Use cases guideline

PREAMBLE

With these use cases, the aim is to validate the concepts and the mechanisms proposed for the implementation of the Producer-Archive Interface. Although it is planned to implement the various concepts with XML schemas and XML documents conforming to these schemas, it is not necessary to use this formalism here. A simpler presentation in a textual form “attribute = value” etc …will be sufficient to describe these uses cases.

For more details, see the documents:

- PAIMAS,
- Producer-Archive Interface : Formal definition and transfer phase specification.

FORMAL DEFINITION PHASE

INTRODUCTION

The Digital Objects to be transferred from the Producer to the Archive, and the relationships between these objects will be described by DESCRIPTORS. All the DESCRIPTORS thus created constitute the Model of Objects to be Transferred (MOT).

The DESCRIPTORS are made up of a set of attributes and values assigned to these attributes in order to characterize the described Digital Objects. Among the attributes, there are attributes known as 'standards' and attributes known as ‘users’, in accordance with Data Entity Dictionnary Specification Language (DEDSL).

The DESCRIPTORS will be built from DESCRIPTOR MODELS specialized for the domain. These DESCRIPTOR MODELS are themselves built from a standardized DESCRIPTOR METAMODEL (cf. the following figure).
Each DESCRIPTOR refers to a model of SLIP. The SLIP makes it possible to specify in detail the information which will have, during the Transfer Phase, to accompany the object described by this DESCRIPTOR.

All the work stages will be described further and will be illustrated by examples.

STANDARD DESCRIPTOR METAMODEL

This metamodel takes the form defined by the following table:

STANDARD DESCRIPTOR METAMODEL

	Attribute_name	Attribute_definition	Attribute occurrence and condition	Attribute_value_type
	Descriptor_type	Defines the type of object described by this DESCRIPTOR. For a given type of DESCRIPTOR, this attribute has a constant value (see the examples).	1..1	identifier
Standard attributes	Name alias Descriptor_ID	Unique identifier for the described object.	1..1	identifier
	Title Alias short description	An object title to give the average user an idea of the object content	1..1	text
	Parent_collection	Parent Collection Identifier *The minimal Model required for the MOT is a hierarchical model made up of collections, sub-collections, … and objects. This model may be enriched by other relationships, as this will be shown by the examples.*	1..1	identifier

	Object_occurrence	This attribute makes it possible to know if the described object is unique (see for example the level 2 data Collection of the Waves experiment on WIND satellite) or if it is an object representative of a set of objects having the same characteristics (for example a data object of a collection of objects). *Examples:* *1..1 unique object* *1..n multiple object, maximum number undefined* *20..30 the number of objects is in the interval 20 to 30*	1..1	m..n m is an integer greater or equal to 1 n is an integer greater or equal to m 'n' means that the upper limit is unknown.
	Definition alias Content	Textual description of the object and its principal characteristics.	0..1	text
	Applicable_slip	The slip is a model specifying in detail the information to be delivered during the Transfer Phase with the object described by this DESCRIPTOR.	1..1	identifier

STEP 1 : DESCRIPTOR MODEL DEFINITION

DESCRIPTION

- **To be done**: define from the METAMODEL, the DECRIPTOR models adapted to the domain.
- **Method**:
    - Identify the categories of digital objects to be described and inserted in the MOT. In the selected example, the 4 following categories are defined: Space_physics_object_descriptor, Space_physics_collection_descriptor, Space_physics_complementary_collection_descriptor, Space_physics_complementary_object_descriptor. This choice is completely free. It is easily possible to adapt this classification to the domain and the domain terminology (this could be for example data_descriptor, collection_descriptor and metadata_descriptor …). Following this identification, a DOMAIN DESCRIPTOR METAMODEL has been built from the STANDARD DESCRIPTOR METAMODEL. The only change between these two metamodels concerns the DESCRIPTOR_TYPE attribute.

Example : SPACE_PHYSICS DESCRIPTOR METAMODEL

	Attribute_name	Attribute_definition	Attribute occurrence and condition	Attribute_value_type
	Descriptor_type	Defines the type of object described by this DESCRIPTOR.	1..1	Enumerated (Space_physics_object_descriptor, Space_physics_collection_descriptor, Space_physics_complementary_collection_descriptor ; Space_physics_complementary_object_descriptor)
Standard attributes				

    - Define, for each category of digital objects, which are the specific attributes (user attributes), complementary to define in order to have a sufficiently precise description of the objects. In our example, 4 DESCRIPTOR Models are defined, corresponding to the 4 types identified above.

- **Output**: the set of domain DESCRIPTORS in the same form as the following example.

EXAMPLE

This example is made up of 4 DESCRIPTOR Models adapted to the CDPP context (Centre de Données de la Physique des Plasmas – http://cdpp.cesr.fr)

DESCRIPTOR Model for the data objects:

	Attribute_name	**Attribute_definition**	**Attribute_value**	**Attribute occurrence and condition**	**Attribute_value_type**
	Descriptor_type	Defines the type of object described by this DESCRIPTOR.	Space_physics_object_descriptor	1..1	Identifier **Constant value**
Standard attributes	Descriptor_ID			1..1	identifier
	Title			1..1	text
	Parent_collection			1..1	identifier
	Object_occurrence			1..1	Form m..n
	Content			0..1	text
	Applicable_slip			1..1	identifier
	Size	Estimated size of data objects		1..1	text
	Format	File format		1..1	Enumerated (flat_binary, ascii, CDF, FITS, flat_ASCII, CEF, PNG)
	Catalog_metadata_description	Description of the content of the metadata catalog	Each data object will be accompanied by 'catalog' metadata including: - the data object identifier - The start date	1..1	Text **Constant value**

			- The stop date - The version number of the software having created this object.		
	Catalog_metadata_schema	Name of the XMLschema	Data_object.xsd	1..1	Identifier **constant value**

Comment: at this stage we have defined a specialized DESCRIPTOR Model for the description of space physics data objects.

The value of the attributes will generally be defined for each Descriptor instance (defined later). However, certain attributes have a value which will be the same for all the data objects of the domain. In this case, these values of attribute are assigned at the level of the Descriptor Model. It is here the case for the attributes 'catalog_metadata_description' and 'catalog_metadata_schema'. The attribute 'catalog_metadata_schema' contains a pointer on the file data_object.xsd which is an XML Schema that must be available, and that structures the metadata defined in the attribute 'catalog_metadata_description'. An example of the XML Schema data_object.xsd is given in annex.

DESCRIPTOR Model for data collections:

	Attribute_name	Attribute_definition	Attribute_value	Attribute occurrence and condition	Attribute_value_type
	Descriptor_type	Defines the type of object described by this DESCRIPTOR.	Space_physics_collection_descriptor	1..1	Identifier **Constant value**
Standard attributes	Descriptor_ID			1..1	identifier
	Title			1..1	text
	Parent_collection			1..1	identifier
	Object_occurrence			1..1	Form m..n
	Content			0..1	text
	Applicable_slip			1..1	identifier
	Applicable_metadata_standard	Name of the metadata standard that will be applied to all the collections.	CDPP_collection_metadata.xsd	1..1	Identifier **constant value**
	Mission	Mission name		0..1	identifier
	Experiment	Experiment name		0..1	identifier

DESCRIPTOR Model for Complementary Data Objects:

This Descriptor is used to describe objects of various type (document, syntactical descriptors, semantic descriptors …) that are usually linked to a data collection or objects of a data collection.

	Attribute_name	**Attribute_definition**	**Attribute_value**	**Attribute occurrence and condition**	**Attribute_value_type**
	Descriptor_type	Defines the type of object described by this DESCRIPTOR.	Space_physics_complementary_object_descriptor	1..1	Identifier **Constant value**
Standard attributes	Descriptor_ID			1..1	identifier
	Title			1..1	text
	Parent_collection			1..1	identifier
	Object_occurrence			1..1	Form m..n
	Content			0..1	text
	Applicable_slip			1..1	identifier
	Catalog_document_description	Each textual document will be accompanied by ‘catalog’ metadata whose content is described by this attribute.	- title - author - origin - version	0..1	Text **Constant value**
	Catalog_document_schema	Name of the XML Schema that structures the ‘catalog’ metadata associated to the textual document.	CDPP_document.xsd	0..1	Identifier **constant value**

	Data_description	Existence of a future syntactical description (in EAST language) and a DED.		0..2	Enumerated (EAST, DED)
	Related_descriptor_ID	Name of the Descriptor_ID completed by the object in question.		1..1	identifier
	Relation_DO_CDO	Description of the relationship with the Data Object		0..1	text

DESCRIPTOR Model for Complementary Data Objects collections:

This type of collection makes it possible to gather various objects which refer to the same collection of data objects or to the same data objects.

	Attribute_name	**Attribute_definition**	**Attribute_value**	**Attribute occurrence and condition**	**Attribute_value_type**
	Descriptor_type		Space_physics_complementary_collection_descriptor	1..1	Identifier **Constant value**
Partie standard	Descriptor_ID			1..1	identifier
	Title			1..1	text
	Parent_collection			1..1	identifier
	Object_occurrence			1..1	Form m..n
	Content			0..1	text
	Applicable_slip			1..1	identifier
	Related_descriptor_ID	Name of the Descriptor_ID with which the current object is in relation.		1..1	identifier
	Relation_DO_CDO	Textual description of the relationship identified by related_descriptor_id		0..1 mandatory if related_descriptor_id is specified	text

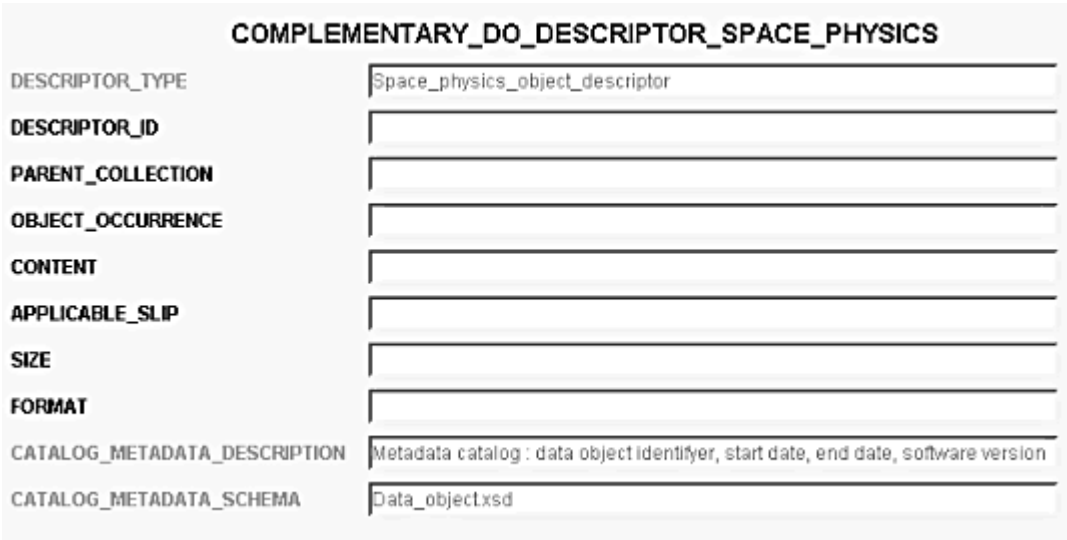
STEP 2 : DESCRIPTOR INSTANCIATION

DESCRIPTION

- **To be done**: instanciate the DESCRIPTORS that have been defined.
- **Method**:
    - For each Descriptor_type and Descriptor_ID, complete the attribute values. For the use cases, this will be made in a purely textual way.

    In the case of DESCRIPTOR Models valid for a domain, the archive will be able to place forms at the producer's disposal (for a given project), in order to facilitate the building of descriptor instances.

    Example of form for the data objects:



COMPLEMENTARY_DO_DESCRIPTOR_SPACE_PHYSICS	
DESCRIPTOR_TYPE	Space_physics_object_descriptor
DESCRIPTOR_ID	
PARENT_COLLECTION	
OBJECT_OCCURRENCE	
CONTENT	
APPLICABLE_SLIP	
SIZE	
FORMAT	
CATALOG_METADATA_DESCRIPTION	Metadata catalog : data object identifyer, start date, end date, software version
CATALOG_METADATA_SCHEMA	Data_object.xsd

    - Check the existence of piece of information that have to be available during the Formal Definition Phase (Catalog_metadata_description, …).
- **Output**: descriptor instances in a textual way.

EXAMPLE

In the example, issued from an example shown during the DAI meeting, is defined:

3 SPACE_PHYSICS_COLLECTION_DESCRIPTOR INSTANCES

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_collection_descriptor
Descriptor_ID	WI_WA_CO
Title	Wind waves data collections
Parent_collection	root
Object_occurrence	1..1
Content	Collection of the waves data collections
Applicable_slip	Empty_slip (nothing to be transferred at this level)
Applicable_metadata_standard	CDPP_collection_metadata.xsd
Mission	WIND
Experiment	Waves

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_collection_descriptor
Descriptor_ID	WA_TNR_L2_CO
Title	Waves data set level 2
Parent_collection	WI_WA_CO
Object_occurrence	1..1
Content	Collection of the calibrated high resolution data from the Thermal Noise Receiver of the Wave experiment
Applicable_slip	CDPP_metadata_slip
Applicable_metadata_standard	CDPP_collection_metadata.xsd
Mission	WIND
Experiment	Waves

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_collection_descriptor
Descriptor_ID	WA_TNR_IMAGES_CO

Title	Waves TNR dynamic spectra browse set
Parent_collection	WI_WA_CO
Object_occurrence	1..1
Content	Collection of dynamic spectra images from the Waves calibrated data
Applicable_slip	CDPP_metadata_slip
Applicable_metadata_standard	CDPP_collection_metadata.xsd
Mission	WIND
Experiment	Waves

2 SPACE_PHYSICS_OBJECT_DESCRIPTOR INSTANCES

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_object_descriptor
Descriptor_ID	WA_TNR L2_DO
Title	Waves data object level 2
Parent_collection	WA_TNR_L2_CO
Object_occurrence	700..n
Content	Each file contains: the waves thermal noise receiver calibrated spectra (mV2/Hz), the calibrated tnr cag … A detailed data description will be provided in EAST and DEDSL languages
Applicable_slip	CDPP_data_object_slip
Size	10 Mb < file size < 30 Mb
Format	Flat_binary
Catalog_metadata_description	Each data object will be accompanied by ‘catalog’ metadata including : - the data object identifier, - The start date, - The stop date, - The version number of the software having created this object.
Catalog_metadata_schema	Data_object.xsd

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_object_descriptor
Descriptor_ID	WA_TNR_IMAGES_DO

Title	Wind waves dynamic spectra image
Parent_collection	WA_TNR_IMAGES_CO
Object_occurrence	700..n
Content	Each 24h image contains the dynamic spectra. The plasma frequency, computed from the neural network is added on the plot.
Applicable_slip	CDPP_transferred_object_slip
Size	50 Kb < file size < 500 Kb
Format	PNG
Catalog_metadata_description	Each data object will be accompanied by 'catalog' metadata including : - the data object identifier, - The start date, - The stop date, - The version number of the software having created this object.
Catalog_metadata_schema	Data_object.xsd

1 SPACE_PHYSICS_COMPLEMENTARY_COLLECTION_DESCRIPTOR INSTANCE

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_complementary_collection_descriptor
Descriptor_ID	WA_TNR_CCO
Title	Waves TNR complementary documents and descriptions
Parent_collection	ROOT
Object_occurrence	1..1
Content	This collection of complementary data objects contains documentary objects, syntactical and semantic descriptions associated with the 'Waves data set level 2' collection and with the objects included in this collection.
Applicable_slip	Empty_slip (nothing to be transferred at this level)
Related_descriptor_ID	WA_TNR_L2_CO
Relation_DO_CDO	Relation of description

2 SPACE_PHYSICS_COMPLEMENTARY_OBJECTS_DESCRIPTOR INSTANCES

Attribute_name	**Attribute_value**

Descriptor_type	Space_physics_complementary_object_descriptor
Descriptor_ID	WA_TNR_CDO_1
Title	TNR documentation
Parent_collection	WA_TNR_CCO
Object_occurrence	2..2
Content	Textual documents describing the Waves experiment Document 1 : measurement principle Document 2 : instrument description
Applicable_slip	CDPP_metadata_slip
Catalog_document_description	- title - author - origin - version
Catalog_document_schema	CDPP_document.xsd

Attribute_name	**Attribute_value**
Descriptor_type	Space_physics_complementary_object_descriptor
Descriptor_ID	WA_TNR_CDO_2
Title	Syntactical and semantic TNR data description
Parent_collection	WA_TNR_CCO
Object_occurrence	1..1
Content	Syntactical description in EAST standard language Semantic description in DEDSL standard language for the Waves TNR data files
Applicable_slip	CDPP_metadata_slip
Data_description	EAST
Data_description	DED
Related_descriptor_ID	WA_TNR_L2_DO
Relation_DO_CDO	Relation of description

STEP 3 : SLIP MODEL DEFINITION

The SLIP Models specify various information that must be actually transferred. In certain cases, the data objects could actually be (and thus physically) transferred within the SIP. In other cases, the producer won't transfer the objects to the archive but will transfer the information needed in order to recover these objects (for example a request to the producer data base). In other cases still, the producer and the archive may share the same storage service. The producer will then limit himself to transfer to the archive the address of the physical objects within the service. It is the case for the CDPP.

Slip Model for data objects (case where the data already exist in the storage service):

Attribute_name	Attribute_definition	Attribute occurrence and condition	Attribute_value_type
Slip_ID	Slip type identifier	1..1	Identifier **Constant value** = CDPP_data_object_slip
Object_ID	Object Identifier	1..1	identifer
Descriptor_ID	Descriptor identifier corresponding to this object	1..1	identifier
Stored_object_access	Composit entity giving access to the various bit sequences in the storage service (see below).	1..n	composit
Extraction_software	Name of the software used to reconstruct the digital object from the bit sequences.	0..1	identifier
Last_object	Flag of last object delivered	1..1	Enumerated (Yes, No)
Catalog_file_name	Name of the XML catalog file associated to Object_ID	1..1	identifier

The 'stored_object_access' attribute is a composit entity used to describe how to access to the different bit sequences constituting the object_ID.

Stored Object_access	Archive	Name of the storage directory in the storage area.	1..1	Enumerated (STAF_PLASMA, STAF_TRANSFER)
	Archive_path	Access path to the data in the storage area.	1..1	identifier
	Archive_object_name	Optional if identical to the data object identifier	1..1	identifier

SLIP Model for data objects (the data are transferred with the Slip):

Attribute_name	Attribute_definition	Attribute occurrence and condition	Attribute_value_type
Slip_ID	Slip type identifier	1..1	Identifier **Constant value** = CDPP_transferred_object_slip
Object_ID	Object Identifier (Data Object, Complementary Data Object, Collections)	1..1	identifer
Descriptor_ID	Descriptor identifier corresponding to this object	1..1	identifier
Object_access	Composit entity giving access to the various bit sequences in the storage service (see below).	1..n	identifier
Extraction_software	Name of the software used to reconstruct the digital object from the bit sequences.	0..1	identifer
Catalog_file_name	Name of the XML catalog file associated to Object_ID	1..1	identifier
Last_object	Flag of last object delivered	1..1	Enumerated (Yes, No)

The ‘object_access’ attribute is a composit entity used to describe how to access to the various bit sequences constituting the SIP when these sequences are physically transferred in the SIP. In this case, a means of checking the integrity of the transferred sequences has been planned.

Object_access	File_name	Bit sequence identifier	1..1	identifier
	path	Access path in the SIP	1..1	identifier
	Checksum	Checksum	1..1	identifer
	Checksum_path	Access path to the checksum	1..1	identifier

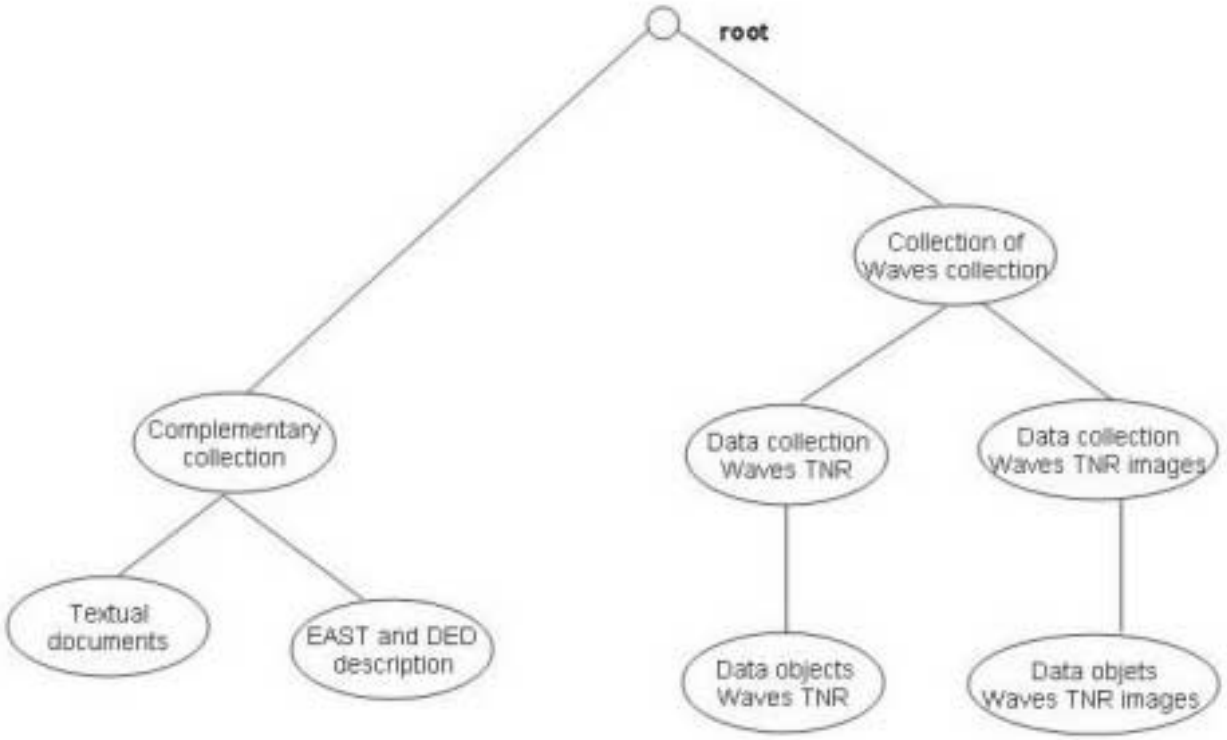
SLIP Model for the transfer of metadata and complementary objects :

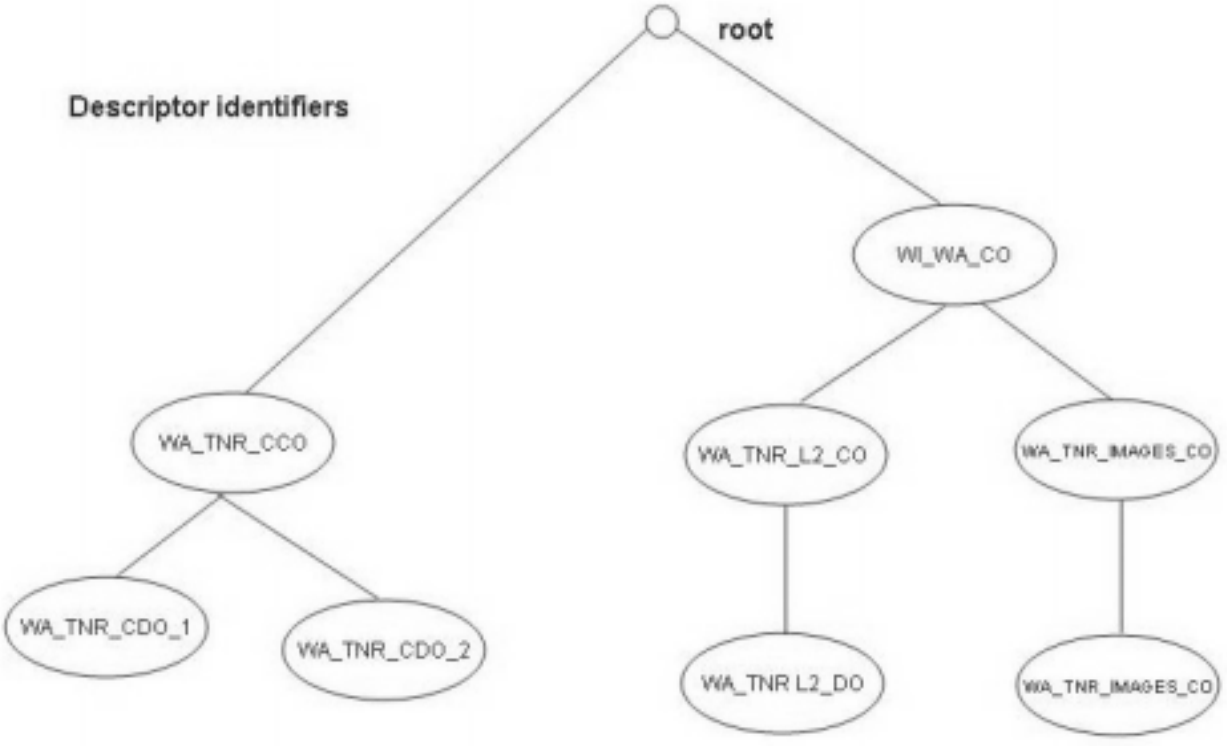
Attribute_name	Attribute_definition	Attribute occurrence and condition	Attribute_value_type

Slip_ID	Slip type identifier	1..1	Identifier **Constant value** = CDPP_metadata_slip
Object_ID	Object Identifier (Data Object, Complementary Data Object, Collections)	1..1	identifer
Descriptor_ID	Identifiant du descriptor correspondant à cet objet	1..1	identifier
Object_access	Composit entity giving access to the various bit sequences in the storage service.	0..n	identifier
Metadata_file	Name of the Object_ID metadata file	0..1	identifier

STEP 4 : MOT DESIGN

The objective here is to give a visual representation of the Model of Objects to be Transferred.





STEP 5 : SIP MODEL DEFINITION

DESCRIPTION

- **To be done**: specify the grouping of the objects to be delivered within the SIP .
- **Method**:
    - From the generic SIP Model below, it will be possible to define one or several categories of SIP.

Attribute_name	Attribute_definition	Attribute_value	Attribute occurrence and condition	Attribute_value_type
SIP_TYPE_ID			1..1	identifier **Constant**
Descriptor_ID			1..n	identifier
TIME_CONSTRAINT_GROUP			0..1	Integer value
SERIAL_NUMBER_IN_CONSTRAINT			0..1	Integer value

- **Output**: the description of the different categories of SIP.

EXAMPLE

The instance SIP1 of SIP Model deals with the syntactical and semantic description of the level 2 Wind Waves data. It shows that this description has to be transferred before any data transferred in SIP of the type SIP2.

Attribute_name	Attribute_value	Attribute occurrence and condition
SIP_TYPE_ID	**SIP1**	1..1
Descriptor_ID	WA_TNR_CDO_2	1..1
TIME_CONSTRAINT_GROUP	A	1..1
SERIAL_NUMBER_IN_CONSTRAINT	1	1..1

The instance SIP2 of SIP Model deals with the transfer of the data objects of the two data collections identified in the MOT. It shows that SIP2 can indifferently contains data objects of the

WA_TNR_IMAGES_CO collection (images), and data objects of the WA_TNR_L2_CO collection, but limits, for size reasons, the number of objects possibly transferred in each SIP.

Attribute_name	Attribute_value	Attribute occurrence and condition
SIP_TYPE_ID	**SIP2**	1..1
Descriptor_ID	WA_TNR_IMAGES_DO	0..50
Descriptor_ID	WA_TNR_L2_DO	0..10
TIME_CONSTRAINT_GROUP	A	1..1
SERIAL_NUMBER_IN_CONSTRAINT	2	1..1

The SIP3 instance of SIP Model shows that the descriptive metadata of WA_TNR_L2_CO data collection and WA_TNR_IMAGES_CO image collection, and the WA_TNR_CDO_1 documents can be transferred together or separately without particular constraint.

Attribute_name	Attribute_value	Attribute occurrence and condition
SIP_TYPE_ID	**SIP3**	1..1
Descriptor_ID	WA_TNR_CDO_1	0..2
Descriptor_ID	WA_TNR_IMAGES_CO	0..1
Descriptor_ID	WA_TNR_L2_CO	0..1

TRANSFER PHASE

INTRODUCTION

SIP1: DED and EAST files: to deliver before any delivery of SIP2,

SIP2: Wind_waves_tnr_l2 data.

EXAMPLES OF SLIP INSTANCES

1. CDPP_METADATA_SLIP INSTANCE

Attribute_name		**Attribute_name**
Slip_ID		CDPP_matadata_slip
Object_ID		WA_TNR_L2_DESCRIPTIONS
Descriptor_ID		WA_TNR_CDO_2
Object_access	File_name	WA_TNR_L2.EAST EAST_PATH
	path	
	checksum	EAST_CHECKSUM_FILE EAST_PATH
	Checksum_path	
Object_access	File_name	WA_TNR_L2.DED DED_PATH
	path	
	checksum	DED_CHECKSUM_FILE DED_PATH
	Checksum_path	

2. CDPP_DATA_OBJECT_SLIP INSTANCE

Attribute_name		**Attribute_name**
Slip_ID		CDPP_data_object_slip
Object_ID		WI_WA_TNR_L2_20011217_V01
Descriptor_ID		WA_TNR L2_DO
Stored_object _access	Archive	STAF_PLASMA
	Archive_path	PLASMA/WIND/WAVES
	Archive_object_name	WI_WA_TNR_L2_20011217_V01.dat

Last_object	FALSE
Catalog_file	Wind_waves_tnr_l2_do_20011217.xml

An example of XML Schema data_object.xsd is given in annex. Below is an example of XML catalog file derived from this schema:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Sample XML file generated by XML Spy v4.4 U (http://www.xmlspy.com)-->
<CATALOG_DESCRIPTION xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="D:\Cnes_wind_waves\Shema Data_Object.xsd">
  <DATA_OBJECT_DESCRIPTION ENTITY_TYPE="CATALOG_DESCRIPTION">
     <OBJECT_IDENTIFIER>Data 1</OBJECT_IDENTIFIER>
          <TIME_PERIOD>
            <START_DATE>2001-12-17T00:00:00-05:00</START_DATE>
            <STOP_DATE>2001-12-17T23:59:59-05:00</STOP_DATE>
          </TIME_PERIOD>
          <SOFT_VERSION>1</SOFT_VERSION>
   </DATA_OBJECT_DESCRIPTION>
 </CATALOG_DESCRIPTION>
```

ANNEX: DATA_OBJECT.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""" -->
    <!-- Root definition                                          -->
    <!-- """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""" -->
    <xs:element name="CATALOG_DESCRIPTION">
        <xs:annotation>
            <xs:documentation>Object Catalog Description </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="DATA_OBJECT_DESCRIPTION" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <!-- """"""""""""""""""""""""""""""""""""" -->
    <!-- Data_Object  definition         -->
    <!-- """"""""""""""""""""""""""""""""""""" -->
    <xs:element name="DATA_OBJECT_DESCRIPTION">
        <xs:annotation>
            <xs:documentation> Data Object description </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="OBJECT_IDENTIFIER"/>
                <xs:element ref="TIME_PERIOD" minOccurs="0"/>
                <xs:element ref="SOFT_VERSION" minOccurs="0"/>
            </xs:sequence>
            <xs:attribute name="ENTITY_TYPE" use="required" fixed="CATALOG_DESCRIPTION"/>
        </xs:complexType>
    </xs:element>
    <!-- """""""""""""""""""""""""""""""""""""""""""""""""""""" -->
    <!-- Attribute definition                         -->
    <!-- """"""""""""""""""""""""""""""""""""""""""""""""""""""-->
    <xs:element name="OBJECT_IDENTIFIER" type="TypeString_1_64">
        <xs:annotation>
            <xs:documentation> Object Identifier </xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="TIME_PERIOD" type="TypeCTimePeriod">
        <xs:annotation>
            <xs:documentation>Temporal Periode ( Start date  - Stop date)</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="SOFT_VERSION" type="xs:integer">
        <xs:annotation>
            <xs:documentation>Software Version Number</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!-- """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""" -->
    <!-- Complex Type definition  TIME_PERIOD and String 64 -->
    <!-- """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""" -->
    <xs:simpleType name="TypeString_1_64">
        <xs:annotation>
            <xs:documentation>String from 1 to  64 char.</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
            <xs:maxLength value="64"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="TypeCTimePeriod">
        <xs:annotation>
            <xs:documentation>Temporal Periode ( Start date  - Stop date), format YYYY-MM-DDThh:mm:ss</xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element name="START_DATE" type="xs:dateTime">
                <xs:annotation>
                    <xs:documentation>Start d
                Date </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="STOP_DATE" type="xs:dateTime">
```

```
                <xs:annotation>
                    <xs:documentation>Stop Date</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:all>
    </xs:complexType>
</xs:schema>
```