The Consultative Committee for Space Data Systems

Report Concerning Space Data System Standards

PRODUCER-ARCHIVE INTERFACE SPECIFICATION (PAIS)

A TUTORIAL

DRAFT INFORMATIONAL REPORT

CCSDS 651.2-G-0

DRAFT GREEN BOOK

April 2014

(2014-04-16)

AUTHORITY

Issue:	Draft Green Book, Issue 0
Date:	April 2014
Location:	Not Applicable

(WHEN THIS INFORMATIONAL REPORT IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and reflects the consensus of technical experts from CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-3).

This document is published and maintained by:

CCSDS Secretariat
Space Communications and Navigation Office, 7L70
Space Operations Mission Directorate
NASA Headquarters
Washington, DC 20546-0001, USA

FOREWORD

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This document is therefore subject to CCSDS document management and change control procedures which are defined in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-3). Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be addressed to the CCSDS Secretariat at the address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- China National Space Administration (CNSA)/People's Republic of China.
- Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.
- Federal Space Agency (FSA)/Russian Federation.
- UK Space Agency/United Kingdom.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Belgian Federal Science Policy Office (BFSPO)/Belgium.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
- Chinese Academy of Sciences (CAS)/China.
- Chinese Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- Danish National Space Center (DNSC)/Denmark.
- Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- National Institute of Information and Communications Technology (NICT)/Japan.
- National Oceanic and Atmospheric Administration (NOAA)/USA.
- National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
- National Space Organization (NSPO)/Chinese Taipei.
- Naval Center for Space Technology (NCST)/USA.
- Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
- South African National Space Agency (SANSA)/Republic of South Africa.
- Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
- Swedish Space Corporation (SSC)/Sweden.
- United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	**Title and Issue**	**Date**	**Status**
CCSDS 651.2-G-0	Producer-Archive Interface Specification (PAIS) – A Tutorial, Draft Informational Report, Issue 0	November 2013	Current draft
CCSDS 651.2-G-0	*Producer-Archive Interface Specification (PAIS) – A Tutorial, Draft Informational Report, Issue 0*	*November 26, 2013*	*Improved Section 2 fixing in particular the SIP Constraints that was wrongly considered as part of the MOT, and introducing a tentative of procedure for modeling a transfer.*

CONTENTS

<u>Figure</u> <u>Page</u>

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

The purpose of this CCSDS report is to provide a tutorial for the Producer-Archive Interface Specification (PAIS) standard.

[NOTE – this section should be eventually reviewed at the very end of the editing of this document]

1.2 RATIONALE

[SCOPE – general tutorial based on practical examples; all PAIS XML elements covered]

1.3 DOCUMENT STRUCTURE

This document is broken down as follows:

Section 1 defines the purpose, scope, structure, definitions for terminology and references to standards and external documents used in this CCSDS report;

Section 2 introduces the concept of transfer, the terminology inherited from PAIS and OAIS, and provides an overview and guidelines for building a Model of Objects for Transfer (MOT) and associated SIPs;

Section 3 introduces the PAIS XML descriptors and provides concrete examples covering all XML elements and provides best practices;

Section 4 describes the PAIS implementation of SIPs their generation, ingestion, validation, either as XFDU or non-XFDU packages;

Section 5 provides a series of use cases that enforce the understanding of the PAIS standard through concrete and complete examples

Section 6 introduces a series of existing software tools that may help implement the PAIS standard.

Annexes:

– Annex A contains the complete PAIS descriptors…
– [NOTE – To Be Continued]

1.4 DEFINITIONS

1.4.1 ACRONYMS AND ABBREVIATIONS

For the purposes of this document, the following acronyms and abbreviations apply.

AIP	Archival Information Package
ASCII	American Standard Code for Information Interchange
CCSDS	Consultative Committee for Space Data Systems
CMC	CCSDS Management Council
CNES	Centre National d'Études Spatiales
CoRoT	COnvection ROtation et Transits planétaires (French); COnvection ROtation and planetary Transits (English)
DIP	**Dissemination Information Package**
ISEE	International Sun-Earth Explorer (international cooperative program)
ESA	European Space Agency
GUI	Graphical User Interface
MB	Mega Bytes
MOT	Model of Objects for Transfer
NASA	National Aeronautics and Space Administration
OAIS	Open Archival Information System
PAIS	Producer Archive Interface Specification
PAIMAS	Producer-Archive Interface Methodology Abstract Standard
SIP	Submission Information Package
XFDU	XML Formatted Data Unit
XML	Extensible Markup Language
ZIP	[TBD]

1.4.2 GLOSSARY OF TERMS

PAIS terminology as defined in reference [1] is used throughout this CCSDS Report. Only brief definitions are provided here.

Archive: An organization that intends to preserve information for access and use by a Designated Community.

Content Information: The set of information that is the primary target for preservation. It is an Information Object comprised of its Content Data Object and its Representation Information. An example of Content Information could be a single table of numbers representing, and understandable as, temperatures, but excluding the documentation that would explain its history and origin, how it relates to other observations, etc.

Collection Descriptor: A set of attributes that describes a view of a single collection of data and that identifies the parent collection of which it is a part.

Data Object: Either a Physical Object or a Digital Object.

Data Object Type: A set of characteristics describing a Data Object (such as the size of this object and the description of its content). Typically there will be multiple Data Objects conforming to the same Data Object Type.

Descriptor: Either a Collection Descriptor or a Transfer Object Type Descriptor.

Descriptor Model: A model that defines the mandatory and optional attributes needed for a Collection Descriptor or a Transfer Object Type Descriptor.

Fixity Information: The information which documents the authentication mechanisms and provides authentication keys to ensure that the Content Information Object has not been altered in an undocumented manner.

Formal Definition Phase: The Formal Definition Phase includes completing the SIP design with precise definitions of the digital objects to be delivered, completing the Submission Agreement with precise contractual transfer conditions such as restrictions on access and establishing the delivery schedule.

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).

Information Package: A conceptual container composed of optional Content Information and optional associated Preservation Description Information. Associated with this Information Package is Packaging Information used to delimit and identify the Content Information and Package Description Information used to facilitate searches for the Content Information.

Model: A data entity described independently from any instance in a data product, and corresponding to a re-usable data entity definition, from which other data entities may inherit the attributes and apply some specialization rules.

Model of Objects for Transfer (MOT): The set of all Descriptors for a given Producer-Archive Project. It is used jointly by the Producer and the Archive to provide a common and understandable hierarchical view of the Producer's Data Objects to be transferred and their organization into collections, and it supports possible additional relationships among them. The hierarchy may be viewed as a tree having leaf and non-leaf nodes. The Data Objects to be transferred, organized as 'Transfer Objects,' are represented by the leaves of the MOT. Thus the nodes of the MOT have a different meaning depending on whether they are leaves or not:

– A leaf node corresponds to a single Transfer Object Type and therefore one exists for each Transfer Object Type Descriptor.

– A non-leaf node corresponds to a collection view of Transfer Object Types, or of a collection of collections. A non-leaf node exists for each Collection Descriptor.

Producer: The role played by those persons or client systems who provide the information to be preserved. This can include other OAISes or internal OAIS persons or systems.

Producer-Archive Project: A Producer-Archive Project is a set of activities and the means used by the information Producer as well as the Archive to ingest a given set of information into the Archive.

Submission Information Package (SIP): An Information Package that is delivered by the Producer to the OAIS for use in the construction or update of one or more AIPs and/or the associated Descriptive Information.

Transfer Object: A set of one or more Transfer Object Groups containing at least one Data Object that are to be transferred to the Archive.

Transfer Object Group: A set of zero or more Data Objects and zero or more Transfer Object Groups.

Transfer Object Type: A set of characteristics describing a Transfer Object (such as the size of this object, the description of its content, and its makeup in terms of one or more Data Object Types). Typically there can be multiple Transfer Objects conforming to the same Transfer Object Type.

Transfer Object Type Descriptor: A set of attributes that describes a Transfer Object Type and that identifies the parent collection of which it is a part.

Transfer Phase: The Transfer Phase performs the actual transfer of the SIP from the Producer to the Archive and the preliminary processing of the SIP by the Archive, as it is defined in the agreement.

Validation Phase: The Validation Phase includes the actual validation processing of the SIP by the Archive and any required follow-up action with the Producer. Different systematic or in-depth levels of validation may be defined. Validations may be performed after each delivery, or later, depending on the validation constraints.

1.5 CONVENTIONS

[SCOPE – this section should specify the diagramming conventions selected for this report, e.g., UML like notation, colors, etc., all when stabilized and agreed by the WG]

1.6 REFERENCES

1.6.1 NORMATIVE TRACK DOCUMENTS

The following publications contain provisions which, through reference in this text, constitute provisions of this Report. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this Report are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS publications.

[1] *Producer Archive Interface Specification (PAIS)*. Draft Recommendation for Space Data System Standards, CCSDS 651.1-R-1. Red Book. Issue 1. Washington, D.C.: CCSDS, October 2013.

[2] *XML Formatted Data Unit (XFDU) – Structure and Construction Rules*. Recommendation for Space Data System Standards, CCSDS 661.0-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, September 2008. [Equivalent to ISO 13527:2010.]

[3] *Producer-Archive Interface Methodology Abstract Standard (PAIMAS)*. Recommendation for Space Data System Standards, CCSDS 651.0-M-1. Magenta Book. Issue 1. Washington, D.C.: CCSDS, May 2004. [Equivalent to ISO 20652:2006.]

[4] *Reference Model for an Open Archival Information System (OAIS)*. Recommendation for Space Data System Standards, CCSDS 650.0-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, January 2002. [Equivalent to ISO 14721:2003.]

1.6.2 NON-NORMATIVE AND ADMNISITRACTIVE TRACK DOCUMENTS

The following documents are referenced in this Report. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Report are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS documents.

[5] *Organization and Processes for the Consultative Committee for Space Data Systems*. CCSDS Record, CCSDS A02.1-Y-3. Yellow Book. Issue 3.

[6] *CCSDS Publications Manual*. CCSDS Record, CCSDS A20.0-Y-3. Yellow Book. Issue 3. Washington, D.C.: CCSDS, December 2011.

2 PAIS AT A GLANCE

It will enable the Producer to share with the Archive a sufficiently precise and unambiguous formal definition of the Digital Objects to be produced and transferred, by means of a model. It will also enable a precise definition of the packaging of these objects in the form of Submission Information Packages (SIPs), including the order in which they should be transferred.

The primary objective of the Producer-Archive Interface Specification (PAIS) standard is to provide concrete XML files supporting the description and the control of transfers from a Producer to an Archive.

A transfer, as seen by the PAIS standard, is the circulation of Data Objects from a Producer to an Archive. The Data Objects are not transferred as independent plain items but rather grouped and encapsulated in higher level objects known as Submission Information Packages (SIPs) providing better control in term of content types, fixity information, inter-relationships and sequencing as outlined in the following figure 2-1.

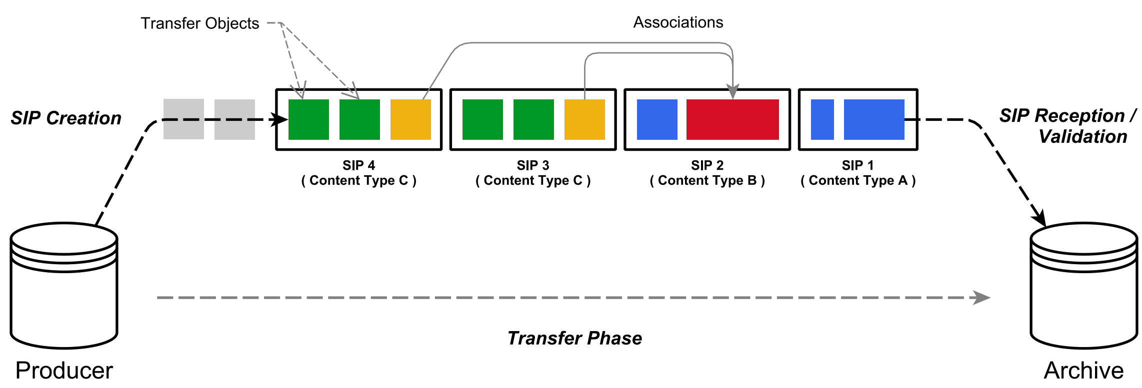

Figure 2-1: Example of Transfer

The Producer is responsible for the creation of SIPs according to content types agreed with the Archive and for their submission in a sequencing order that may also have been negotiated with the receiving Archive. In the example above, the Producer has generated and submitted four SIPs, one of Content Type A, the second of Content Type B and the remainders of Content Type C. As suggested by their names, the Content Types govern the actual content allowed for a SIP in term of structure and data format.

According to the PAIS standard the contents of the SIPs are decomposed in Transfer Objects (i.e. depicted as colored boxes in the figure 2-1 above) holding one or more trees of Groups (i.e. usually denoting folders) organizing the Data Objects subject of the transfer (i.e. usually a single file or a small set of files). A typical example of Transfer Object could be an Earth Observation product composed of various metadata and data files (i.e. the Data Objects) organized in a tree of folders (i.e. the Groups). The PAIS standard supports the control of these objects through the description of their types, namely the Transfer Object Types, Group Types and Data Object Types.

According to the PAIS, the definition of these Content Types is given by a “SIP Constraints” XML document that can be as short as the following one:

```
<sipConstraints xmlns="urn:ccsds:schema:pais:1">
   <producerArchiveProjectID>MyProject</producerArchiveProjectID>
   <sipContentType>
      <sipContentTypeID>Content Type A❶</sipContentTypeID>
      <authorizedDescriptor>
         <descriptorID>Blue Descriptor ID❷</descriptorID>
         <occurrence>❸
            <minOccurrence>2</minOccurrence>
            <maxOccurrence>2</maxOccurrence>
         </occurrence>
      </authorizedDescriptor>
   </sipContentType>
</sipConstraints>
```

Table 2-1: Example of SIP Constraints Content

Node	Content
sipConstraints	
@xmlns	urn:ccsds:schema:pais:1
producerArchiveProjectID	MyProject
sipContentType	
sipContentTypeID	Content Type A❶
authorizedDescriptor	
descriptorID	Blue Descriptor ID❷
occurrence❸	
minOccurrence	2
maxOccurrence	2

This “SIP Constraints” document shall include all the Content Type definitions although only the Content Type A ❶ has been described in the example for simplicity. This Content Type A accepts only one Transfer Object Type identified as “Blue Descriptor ID” ❷. The example also defines that two and only two objects of this type are expected per SIP of this Content Type ❸. The “SIP Constraints” document can also define the sequencing constraints, for example, to force the transfer of SIPs of Content Type B prior to those of Content Type B.

The “Blue Descriptor ID” ❷ refers to a Transfer Object Type that has to be defined in a separate “Transfer Object Type descriptor” XML document as the following one:

```
<transferObjectTypeDescriptor xmlns="urn:ccsds:schema:pais:1">
   <identification>
      <descriptorModelID>CCSD0014</descriptorModelID>
      <descriptorModelVersion>V1.0</descriptorModelVersion>
      <descriptorID>Blue Descriptor ID❶</descriptorID>
   </identification>
```

```
    ...
    <groupType>❷
       <groupTypeID>Blue Group</groupTypeID>
       ...
       <dataObjectType>❸
          <dataObjectTypeID>Blue Data Object</dataObjectTypeID>
          ...
       </dataObjectType>
    </groupType>
</transferObjectTypeDescriptor>
```

The descriptor clearly declares the “Blue Descriptor ID” ❶ and the content tree composed of one “Blue Group” Group Type❷ holding one “Blue Data Objet” Data Object Type❸. Some parts of the example have been truncated and replaced by “…” for simplicity. Those parts (not all optional) are generally dedicated to the control of the occurrences, sizes and associations between the types. All elements of these descriptors and related concepts are addressed in the following sections of this report.

In addition, the PAIS standard specifies the minimal set of metadata that shall be attached to a SIP for the complete typing of all the objects it contains i.e. the mapping of the objects to the PAIS descriptor types. The PAIS standard also defines a default SIP format based on the CCSDS XFDU recommended standard (see reference [2]). According to the XFDU standard, the SIPs are containers of any type (i.e. usually a ZIP archive or a root folder), that hold the Data Object files organized as an arbitrary number of nested folders. This dataset structure is accompanied by an XFDU Manifest XML document that registers all the Data Objects and, when specialized as defined by the PAIS, univocally identifies their types in a PAIS Producer-Archive Project i.e. the PAIS Data Object Types, Group Types, Transfer Object Types, SIP Content Type, etc.

The list of methods for writing PAIS descriptors is endless and it is likely that none is suitable for all project contexts. . Nevertheless, the following workflow gives an overview of the major steps that are usually to be addressed during a project definition:

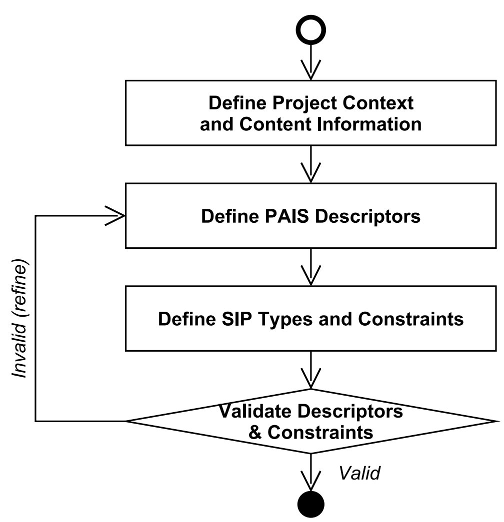

Figure 2-2: Typical steps driving a PAIS Producer-Archive Project definition

Finally, Producer-Archive Projects can benefit from the PAIS standard by writing a set of XML documents according to a formal XML language, validate these descriptors against XML Schema documents provided in the annex of the standard and develop or reuse tools for building, transferring, receiving and validating SIPs.

3 MODELING TRANSFERS

The PAIS standard in reference [1] specifies material for the modeling and control of the transfer of Digital Objects from a Producer to an Archive. This material consists of a set of XML language specifications and construction rules supporting the concrete implementation of the Producer-Archive Interface Methodology Abstract Standard (PAIMAS), itself deriving from the Reference Model for an Open Archival Information System (OAIS) (see references [3] and [4]). The XML schemata defined by the PAIS do not intend to implement all aspects of the wide PAIMAS abstract standard but are focused on the specification of the Model of Objects for Transfer and the associated SIPs during the Formal Definition Phase, the validation and follow-up of the SIPs during the Transfer Phase and the Validation Phase as defined in PAIMAS.

3.1 MODEL OF OBJECTS FOR TRANSFER

The Model of Objects for Transfer (MOT) is a set of PAIS XML descriptors controlling a Producer-Archive Project. It shall be composed of at least one Collection Descriptor and one Transfer Object Descriptor as shown in the figure below.

The MOT is a logical concept that has no default physical representation enforced by the PAIS standard. A MOT could be represented by a file system folder or a ZIP archive containing the PAIS XML descriptors, a manifest of whatever format referencing the descriptors or any other sort of gathering technique.

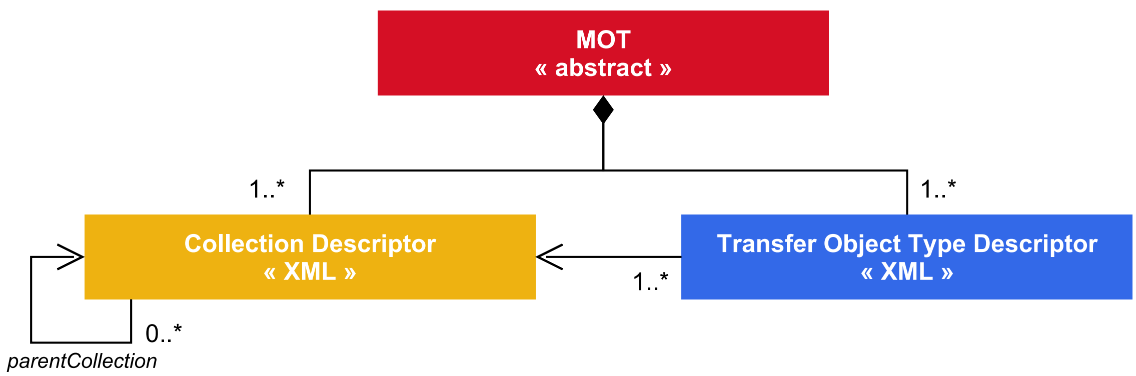

Figure 3-1: Model of Objects for Transfer (MOT)

Each Transfer Object Type Descriptor models a unit of transfer broken down in a tree of Group Types and sub-Group Types of Data Objects. The Transfer Object Type Descriptor is further described in section 3.1.1.

The Collection Descriptors organize the Transfer Object Types in a logical tree, or more formally a forest, of an unlimited levels of parent Collections. The Collection Descriptor is further described in section 3.1.2 below.

According to this definition, the minimal MOT is composed of two XML files and defines one Transfer Object Type, part of one Collection.

3.1.1 TRANSFER OBJECT TYPE DESCRIPTOR

A Transfer Object Type Descriptor defines a type of Objects, i.e. a Transfer Object Type, that can be transferred from the Producer to the Archive. Each descriptor defines one and only one Transfer Object Type that has to be part of a Collection.

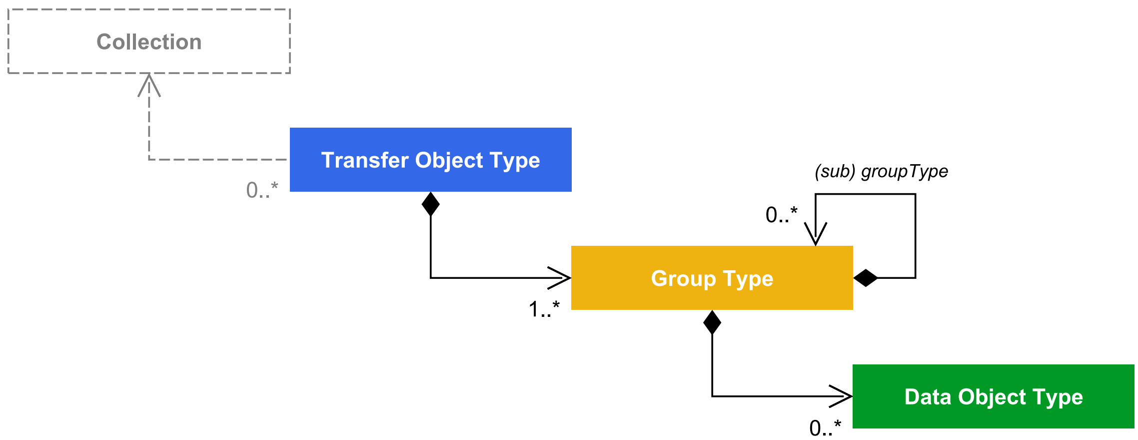

Figure 3-2: Transfer Object Descriptor

A Transfer Object Type is univocally identified across the overall Producer-Archive Project and decomposes the object in one or more trees of Group Type nodes and Data Object Type leaves. A Transfer Object Type defines the minimal and maximal number of objects of this type that could occur in the overall project. It may optionally define the minimal and maximal size of an object of this type. The complete specification of the Transfer Object Type is provided in section 3.2 of PAIS standard (see reference [1]) and illustrated throughout the section 4 of this report.

A Transfer Object Type must contain at least one Group Type, all accepting as many sub-group types as required up to the Data Object Type leaves. The Group Type definition specifies the minimal and maximum number of occurrences expected within its parent Group or its parent Transfer Object. Depending on configurable criteria, the Group Types may correspond to physical folders or ZIP entries. They may also represent a purely logical group with no physical counterpart.

The Data Object Type is the lowest level of description of the MOT and usually corresponds to a single file type. It may also represent multiple files if this set of files can be considered as a single Data Object at the transfer level. This feature avoids data descriptions considered unnecessary or too fine for a given Producer-Archive Project. For example, it may be convenient to consider an Earth Observation product as a single Data Object although it is composed of multiple files. The modeling of the header files, the image bands and other auxiliary files composing this product may not be of interest if they are never disjoint and never referenced individually during the transfer. The Data Object Type specifies the minimal and maximal number of occurrences expected within its instance of parent Group Type. It is also possible to control the minimal and maximal number of files composing a Data Object.

Any of the Transfer Object Types, Group Types and Data Object Types can be interrelated or associated to a Collection. The semantics of these associations are free and are to be agreed between the Producer and the Archive during the Project definition. Typical semantics include the data/metadata relationship or the association with a representation information that may support the cataloguing and the validation activity on the Archive side. These relations are further discussed in section 4.5 of this report.

3.1.2 COLLECTION DESCRIPTOR

Each Collection Descriptor defines one Collection that references a parent Collection (except for the root node for which the parent Collection is “none”) and that can be referenced by any number of Transfer Object Types. Thus the set of Collections defined in a MOT forms a forest i.e. a series of trees. It is not formally prohibited to have “leaves” Collections in a MOT, i.e. Collections with no child Collection or Transfer Object Type, but this is not a recommended practice since those Collection have no interest for a transfer.

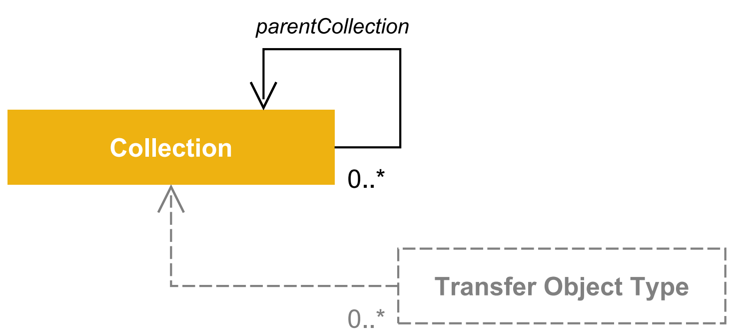

Figure 3-3: Collection Descriptor

A Collection is a logical grouping concept whose semantics are to be determined by the Producer and the Archive. For example, the Collection may drive the Archive behavior in term of cataloguing, validation means or storage. Typical Collections could distinguish project documentation from data production, could group products according to their processing levels, the production phases, etc. The complete specification of Collection is provided in section 3.3 of PAIS standard (see reference [1]) and illustrated throughout the section 4 of this report.

3.2 SUBMISSION INFORMATION PACKAGE (SIP)

The Submission Information Package (SIP) is the actual physical unit of transfer that can hold one or more Transfer Objects. The PAIS standard defines a SIP Constraints file and various rulesfor specifying and controlling the SIP contents based on the MOT elements introduced in the sections above.

3.2.1 SIP CONSTRAINTS

The SIP Constraints file is an XML document defining all the SIP Content Types that may occur in a Producer-Archive Project. At least one SIP Content Type must be defined for the project.

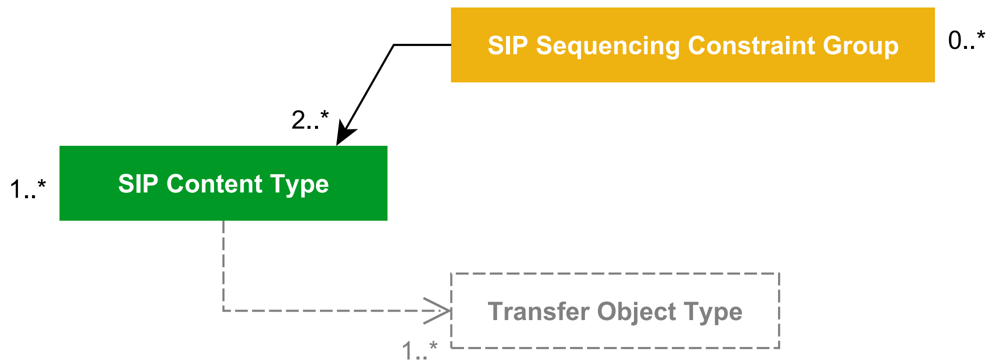

Figure 3-4: SIP Constraints

Each SIP Content Type specifies the Transfer Object Types authorized for this type of SIP and, optionally, their minimal and maximal numbers of occurrences allowed for each SIP instance.

The SIP Constraints file may also define one or more SIP Sequencing Constraint Groups that impose a sequencing order between two or more SIP Content Types. This feature may be convenient to secure the prior transfer of representation information required for the validation of the successive packages.

3.2.2 SIP MODEL

The PAIS standard defines an abstract SIP Model that has to be followed by any PAIS compliant SIP. Unless specified otherwise, any reference to SIP in this report is supposed compliant to the PAIS SIP Model.

According to this SIP Model, a SIP is composed of one mandatory SIP Global Information section and optional Transfer Objects to Delete and Transfer Object sections as outlined in the figure 2-6 below. At least one of the two sections is mandatory.

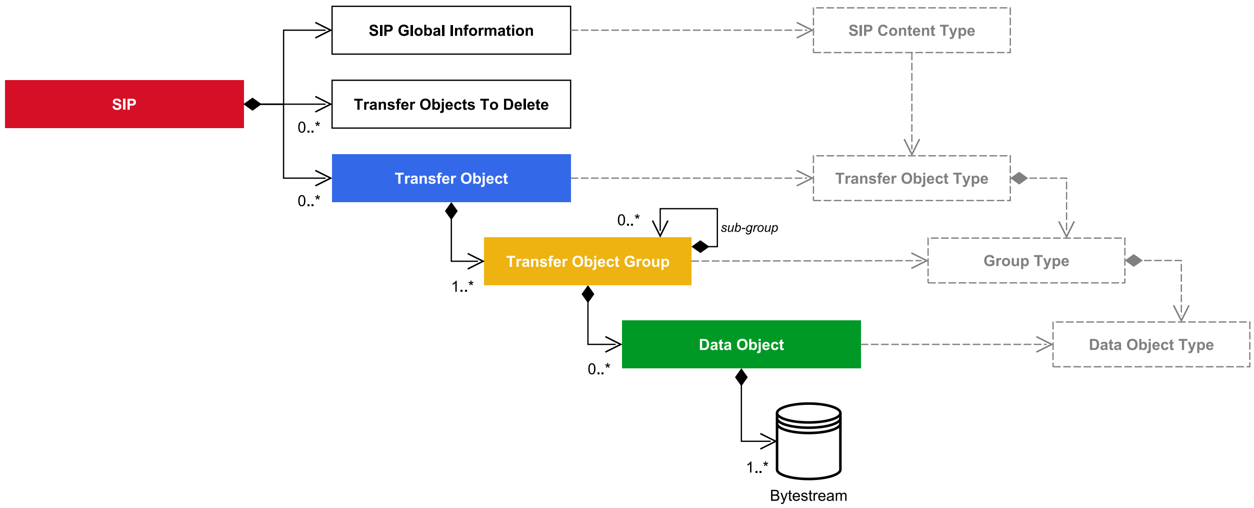

Figure 3-5: SIP Model

The SIP Global Information includes the identification of the SIP with respect to the Producer-Archive Project, the actual source that has generated this SIP and, in particular, a reference to the SIP Content Type it instantiates. The SIP Global Information also includes a sequence number necessary for disambiguation when the MOT descriptors do not provide fixed occurrence values.

The Transfer Objects are instances of Transfer Object Types of the Producer-Archive Project's MOT. A Transfer Object of a SIP must be of a Transfer Object Type authorized by the SIP Content Type referenced by the SIP Global Information. Similarly, the number of Transfer Objects of an authorized type must be included in the range defined in the SIP Content Type.

Following the type hierarchy, the Transfer Objects are composed of Transfer Object Groups, instances of Group Types, and Data Objects, instances of Data Object Types. The actual instances shall explicitly reference the corresponding MOT types to allow the formal validation of the SIP structure at ingestion.

Finally, the Data Objects reference one or more Byte Streams denoting the physical files or any alternative representation of the data, depending on the selected encoding of the Data Object or any parent Transfer Object Group.

3.3 A METHODOLOGY FOR MODELING A TRANSFER

As introduced in the section 2 above, there are probably countless methods that could apply for the implementation of PAIS descriptors, but it is likely that none is suitable for all project contexts. However, the following workflow diagram introduces a typical methodology illustrating major steps that most implementers will have to overcome.

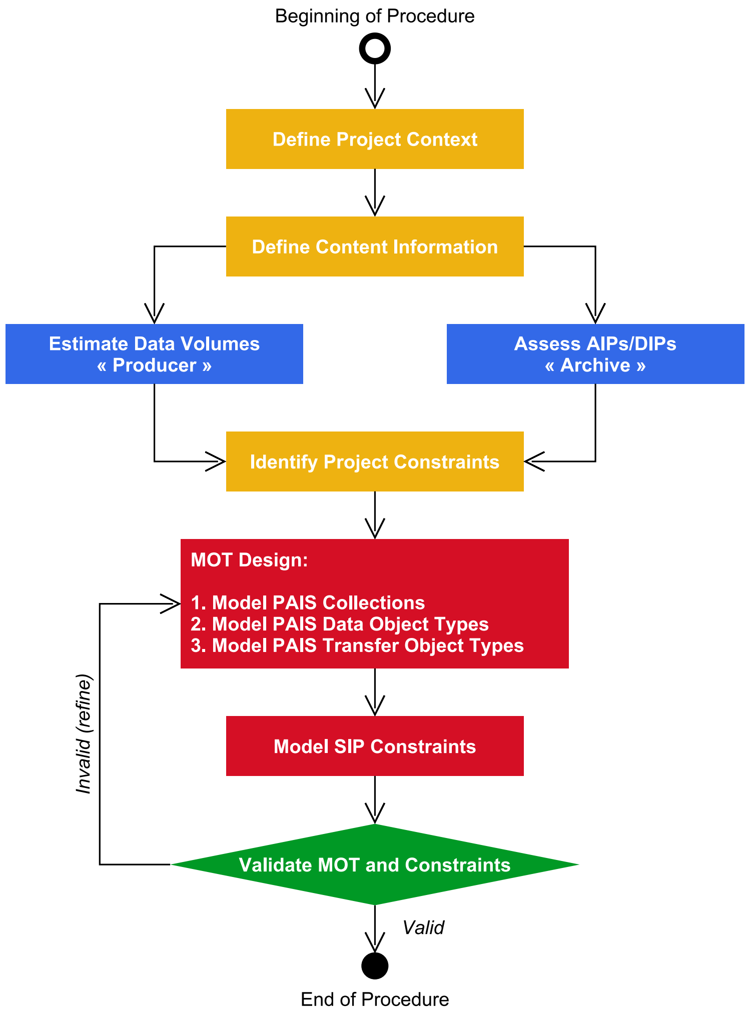

Figure 3-6: A Methodology for Modeling a Transfer

The workflow steps can be summarized as follows:

- **Define Project Context**: according to PAIMAS standard (see reference [3]), this step represents a preliminary phase that can be long during which the Producer and the Archive converge towards a common project of transfer. At PAIS level, the critical outputs are the Producer-Archive Project identifier and the potential Producer Source identifiers that will be necessary in the header of most of the PAIS descriptors and associated XML documents;

- **Define Content Information**: at this step the Producer and the Archive agree on what kind of Digital Objects are to be transferred. From this step, the workflow forks in two branches involving the Producer and the Archive separately;

- **Estimate the Data Volumes**: on the one hand the Producer determines the volumes of each type of Digital Objects. This may depend on the variety of formats, of data sources, etc.

- **Assess AIPs/DIPs**: on the other hand, the Archive has to consider how and where it will host/store the Digital Objects in term of Archival Information Packages (AIPs). A preliminary analysis of the Dissemination Information Packages (DIPs) is usually a good practice for the proper design of AIPs. This step may also identify the validation procedures that will be required at ingestion to safelypopulate the archive and guarantee a good quality of dissemination in output of the Archive;

- **Identify Project Constraints**: this steps consists in merging the merge of the information collected during the two previous parallel steps and _confronting them with the common resources that all parties can set up for the planned transfer e.g. network resources, hardware, manpower, etc. The objective is to identify the project constraints that can drive and influence the modeling of the objects to be transferred in term of grouping, sizing and sequencing. The establishment of a bulk transfer of a repository of small documents may actually be very different from a transfer of large measurements to be acquired from a spacecraft as the mission is in progress. The former requires small throughput for a short period of time whereas the latter requires better network performances and an availability in the longer term;

- **MOT Design**: at this step, all the project information (e.g. data objects, formats, collections etc.) and constraints are known and it becomes possible to model the transfer and write the PAIS descriptors for the transfer project i.e. the Collection descriptors and the Transfer Object Type descriptors. It is recommended at this step to model only the elements that are actually required or justified for the transfer. The PAIS does not intend to be a data format descriptor that may serve as a Representation Information outside the context of a transfer. Thus, modeling all files directories because they are there may be more cumbersome than useful and should not be considered as a good practice. The Table 3-1 below provides a series of justifications that may help to develop optimal models;

- **Model SIP Constraints**: this step consists in writing the SIP Constraints XML document that defines the SIP Content Types allowed in the transfer and potentially the sequencing constraints e.g. SIPs holding data specifications or auxiliary data may be required first to allow the validation or the reprocessing of the primary data subject of the transfer;

- **Validate MOT and Constraints**: at this decision step the PAIS descriptors and the SIP Constraints XML documents are validated. They have to be well-formed according to the XML recommendation and validated against the PAIS XML Schemas provided in the annex of the standard. It is recommended to complete this automatic validation with careful reviews involving both the Producer and the Archive and to simulate the transfer of all potential SIP Content Types in an environment as close as possible to the target operational one. In case of failure, the workflow must be reiterated from the "MOT Design" step.

Table 3-1: Recommended Justification of PAIS elements

PAIS Element	Justification
Collection	Because at least one Collection description is required per project
	Because it is the only way for the Archive to be able to sort or classify the received data e.g. the Archive may behave differently upon reception of Representation Information or Data or Documents, …
Data Object Type	Because no other existing Data Object Type can represent the file
	Because an occurrence constraint is to be controlled during the transfer or is needed by the Archive for estimating the number of files before their actual transfer and cataloguing
	Because the format is to be determined a priori or is needed by the Archive e.g. for data format validation
	Because it is the source or the target of a relation/association needed by the Archive e.g. for validation, processing, etc.
	Because a specific Producer information is to be attached to the Data Object Type as an extension point
Group Type	Because at least one Group Type is required per Transfer Object Type
	Because

4 WRITING XML DESCRIPTORS

[TOC Provided temporarily for information]

[NOTE – The following paragraphs beginning with TOPIC are subjects and concepts to be addressed. They are provided to support discussions and are supposed to be removed in a successive version of the document.]

TOPIC – This section describes the XML descriptors composing the MOT (Collections and Transfer Object Descriptors). Basic understanding of the MOT concepts introduced in the section 3.1 above is required.

4.1 STRUCTURES AND CONSTRUCTION RULES

TOPIC – Based on POLDER Data Set, see section 6.2. This may optionally be introduced here but very briefly as the derived examples/samples of the following sections shall be self-explanatory.

4.1.1 ORGANIZATION XML DOCUMENTS

TOPIC – The physical organization of XML documents is basically free e.g. as files or database entry, in the same directory or in separate ones, etc.

TOPIC – Some examples could however help figuring out how XML documents could be organized

TOPIC – An example: set of XML files grouped in a directory (diagram and naming rules to be updated)

```
├── polder-pais-collection-l0.xml
├── polder-pais-collection-l1.xml
├── polder-pais-collection-polder.xml
├── polder-pais-sip-constraints.xml
├── polder-pais-transfer-object-aux-data.xml
├── polder-pais-transfer-object-l0-data.xml
├── polder-pais-transfer-object-l0-repinfo.xml
├── polder-pais-transfer-object-l1-data.xml
├── polder-pais-transfer-object-l1g1-data.xml
├── polder-pais-transfer-object-l1gb-data.xml
├── polder-pais-transfer-object-l1-repinfo.xml
└── polder-pais-transfer-object-project-doc.xml
```

Figure 4-1: Example of PAIS XML documents in a directory

TOPIC – Example of naming convention (proposal):

```
<project-id>-pais-<type>[-<id>].xml
```

Where:

`<type>` can either be `collection`, `transfer-object` or `sip-constraints`;

`[-<id>]` is either the `descriptorId` of the Collection or Transfer Object Descriptors or nothing for SIP Constraints files that is unique.

4.1.2 XML NAMESPACE

TOPIC – PAIS descriptors are standard XML 1.0 documents that may begin with a standard XML prolog.

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

TOPIC – Current version of the PAIS produces elements in the "`urn:ccsds:schema:pais:1`" namespace. As for any XML document, it could be used as the default namespace i.e. non-prefixed PAIS elements, or with explicit namespace prefix i.e. preference to "`pais`" prefix but any other token would work.

TOPIC – As much as possible, it is recommended to use the **default namespace** (especially because this form is easier to read by humans).

TOPIC – Example of namespace usage – **default namespace**

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

<collectionDescriptor xmlns="urn:ccsds:schema:pais:1">
   <identification>
      <descriptorModelID>CCSD0015</descriptorModelID>
```

```
    […]
</collectionDescriptor>
```

TOPIC – Example of namespace usage – **namespace prefix** – may be useful to distinguish PAIS elements from other from different namespaces (only valid for extended PAIS descriptors)

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

<pais:collectionDescriptor xmlns:pais="urn:ccsds:schema:pais:1">
   <pais:identification>
      <pais:descriptorModelID>CCSD0015</pais:descriptorModelID>
      […]
</pais:collectionDescriptor>
```

TOPIC – The prolog and namespace declarations will not be repeated along the examples of this section. Tabular representation is preferred.

4.1.3 COLLECTIONS

TOPIC – One XML document per Collection descriptor

TOPIC – Any Transfer Object must have a parent Collection

TOPIC – Example of root collection: POLDER Collection

Table 4-1: Example of Root Collection

Element	Sample Value
collectionDescriptor ❶	
Identification	
descriptorModelID	CCSD0015
descriptorModelVersion	V1.0
descriptorID	POLDER
description	
collectionTitle	POLDER 1 and 2 Products
collectionDescription	POLarization and Directionnality of the Earth's Reflectance
Relation	
parentCollection ❷	NONE

TOPIC – All elements in the table are mandatory. This example corresponds to the minimal set of elements required for a collection. Collection descriptors accept a few more optional elements but this example would fit most of use cases.

TOPIC – `descriptorModelID` and `descriptorModelVersion` univocally identify the PAIS specification. Should a PAIS version break the collection descriptor structure and construction rules, these identifiers would have be different from `CCSD0015` and `V1.0`.

TOPIC – The `descriptorModelID` and `descriptorModelVersion` are defined in the PAIS XML Schema as unrestricted `xs:string` allowing empty strings, tabs etc. A good practice would be to restrict these elements to the values expected for the Producer-Archive project as described in section 4.7 below.

ACTION – (from MM) Explain more the CCSD0015: how it is registered and reference where more information could be found on that subject.

TOPIC – Any change in the CCSDS PAIS XML Schema (model ?) requires an update of the `descriptorModelID`.

TOPIC – `descriptorID` univocally identifies the Collection within the Producer-Archive project. The recommended practice is to consider distinct values for whatever type of ID of the project MOT and SIP Constraints e.g. Collection, Transfer Object Type, Group Type, etc. The descriptor identifier is defined in the XML Schema as an `xs:string` which authorizes empty strings or whatever content of any length. As for `descriptorModelID`, a recommended practice is to restrict the string to a controlled space for example throughout a pattern or an enumeration valid for the overall project. At least one that reject empty strings. See section 4.7.

TOPIC – `collectionTitle` and `collectionDescription` can greatly help the understanding of the purpose and scope of the Collection and should be written with care. Empty strings or duplicate of Collection identifier is to be considered as a poor practice. Formal projects could allow restrict the PAIS XML Schemas to prevent those situations.

TOPIC – A child collection: L0 collection

Table 4-2: Example of Child Collection

Element	Sample Value
`collectionDescriptor`	
… removed for brevity …	
`relation`	
`parentCollection` ❶	**`POLDER`**

TOPIC – The `relation/parentCollection` element identifies the parent Collection through its `descriptorID`.

TOPIC – A collection can be associated with other collections or other PAIS entities. Forward to section 4.5.

TOPIC – Collections are logical entities that do not have payload by default but this could be achieved through an “any” child element. Forward to section 4.7.2.

4.1.4 TRANSFER OBJECTS

TOPIC – Introduce the section.

Table 4-3: Example of Transfer Object Type for a series of L0 Products

Element	Sample Value
transferObjectTypeDescriptor ❶	
identification	
descriptorModelID	CCSD0014
descriptorModelVersion	V1.0
producerSourceID	CNES
descriptorID	L0DATA
description	
transferObjectTypeTitle	POLDER Level 0 Transfer Object
transferObjectTypeDescription	A single POLDER Level 0 product (TAR)
transferObjectTypeOccurrence	
minOccurrence	1
maxOccurrence	3
namePreservationRule	*Use the Source names*
relation	
parentCollection	L0
groupType	
groupTypeID	L0GROUP
groupTypeStructureName	directory
groupTypeOccurrence	
minOccurrence	1
maxOccurrence	1
dataObjectType	
dataObjectTypeID	L0DATAOBJECT
dataObjectTypeOccurrence	
minOccurrence	1
maxOccurrence	1

TOPIC – identification child as for Collections identification described in the previous section apart for the new and optional producerSourceID.

TOPIC – `producerSourceID` is referring to an entity of the Producer that helps identifying univocally the Producer or classifying the objects according to the sources. When used, propose a restriction to an enumeration.

TOPIC – `description` is similar to the Collection for the `collectionTitle` and `collectionDescription`. The `transferObjectTypeOccurrence` controls the number of occurrences of the Transfer Objects of this type **among the overall Producer-Archive project**. Refer to section 4.2 for further explanations about occurrences.

ACTION – Determine if the `namePreservationRule` element should be addressed here or in a separate section (4.x ?)

TOPIC – The `relation/parentCollection` element identifies the parent Collection through its `descriptorID`.

TOPIC – A Transfer Object Type can be associated with other PAIS entities. Forward to section 4.4.

TOPIC – Reminder: Tranfer Object Type is broken down in a tree of `groupType`'s and `dataObjectType`'s. It contains at least one `groupType`. Reference section 3.1.1.

TOPIC – The `goupType` element and its children are described in the following section 4.1.5.

TOPIC – The `dataType` element and its children are described in the following section 4.1.6.

4.1.5 TRANSFER OBJECTS – GROUP

TOPIC - Introduce the section

Table 4-4: Example of Group Type

Element	Sample Value
transferObjectTypeDescriptor	
… removed for brevity …	
groupType ❶	
groupTypeID	L0GROUP
groupTypeDescription	*Level 0 Group Type*
groupTypeStructureName ❷	directory
groupTypeOccurrence	
minOccurrence	1
maxOccurrence	1
groupType ❸	
… removed for brevity …	

dataObjectType ❹	
… removed for brevity …	
dataObjectType ❺	
… removed for brevity …	

TOPIC – Complete obvious fields description

TOPIC – A group may or may correspond to a physical entity.

TOPIC – Even if a physical structure exists, the tree of groups may not match exactly the physical structure. Actually, the tree of groups may be interleaved with logical groups not present in the physical structure. The tree of groups may also skip some levels of the physical structure. See section 3.3.

TOPIC – Usually, the `groupTypeStructureName` is "`directory`" when the groups of this type correspond to a physical file system folder or equivalent as folders in a ZIP or Tar. Otherwise, the "`set`" structure is generally adequate for most cases of logical grouping.

TOPIC – For interoperability purpose, it is recommended to consider the following syntax for the structure names defined in the PAIS BB: "`directory`", "`set`", "`sequence`" and "`undescribed`" i.e. lowercase.

TOPIC – It is recommended to restrict the `groupTypeStructureName` to a limited enumeration.

TOPIC – Complete `groupTypeStructureName` definition derived from PAIS BB including sequence and undescribed.

ACTION – Examples and explanations from Don ?

TOPIC – A `groupType` accepts an unlimited number of nested `groupType`'s. The depth of nesting is also unlimited.

TOPIC – A `groupType` accepts an unlimited number of `dataObjectType`'s.

TOPIC – More about occurrences in section 4.3.

TOPIC – A `groupType` also accepts other elements e.g. encoding, associations

TOPIC – Groups can be encoded cf. section 0.

TOPIC – Groups can be associated to other PAIS objects cf. section 4.5.

4.1.6 TRANSFER OBJECTS – DATA OBJECTS

Table 4-5: Example of Data Object Type

Element	Content
`transferObjectTypeDescriptor`	
… removed for brevity …	
`groupType` ❷	
… removed for brevity …	
`dataObjectType` ❶	
`dataObjectTypeID`	`L0DATAOBJECT`
`dataObjectDescription`	*`Level 0 Data Object`*
`dataObjectTypeOccurrence`	
`minOccurrence`	`1`
`maxOccurrence`	`1`

TOPIC – Describe obvious elements

TOPIC – A `dataObjectType` element much have a parent `groupType`

4.2 MANAGEMENT OF MOT IDENTIFIERS

ACTION – Move ID rulings introduced in the previous section to the current one.

TOPIC – Apart for `producedSourceID` (TBC) that is not an internal ID, all IDs have to be unique across the overall Producer-Archive project and therefore across all XML descriptors composing a MOT. This is a mandatory constrain because the IDs shall be referenced from outside the MOT and especially from the SIPs. The ID references from outside the MOT shall univocally identify the target resource independently from its semantic.

TOPIC – The IDs of the MOT are not typed as xs:ID because it would have been of no help for guarenteeing of uniqueness across multiple XML documents. It could have even introduced a wrong feeling of confidence to the users.

TOPIC – Methodology for verifying quickly the uniqueness would help e.g. xmllint, etc. All platforms and free solution should be preferred.

4.3 OBJECTS OCCURRENCES AND SIZES

This section describes how to control the occurrences of the Transfer Objects, Groups, Data Objects and Files of Data Objects. It also covers the control of the Transfer Objects size.

TOPIC – Focused on XML elements, not on PAIS occurrence logic. Cf. section 3.1.

4.3.1 OCCURRENCE TYPE

Table 4-6: Definition of Occurrence Type

Element	**Type**
occurrenceType	
minOccurrence	**xsd:nonNegativeInteger** ❶
❸ { **maxOccurrence**	**xsd:nonNegativeInteger**
maxUnknow	*- empty when used -* ❷

TOPIC - Describe minOccurrence shall have a positive or null integer value. Represents the inclusive minimum number of occurrences allowed for the objects. Null expresses an optional object.

TOPIC – Restricts nonNegativeInteger to a maximum suitable for computer. Basically xsd:int or xsd:integer.

TOPIC – Describe maxOccurrence:

The maxOccurrence shall have a positive or null integer value by definition. The null value may be confusing and could corresponds a situation where the objects of this type are unexpected, disabled or denied. This may help during the development of the PAIS descriptors or during some ad hoc situations. Otherwise needed, it is considered a good practice to restrict the maxOccurrence element to avoid 0 value, for example through the xsd:positiveInteger. See section 4.7.3 for more information about how to restrict types.

TOPIC – Describe maxUnknown:

The maxUnknow is an empty element when use. It expresses that number of occurrence is variable and virtually illimited.

The maximumOccurrence and maxUnknown are mutually exclusive so only one should be specified in the same parent element.

4.3.2 OCCURRENCE CONTROL USE CASES

Table 4-7: A bounded number of Transfer Objects

Element	**Value**
transferObjectTypeOccurrence	
minOccurrence	64 ❶
maxOccurrence	128 ❷

Table 4-8: An unlimited number of Groups

Element	Value
`groupTypeOccurrence`	
`minOccurrence`	`444`
`maxUnkown`	*- none -* ❶

Table 4-9: An optional Data Object

Element	Value
`dataObjectTypeOccurrence`	

minOccurrence	**0** ❶
maxOccurrence	1 ❷

Table 4-10: A denied Data Object

Element	**Value**
dataObjectTypeOccurrence	

minOccurrence	0 ❶
maxOccurrence	0 ❷

ACTION – What does a `maxOccurrence` of 0 exactly means ?

Table 4-11: A fixed number of Data Object Files

Element	**Value**
dataObjectTypeFileOccurrence	
minOccurrence	2 ❶
maxOccurrence	2 ❷

4.3.3 INVALID OCCURRENCE DEFINITIONS

Table 4-12: Wrong Negative Bound

Element	**Value**
dataObjectTypeFileOccurrence	
minOccurrence	-1 ❶
maxOccurrence	2

TOPIC - (?) Recommended practice about the interpretation of occurrences constraints (highlighted in ISEE use case)

4.3.4 TRANSFER OBJECT SIZES

4.4 OBJECTS ENCODINGS

TBD

4.5 OBJECTS RELATIONS

TBD

4.6 SEQUENCING CONTROL

TBD

4.7 CUTOMIZATION – EXTENSIONS AND SPECIALIZATIONS

TOPIC – Explain the interest of specializations i.e. restriction of the .

4.7.1 PAIS XML SCHEMAS

TBD – This section is necessary for the following but could have been placed at the very beginning of this chapter.

TOPIC – Display the PAIS schemas and their relationship.

TOPIC – Official CCSDS SANS XML Schemas repository: http://sanaregistry.org/r/daixml

TBD – Will this repository contain the XML Schemas at the first level or will it be structured with a TBD tree of sub-folders, for example to support multiple versions, or other material e.g. sample code/snipets of this tutorial, copies of software prototype, etc. ?

4.7.2 EXTENSIONS

TOPIC – This section explains how to deal with the “any” elements.

TOPIC – Remind the goals of the “any” elements.

Table 4-13: Example of Extended Collection

Element	Content
collectionDescriptor	
Identification	
descriptorModelID	CCSD0015
descriptorModelVersion	V1.0
descriptorID	POLDER
[…]	
relation	
parentCollection	NONE
any ❶	
myPayload ❷	
@xmlns ❸	urn:my:namespace
foo	
bar	

TOPIC – ❶ Allowed only once and accepts only **one** child ❷ with a namespace ❸ different than the one of the PAIS.

TBD – This example is abstract. Should it be replaced wih a more concrete one? Ok put for example the author, producer collection ID, hardware that produced the collection descriptor, etc.

TBD – Multiple examples could be provided for the “any” elements of the PAIS descriptors. All? Which one?

4.7.3 XML SCHEMA TYPE RESTRICTIONS – RECOMMENDED PRACTICE

TOPIC – The easiest means for the customization of the PAIS XML Schemas is to edit copies the originals. However, a free editing makes difficult the verification and validation of the conformity with the genuine definitions. XML Schema provides mechanism for that purpose through the `xs:redefine` element.

TOPIC – Describe how to restrict types through the `xs:redefine` directive.

TOPIC – `xs:redefine` does help for controlling explicit definitions as types or rules, but does not help for implicit ones such as a change of documentation/annotation.

Table 4-14: Example of Restricting XML Schema Declaration

Element	Content
xs:schema	
@xmlns:xs	http://www.w3.org/2001/XMLSchema
@xmlns:pais	urn:ccsds:schema:pais:1
@targetNamespace	urn:ccsds:schema:pais:1
@elementFormDefault	qualified
@attributeFormDefault	unqualified
xs:redefine ❶	
@schemaLocation	<somewhere>/ccsds-pais-common-types.xsd
[…]	
<restricted types> ❷	
[…]	

TOPIC – `xs:redefine` ❶ is very similar to `xs:include` statement.

TOPIC – Only the PAIS original types that need to be changed are to be redefined in ❷, not all.

TOPIC – In XML Schema the restriction is the standard mechanism for sub-typing, or controlling built-in type constraints e.g. pattern, ranges, etc. Restrictions are not dedicated to `xs:redefine` but are very common for XML Schemas that are not based only on direct reference to the built-in types.

Table 4-15: Example of Restricted Type – Complex Type

Element	Content
xs:complexType	
@name ❶	associationType
xs:complexContent	
xs:restriction	
@base ❷	pais:associationType
[…]	
<restricted content>	
[…]	

Table 4-16: Example of Restricted Type – Simple Type – String Patterns

Element	Content
xs:element	
@name	targetID
xs:simpleType	
xs:restriction	
@base	xs:string
xs:pattern ❶	
@value	POLDER_.*
xs:minLength ❷	
@value	10
xs:whiteSpace ❸	
@value	preserve

Table 4-17: Example of Restricted Type – Simple Type – String Patterns

Element	Content
xs:element	
@name	relationType
xs:simpleType	
xs:restriction	
@base	xs:string
xs:enumeration ❶	
@value	representationInformation
@value	dependency

TOPIC – Continue restriction example for numerical values, occurrence control, etc.

TBD – Introduce "open" enumeration technique e.g. an content-free attribute can be restricted to a enumeration including "other". This specific value would specify that the value is not part of the enumeration but should be found in another attribute, for example an `@otherAttribute`. This mechanism was successfully used in XFDU and SAFE.

4.7.4 RESTRICTING XXX

TBD – Describe restrictions at least for all rows of the following table. Any row more should correspond to a new sibling section "Resctricting XXX".

Element	Restrictions
Collection Descriptor	
descriptorModelID	fixed
descriptorModelVersion	fixed
descriptorID	pattern recommended, length > 1, space preserve
collectionTitle	length > 5, space preserve
collectionDescription	length > 5, space collapse
unitsType	restrict enum to minimal set e.g. MB or GB
relationType	restrict to actually used e.g. representationInformation, dependency, use, etc.
Transfer Object Type Descriptor	
descriptorModelID	fixed
descriptorModelVersion	fixed
descriptorID	pattern recommended, length > 1, space preserve
transferObjectTypeTitle	length > 5, space preserve
transferObjectTypeDescription	length > 5, space collapse
minOccurrence	restrict xs:nonNegativeInteger to a type applicable for the project and for the target hardware e.g. xs:short, xs:int, etc. May constrain to an actual minimum of minimum.
maxOccurrence	restrict xs:nonNegativeInteger to a type applicable for the project and for the target hardware e.g. xs:short, xs:int, etc. May constrain to an actual maximum of maximum
maxUnknown	deny if not applicable
minSize	Should be positibe or null
maxSize	May be restricted to the actual/total Archive capabilities or any intermediaet system that could not handle the size e.g. ZIP.
namePreservationRule	deny if not used or restrict to an enumeration
dataObjectTypeID	pattern recommended, length > 1, space preserve
dataObjectTypeFileOccurrence	deny if not required/supported
mimeType	restrict to an agreed set from IETF e.g. text/xml, etc.
registrationAuthority	deny if not accepted or restrict

	to an agreed enumeration
registeredID	deny if not accepted or restrict to an agreed enumeration or at a minimum, a pattern (not recommended)
To be continued	

TOPIC – Introduce a common policy for restricting all ID values among the project e.g. min one char, printable characters, pattern, etc.

5 BUILDING AND MANIPULATING SIPS

[SCOPE – all about SIPs]

[NOTE – the structure of this section is to be defined]

5.1 XFDU SIPS

The PAIS standard proposes one implementation of the recommended SIP Model based on the CCSDS XML Formatted Data Units (XFDU) packaging standard (see reference [2]).

As a brief introduction to the most relevant features of this standard, an XFDU package is a container, usually a ZIP archive file, composed of one Manifest XML file referencing all the other files of the package. The Manifest file consists of a series of sections among which a Package Header containing general information, a Information Package Map providing a logical break-down of the package content in a tree of nested Content Units, and a Data Object Section referencing all the packaged files.

The XFDU PAIS SIP definition makes use of the XFDU extendibility mechanisms to implement the SIP Model introduced in the previous section. This implementation is depicted in the following figure and completely defined in the PAIS standard (see section 6 of reference [1]).

NOTE –MIME types origin (MOT ?)

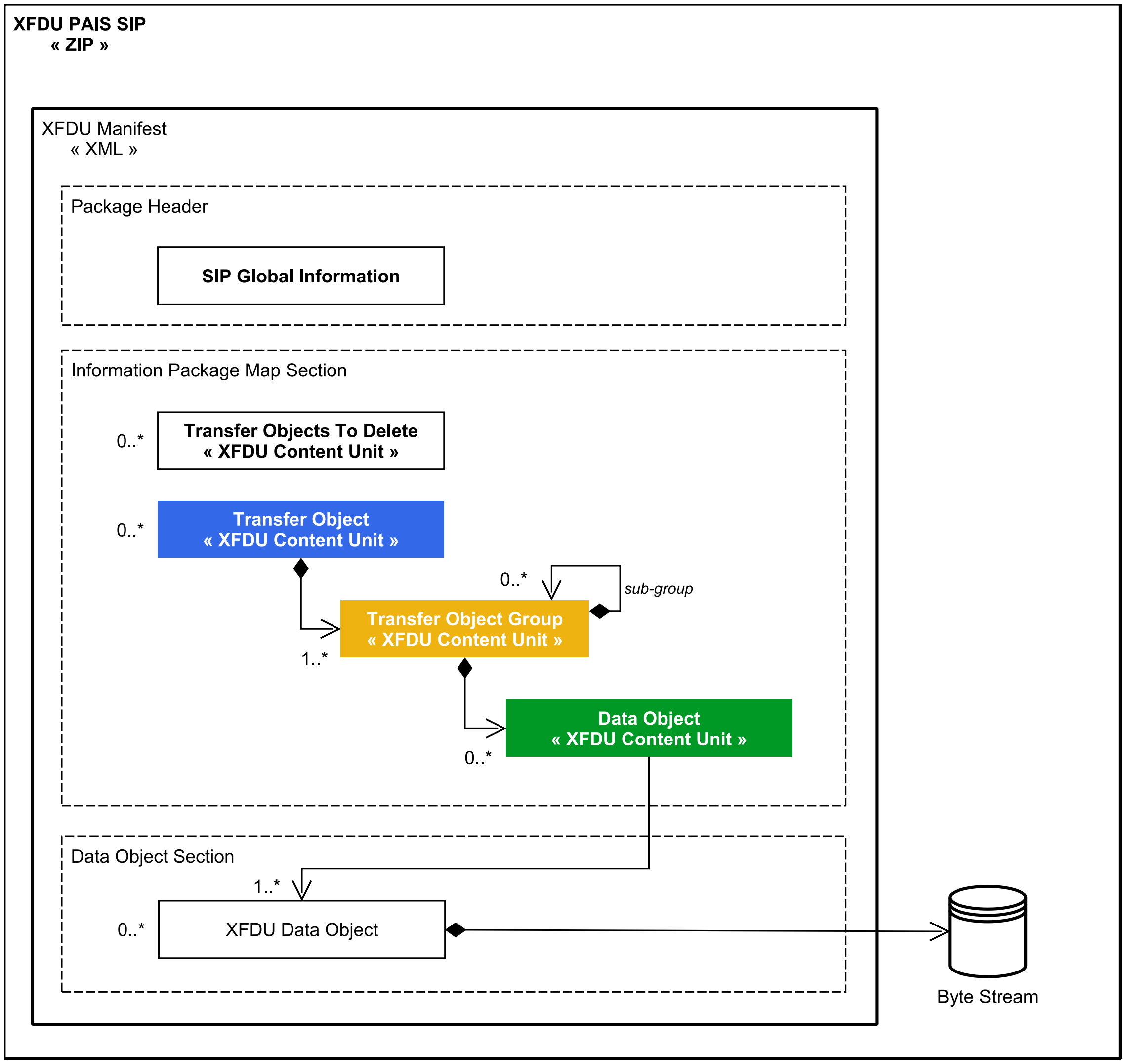

Figure 5-1: XFDU PAIS SIP

The SIP Global Information is implemented as an extension of the Package Header section of the XFDU Manifest.

The Transfer Objects to Delete are implemented as an extension of a Content Unit of the Information Package Map Section of the XFDU Manifest.

The Transfer Objects and their Transfer Object Groups and Data Objects children are also implemented as extensions of Content Unit of the Information Package Map Section. However, because the XFDU does not authorize the direct references to the packaged files, the Data Objects Content Units make use of so called Data Object Pointers to reference entries of the Data Object Section of the XFDU Manifest. The Byte Streams are then referenced from this latter Data Object Section.

TOPIC – Emphasize that elements of the Manifest refer to IDs of the Model e.g. TranferObject/descriptorID, associatedDescriptorGroupTypeID, associatedDescriptorDataObjectTypeID

TOPIC – Explain that SIP/…/transferObjectGroupInstanceName corresponds to the name of the goup (instance of GroupType). The transferObjectGroupInstanceName is mandatory only and only for group has a "directory" structure. The structure of a group is defined by its type's groupTypeStructureName element in the MOT. It is optional otherwise.

Practical examples of XFDU PAIS SIP implementation are provided an discussed in section 5 below. The section 5 below includes practical examples of XFDU PAIS SIPs that may help figuring out the

5.2 NON-XFDU SIPS

[SCOPE – any best practice inherited from a PDS4 study to be done.]

[NOTE – check with WG if this practice should be recommended in this document]

TOPIC – Not reuse SIP model elements as a baseline but start from scratch.

TOPIC – Propose an implementation checklist

6 USE CASES

This section registers a series of use cases that were generally elaborated during the development and validation of the PAIS standard. These examples do not claim to provide turnkey solutions for operations. They may, however, improve the understanding of the PAIS standard through concrete cases built from various aspects. They may also help implementers start up their projects with patterns and snippets they can arrange at their discretion.

6.1 ISEE – A TYPICAL USE CASE

[NOTE – Section temporarily moved to `651x2g0-[6.1]-isee.docx` document]

6.2 POLDER – REPRESENTATION INFORMATION USE CASE

[NOTE – Section temporarily moved to `651x2g0-[6.2]-polder.docx` document]

6.3 PLANETARY DATA SYSTEM – A NON-XFDU SIP IMPLEMENTATION

[NOTE – Section temporarily moved to `651x2g0-[6.3]-pds4.docx` document]

6.4 COROT – END OF MISSION BULK TRANSFER

[NOTE – Section temporarily moved to `651x2g0-[6.4]-corot.docx` document]

6.5 [ESA-SAFE-DATA-TBD] – TRANSFER OF SAFE PRODUCTS

[NOTE – Section temporarily moved to `651x2g0-[6.5]-safe.docx` document]

7 SOFTWARE TOOLS

[SCOPE – this section should introduce the prototypes and libraries. The non-operational status of the described software shall be noticed]

7.1 CNES PROTOTYPE

[SCOPE – this section is currently inherited from 651.3y0 record and needs to be integrated, e.g., the description of the PAIS descriptors is redundant in this document. The availability and licensing of this software should be addressed]

The CNES implementation was developed to cover the following functions during the following main phases of a Producer-Archive Project:

- During the Formal Definition Phase:
  - MOT creation and visualization:
    - MOT structure design;
    - Descriptors instantiation and validation with XSD Models, or direct import of Descriptors received from the Producer;
    - MOT visualization (see TC1 and TC2 MOT figure 4-6 and 4-7), using an easy GUI;
  - SIP specification: SIP content definition, and sequencing constraints between SIPs;
- During the Transfer and Validation Phases (on the Archive side):
  - State of the delivered Transfer Objects in the MOT and detailed follow-up (see figure 4-8 and 4-9, where the number of delivered objects can be seen on the window, here 0);
  - Using the same graphical visualization;
  - Validations performed on the delivered SIPs (integrity, Data Objects occurrences, SIP contents, SIP sequencing constraints). They are applied on the XFDU manifest (not on the attached data).

It is supposed to be installed on the Archive side.

It is implemented in the JAVA programming language. It includes the Open Sources Xample (XML form) and JGraph (graphical design of the MOT).

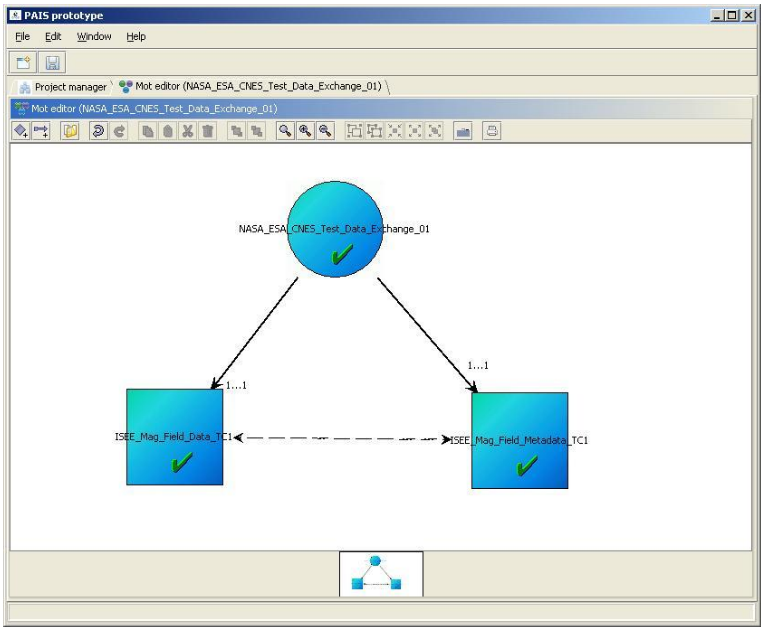

Figure 7-1: CNES Prototype – TC1 MOT visualization

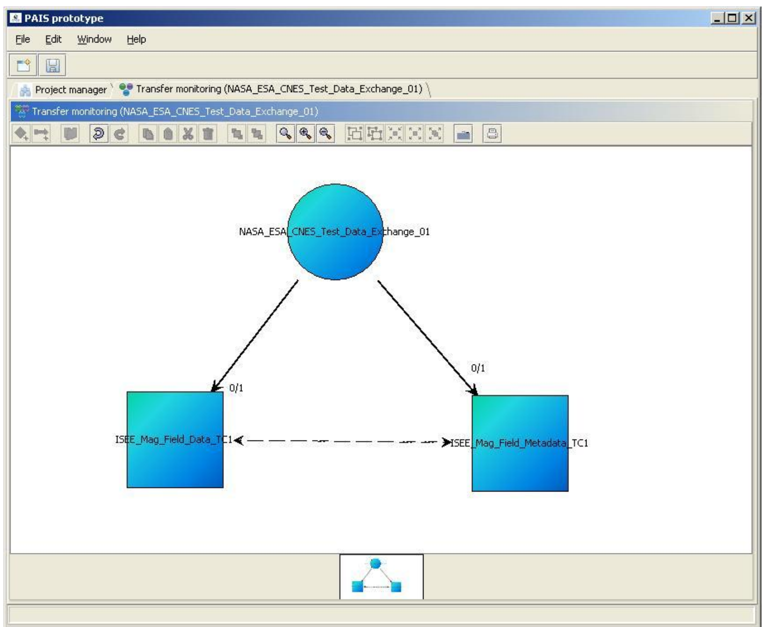

Figure 7-2: CNES Prototype – TC1 transfer follow up and validation

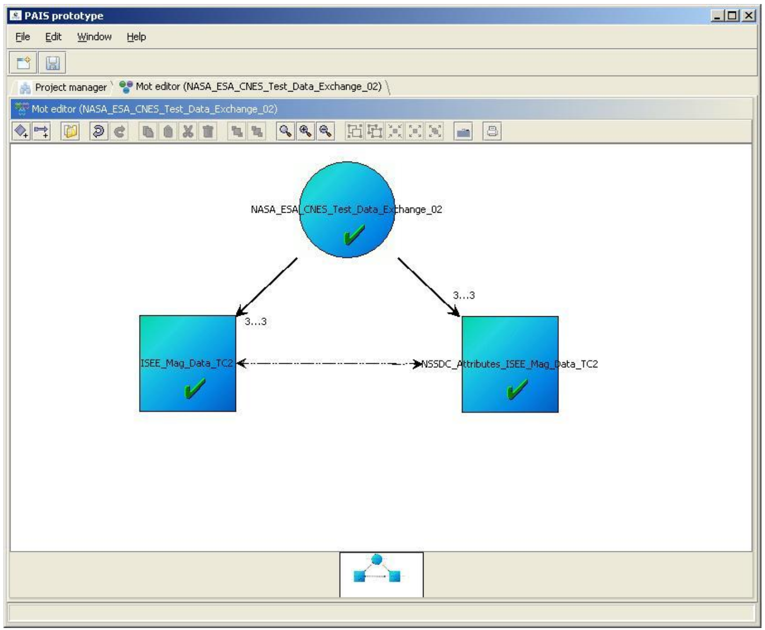

Figure 7-3: CNES Prototype – TC2 MOT visualization

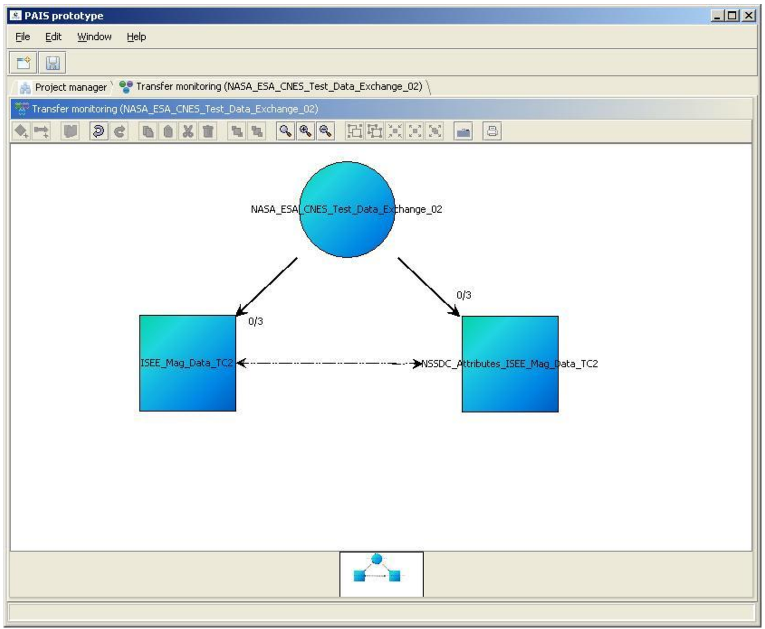

Figure 7-4: CNES Prototype – TC2 transfer and follow up

7.2 ESA SIP BUILDER

[SCOPE – this section is currently inherited from 651.3y0 record and needs to be integrated, e.g., the description of the PAIS descriptors is redundant in this document. The availability and licensing of this software should be addressed]

The ESA prototype, also called "ESA SIP Builder", is a Java command line software dedicated to the generation of SIPs according to the PAIS XFDU SIP Model specified in section 6 of the PAIS Draft Recommended Standard (see reference [1]). The following diagram identifies the main PAIS elements that are critical for the ESA SIP Builder software.

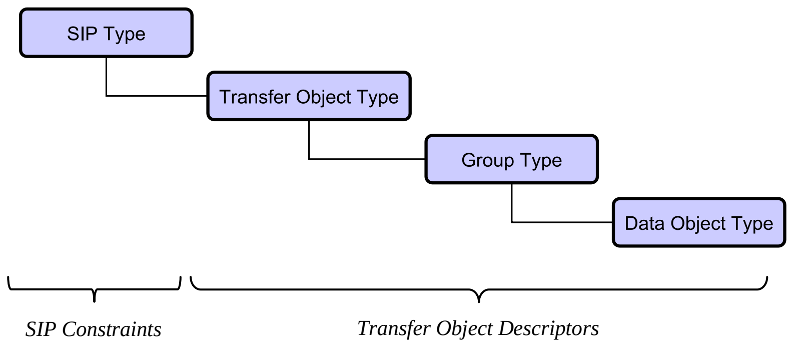

Figure 7-5: Identification of PAIS elements used by the ESA SIP Builder

On top of these standard PAIS definitions, the ESA SIP Builder introduces the concept of so called Collectors responsible for the supply of the actual data files and folders to be conveyed by the output SIPs.

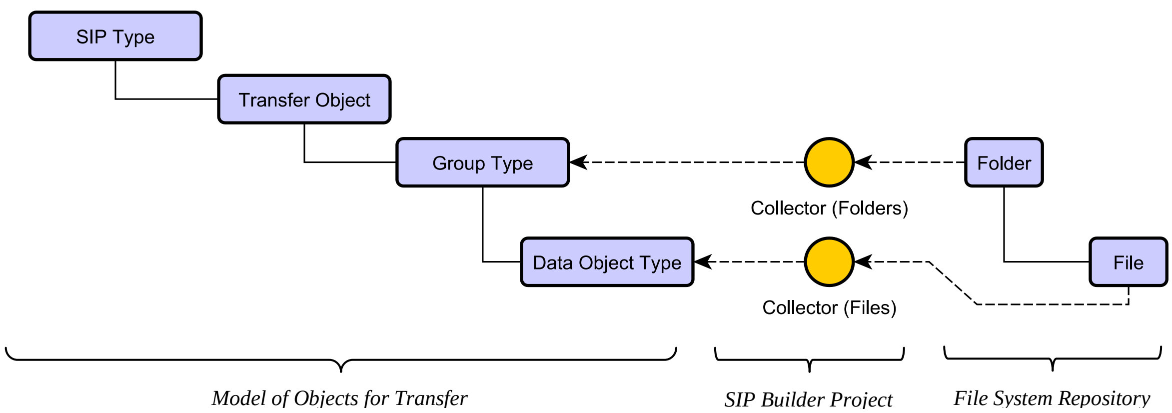

Figure 7-6: File and Folder Collectors

The ESA SIP Builder Collectors are attached to a target Group Type or a Data Object Type and hold one or more inclusion or exclusion patterns (file naming rules) that select files and folders in a given file system repository.

The ESA SIP Builder processes the input descriptors and collectors in an internal in-memory tree of objects before any conversion from-to the disk as XFDU packages. For disambiguation with PAIS and XFDU elements or objects, the internal in-memory objects are further prefixed with the term "Logical" e.g. Logical SIP composed of Logical Groups themselves composed of Logical Data Object.

The main steps of the ESA SIP Builder processing are the following:

– **Read and validate** the Collection Descriptors and Transfer Object Descriptors
– **Attach Collectors** to the descriptors Group Types and Data Objects Types
– **Select the SIP Types** to be produced from the SIP Constraints
– **Sort the SIP Types** from the SIP Constraints, if any
– **For each SIP Type**:
    For each authorized descriptor of the SIP Type:
        For each Group Type and sub-Group Type of the descriptor:
            Run the attached Collector, if any
            For each collected folder or once if none:
                For each Data Object Type:
                    Run the attached Collector considering the current Group Type instance name as context location
                    For each collected file:
                        Create a Logical Data Object of the current Data Object Type
            Create as many Logical Groups as necessary to hold the created Logical Data Objects or intermediate Logical Groups
        Create as many Logical Transfer Objects as necessary to hold the created Logical Groups
    Create as many Logical SIPs as necessary to hold the created **Logical Transfer Objects**
– **Validate the Logical SIPs** against descriptor constraints and limits e.g. occurrences, sizes, etc.
– **Convert the Logical SIPs** to definitive **XFDU** packages

A special attention has to be paid for the three "**as many as**" used in the above processing tree. The general strategy is a top-down analysis of the PAIS elements from the SIP Types to the Data Object Type leaves. However, the ESA SIP Builder Collectors may select more folders or files than allowed for a given Data Object Type or a Group Type cf. `maxOccurrence` usage in PAIS descriptors and constraints file. The ESA SIP Builder considers that extra instances of the parent Group Types, Transfer Object Types or even SIP Types need to be created. The use of `maxOccurrence`'s is even the only means for building multiple instances of Group Types or Data Object Types in the current implementation. It is not said that this is the best strategy but it seemed reasonable so far, limiting the complexity of configuration to simple patterns. Future improvements could be considered by implementing less "passive" Collectors capable of, for example, expressing grouping rules or more complex mapping.

The following practical example is probably more efficient than any further theory. It derives from a test case elaborated by NASA from data acquired by the International Sun-Earth Explorer (ISEE) first two spacecrafts.

The input data set of about 2 Gb has the following structure with two main directories isee1 and isee2 that share the same layout. The data are broken down in individual directories matching the year of acquisition. Each data file (`.asc-gz`) corresponds to a day of acquisition and has an associated metadata file (`.asc-gz_att`) in an `attrib` sub directory.

```
.
├── isee1
│   └── mag
│       └── 60_sec_ascii_gz
│           ├── 1977
│           │   ├── attrib
```

```
│              │    │    ├── isee1_mag_60s_0001_1977_295.asc-gz_att
│              │    │    ├── isee1_mag_60s_0002_1977_298.asc-gz_att
│              │    │    ├── isee1_mag_60s_0003_1977_300.asc-gz_att
│              │    │    ├── isee1_mag_60s_0004_1977_302.asc-gz_att
│              │    │    └── ...
│              │    ├── isee1_mag_60s_0001_1977_295.asc-gz
│              │    ├── isee1_mag_60s_0002_1977_298.asc-gz
│              │    ├── isee1_mag_60s_0003_1977_300.asc-gz
│              │    ├── isee1_mag_60s_0004_1977_302.asc-gz
│              │    └── ...
│              ├── 1978
│              │    └── ...
│              ├── ...
│              └── 1987
│                   └── ...
└── isee2
     └── mag
          └── 60_sec_ascii_gz
               ├── 1979
               │    ├── attrib
               │    │    ├── isee2_mag_60s_0001_1977_295.asc-gz_att
               │    │    └── ...
               │    ├── isee2_mag_60s_0001_1977_295.asc-gz
               │    └── ...
               ├── ...
               └── 1987
                    └── ...
```

In order to simulate an archiving project of this data set, NASA has designed a series of PAIS descriptors and SIP constraints files. The detail about these files is out of the scope of the present document but the main elements relevant for the present illustration are summarized in the following diagram.

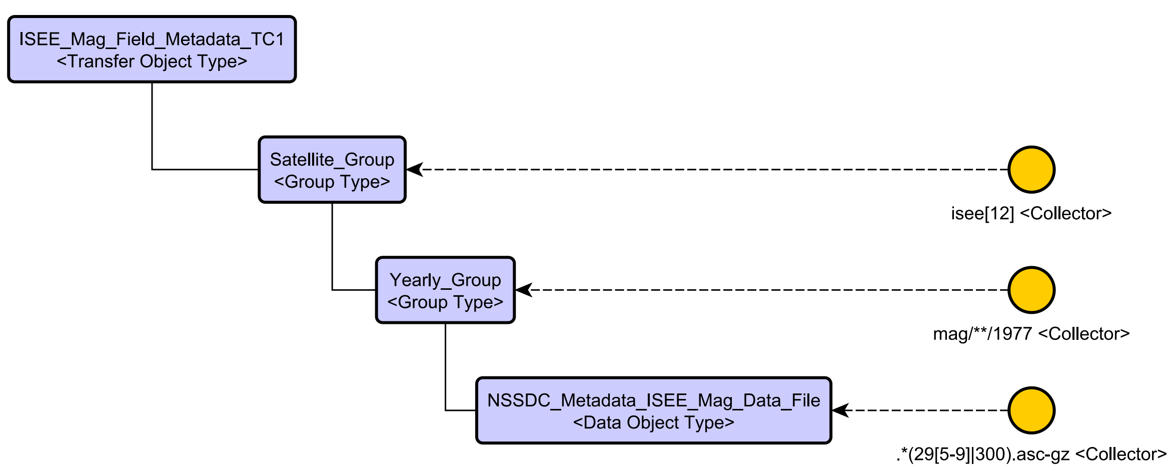

Figure 7-7: Example of Collectors supplying TC1 Groups and Data Objects

The descriptors define, among others, a Transfer Object Type identified by `ISEE_..._Metadata_TC1`. This latter is described has composed of a `Satellite_Group` Group Type, itself composed of a `Yearly_Group` Group Type that accepts a `NSSDC_Metadata_ISEE_Mag_Data_File` Data Object Type.

This tree of types matches quite well the layout of the data set described earlier where the `Satelite_Group` would correspond to the `isee1` and `isee2` directories, the `Yearly_Group` to the `1977`, `1978`, ... and `1987` directories and finally the `NSSDC_..._Mag_Data_File` would represent the metadata files in the `attrib` sub-directories. Three collectors are necessary to express this mapping to the ESA SIP Builder. They are represented in Yellow in the above diagram.

You may notice that different patterns could have produced the same results. For example, the pattern "`isee[12]`" could have been replaced by "`isee.`" (any trailing character in regular expression) because only `isee1` and `isee2` are present in the input data set. Similarly, the "`mag`" prefix for the `Yearly_Group` filters nothing as far as there is no other directory than the `mag` one for any spacecraft.

More specifically, you may notice that some collecting patterns includes some constraints that limit the file selection to a subset of what is actually available in the input data set. For example the collector attached to the `Yearly_Group` type selects only the files under the `1977` directory. Similarly, the "`.*(29[5-9]|300).asc-gz_att`" of the Data Object Type collector filters only those files acquired between the 295^{th} and the 300^{th} days of the year, included. These constraints are derived from the descriptors documentation e.g. a `transferObjectTypeDescription` element was containing "Selected ISEE 1,2 Magnetic_Field Metadata grouped by Spacecraft and then by Yearly Directories (1977 only) for days 295-300 inclusive." and thus required specific collector patterns for building SIPs with the right content.

The ESA SIP Builder interpretation of the input descriptors and collection patterns for this example can be summarized by the following diagram where a Transfer Object, on the left, will be generated with two sub-groups, one for each satellite, each containing a single year group containing three files. The tree on the right correspond to the input data set.

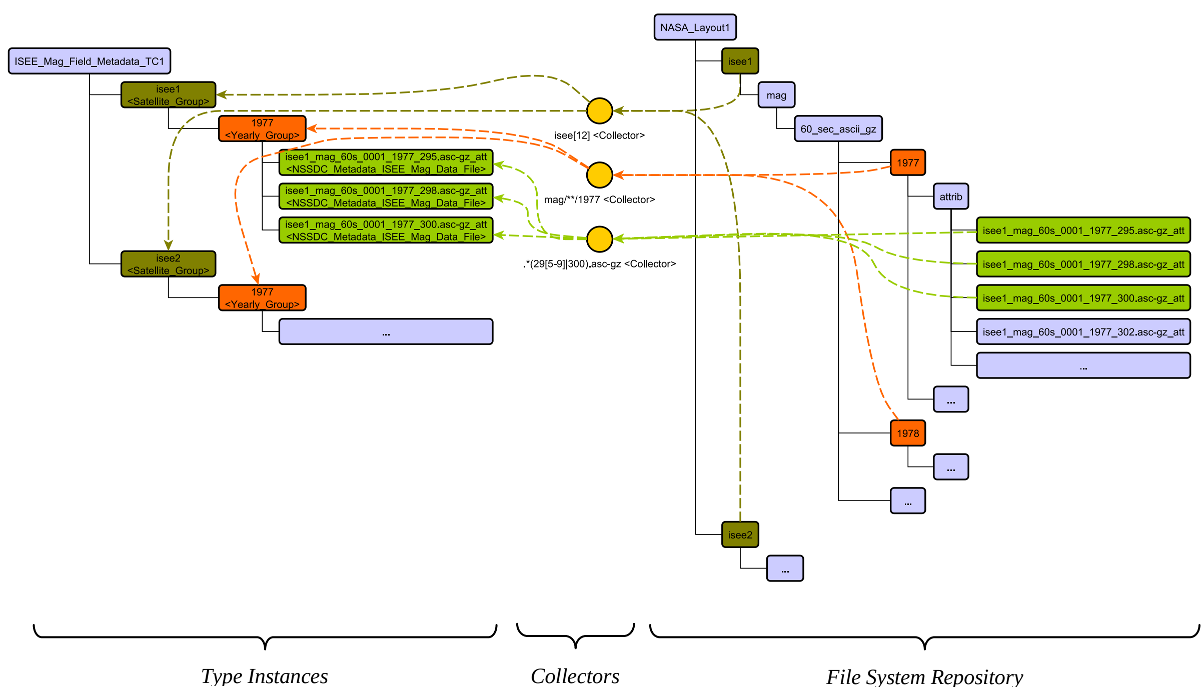

Figure 7-8: Collectors mapping from test data to SIP Groups and Data Objects

The configuration of the ESA SIP Builder is provided through a so called project file which is an XML document.

ANNEX A

PAIS – FULL MODEL

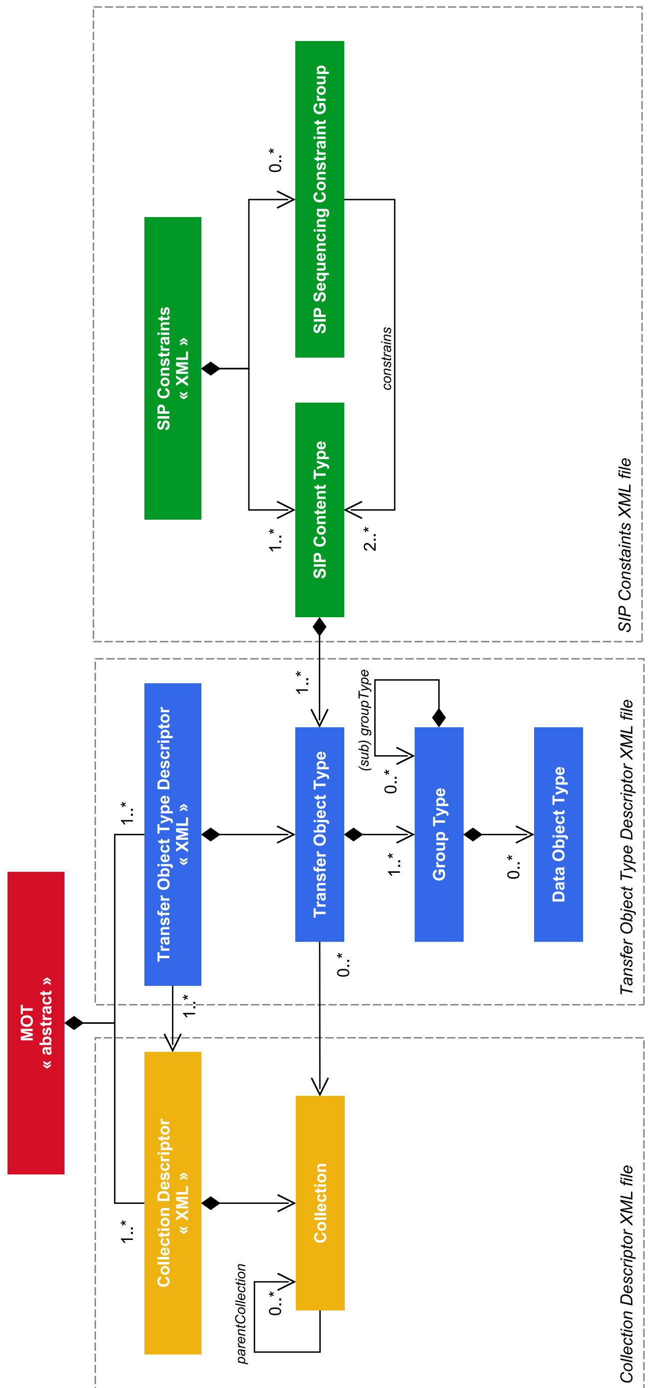

Figure A-1: PAIS – Full Model

ANNEX B

ISEE USE CASE DESCRIPTORS

[NOTE – Annex temporarily moved to `651x2g0-[6.1]-isee.docx` document]

ANNEX C

POLDER USE CASE DESCRIPTORS

[NOTE – Annex temporarily moved to `651x2g0-[6.1]-isee.docx` document]

INDEX

[SCOPE – at least all PAIS XML element shall be indexed]