (UNKNOWNCAUSE) RangeLines -
                position #7, #8, #9, #10 -->
                <safe:corruptedElements>
                   <safe:location>
                      <safe:path>
                         measurements/rangeLine[
                            position() > 6 and position() &lt; 11]
                      </safe:path>
                   </safe:location>
                </safe:corruptedElements>
             </safe:qualityInformation>
          </xmlData>
       </metadataWrap>
```

```
      </metadataObject>

      <!-- Orbital reference of the measurements -->
      <metadataObject ID="measurementOrbitReference"
                      classification="DESCRIPTION" category="DMD">
         <metadataWrap textInfo="Orbit Reference"
                       vocabularyName="SAFE" mimeType="text/xml">
            <xmlData>
               <safe:orbitReference>
                  <safe:orbitNumber type="start"
                                    groundTrackDirection="ascending"
                                    >79</safe:orbitNumber>
                  <safe:relativeOrbitNumber
                     type="start">7</safe:relativeOrbitNumber>
                  <safe:cycleNumber>27</safe:cycleNumber>
                  <safe:phaseIdentifier>2</safe:phaseIdentifier>
               </safe:orbitReference>
            </xmlData>
         </metadataWrap>
      </metadataObject>

      <!-- Frame set of the measurements -->
      <metadataObject ID="measurementFrameSet"
                      classification="DESCRIPTION" category="DMD">
         <metadataWrap textInfo="Frame Set" vocabularyName="SAFE"
                       mimeType="text/xml">
            <xmlData>
               <safe:frameSet>
                  <safe:footPrint
                     srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
                     <gml:coordinates>-44.286612,-14.937040 -75.312935,-11.387472
                     -75.312935,-11.387472</gml:coordinates>
                  </safe:footPrint>
               </safe:frameSet>
            </xmlData>
         </metadataWrap>
      </metadataObject>

      <!-- Measurement Data Set Index Representation Information -->
      <metadataObject ID="measurementIndex"
                      classification="DESCRIPTION" category="DMD">
         <dataObjectPointer dataObjectID="measurementIndexData"/>
      </metadataObject>

      <!-- Measurement Schema -->
      <metadataObject ID="measurementSchema"
                      classification="SYNTAX" category="REP">
         <metadataReference locatorType="URL" href="rep-info/measurement.xsd"
                            vocabularyName="SDF" mimeType="text/xml"/>
      </metadataObject>

      <!-- Measurement Data Set Index Representation Information -->
      <metadataObject ID="measurementIndexSchema"
                      classification="SYNTAX" category="REP">
         <metadataReference locatorType="URL" href="rep-info/index.xsd"
                            vocabularyName="SDF" mimeType="text/xml"/>
      </metadataObject>

   </metadataSection>

   <!-- ====================================================================
        DATA OBJECT SECTION
        ==================================================================== -->
```

```
   <dataObjectSection>

      <!-- Measurement -->
      <dataObject ID="measurementData"
                  repID="measurementSchema">
         <byteStream mimeType="application/octet-stream">
            <fileLocation locatorType="URL" href="measurement.dat"
                          textInfo="Measurement Data Set File"/>
            <checksum
               checksumName="MD5">3bf095eedf4b8c9c87252dda8aeec1c0</checksum>
         </byteStream>
      </dataObject>

      <!-- Index -->
      <dataObject ID="measurementIndexData"
                  repID="measurementIndexSchema">
         <byteStream mimeType="application/octet-stream">
            <fileLocation locatorType="URL" href="measurement-index.dat"
                          textInfo="Measurement Data Set Index File"/>
            <checksum
               checksumName="MD5">d5fg4d4g3fds45s3s3d4fs36d3f45</checksum>
         </byteStream>
      </dataObject>

   </dataObjectSection>

</xfdu:XFDU>
```

Appendix B. GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of

[1] http://www.fsf.org/

Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies.

If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties — for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their

names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document

under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See Copyleft[2].

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

[2] http://www.gnu.org/copyleft/

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright © YEAR YOUR NAME

Permission is granted to copy, distribute and/or modify this document under the
terms of the GNU Free Documentation License, Version 1.3 or any later version
published by the Free Software Foundation; with no Invariant Sections, no
Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in
the section entitled “GNU Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with… Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts
being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Appendix C. SAFE Document Improvement Proposal

Document Improvement Proposal		
Document I.D. PGSI-GSEG-EOPG-FS-05-0002	**Document Title.** Standard Archive Format for Europe Control Book - Volume 2 - Recommendation for specialisations	**Document Status.** Issue 1 Revision 11 - 2010-06-22
Recommended Improvement. (identify clauses, subclauses and include modified text and/or graphic, attach pages as necessary)		
Reason for Recommendation.		
Originator of Recommendation.		
Name:	Organization:	
Addres:	Phone: Fax: E-Mail:	**Date of Submission.**
Send to ESRIN Secretariat .		
Name: Dario Romano	Address: ESRIN , Via Galileo Galilei - 00044 - Casella Postale 64 Frascati - Italy	Phone: (39) 06 941801 Fax: (39) 06 94180 280

Index