

EXPRESS Data as HDF5 Mapping Specification Version 0.5

This document defines the mapping of EXPRESS-driven data into HDF5.

Table of contents

| | |
|--|----|
| 1 Scope..... | 3 |
| 2 Normative References..... | 3 |
| 3 HDF5 in a Nutshell..... | 4 |
| 4 EXPRESS-HDF5 Approach..... | 6 |
| 5 EXPRESS Data Represented as HDF5..... | 7 |
| 5.1 HDF5 Link Names..... | 7 |
| 5.2 The Population of EXPRESS Schemas as HDF5..... | 8 |
| 5.3 EXPRESS Schema Information as HDF5..... | 10 |
| 5.4 EXPRESS Defined Type Information as HDF5..... | 12 |
| 5.5 EXPRESS Enumeration Types as HDF5..... | 14 |
| 5.6 EXPRESS Select Types as HDF5..... | 16 |
| 5.7 EXPRESS Entity Type Information as HDF5..... | 18 |
| 5.8 Simple EXPRESS Entity Instances as HDF5..... | 19 |
| 5.9 EXPRESS Entity Instances with related Aggregate Instances..... | 21 |
| 5.10 EXPRESS Entity Instance Identifiers..... | 22 |
| 5.11 EXPRESS Datatype Values as HDF5..... | 23 |
| 5.12 EXPRESS Array Datatype Values as HDF5..... | 24 |
| 5.13 EXPRESS Aggregate Datatype Values as HDF5..... | 25 |
| 6 Examples..... | 26 |

EXPRESS Data as HDF5 Mapping Specification Version 0.5

| | |
|---|----|
| 7 EXPRESS-HDF5 Mapping Issues..... | 27 |
| 7.1 Issues Against V 0.4..... | 27 |
| 7.2 Issues Against V 0.2..... | 28 |
| 7.3 Issues Against V 0.1..... | 28 |
| 7.4 Initial Issues from V 0.1 development..... | 28 |
| 8 Relationship to ISO 10303-21 Part 21 Clear text encoding..... | 28 |
| 9 Bibliography..... | 29 |

Warning:

Version 0.5 extends the scope of the mapping to include all EXPRESS-driven data. In some cases, it also now included two proposed mappings for some constructs both of which are being considered and for which comparative testing is important. The mapping may change significantly as the EXPRESS community gains HDF5 experience.

1. Scope

The following are within the scope of this specification:

- multiple populations of an EXPRESS schema in an HDF5 file;
- populations of the same EXPRESS schema but with different encodings into HDF5;
- populations of multiple EXPRESS schemas in the same HDF5 file;
- the mapping of EXPRESS entity instances into HDF5;
- the mapping of Entity and Attribute into HDF5;
- the mapping of Enum into HDF5;
- the mapping of all EXPRESS simple datatypes into HDF5;
- the mapping of defined types that specialize other defined types or EXPRESS simple types;
- the mapping of n-dimensional arrays of a single type into HDF5;
- the mapping of simple single-dimensional lists, sets and bags of a single type are included into HDF5;
- the mapping of multidimensional lists, sets and bags into HDF5;
- the mapping of EXPRESS select types into HDF5;
- the mapping of attribute redeclarations into HDF5;
- the mapping of arrays, lists, sets and bags not of a single type into HDF5;
- the mapping of Defined types of arrays into HDF5;

The following are outside the scope of this specification:

- representing EXPRESS rules, functions and/or procedures using HDF5;
- abstract entity;
- representing complex numbers using HDF5;
- the mapping of inverse attributes into HDF5;
- representing derived attributes using HDF5.

2. Normative References

The following normative references are applicable for this specification.

ISO 10303-11:2004, Industrial automation systems and integration - Product data representation and exchange - Part 11: Description methods: The EXPRESS language reference manual.

HDF5 Release 1.6.5 , November 2005 [cited 2006-10-01]. Available from World Wide Web: http://www.hdfgroup.com/HDF5/doc/RM_H5Front.html.

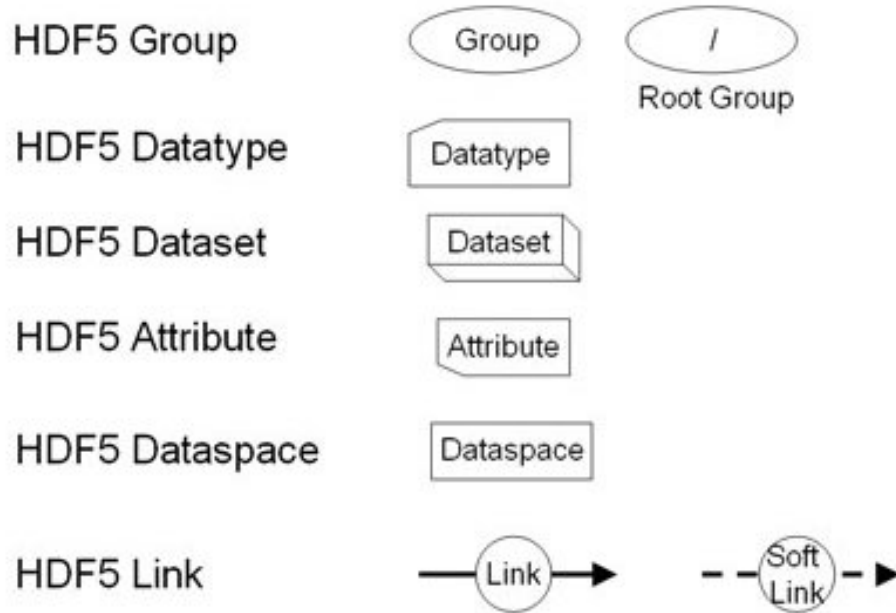
HDF5 User's Guide, HDF5 Release 1.6.5, November 2005 [cited 2006-10-01]. Available from World Wide Web: <http://hdfgroup.com/HDF5/doc/UG/>.

3. HDF5 in a Nutshell

For those less familiar with HDF5, it's perhaps simplest to start ignoring some of the complexity and focusing on a simple use of its basic architecture as follows:

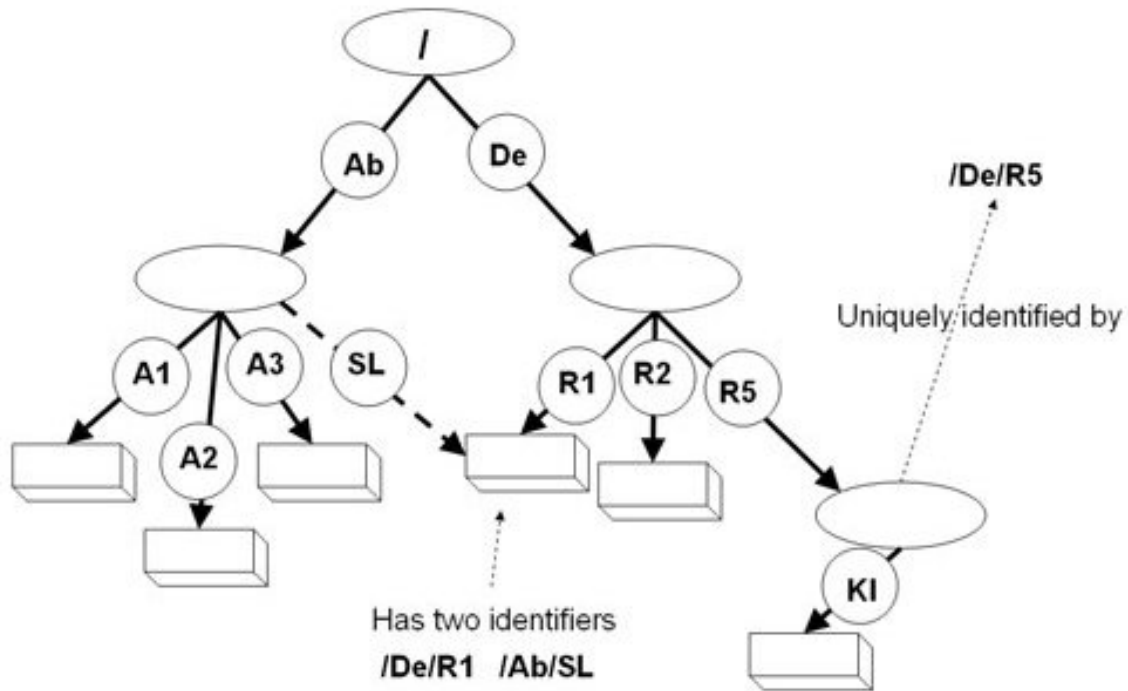
- An HDF5 file can contain data and objects of many types (logical groupings of data, datatype definitions, bitmap images, large arrays of data, etc).
- An HDF5 Group is a logical structure within an HDF5. HDF5 Groups contain other HDF5 Groups and HDF5 Datasets. An HDF5 Group is similar in concept to a directory or folder in a computer file system.
- The topmost HDF5 Group is called the Root Group and there is one Root Group per HDF5 file.
- An HDF5 Dataset contains the data in the file. An HDF5 Dataset contains the definition of HDF5 Datatypes, the definition of the dimensions of the array it contains called an HDF5 Dataspace and the data array itself.
- An HDF5 Datatype can be a simple type (e.g. integer) or a compound type similar to a C struct and can be named.
- An HDF5 Dataspace is the definition of the dimensions of the array and also contains information about how it is stored (e.g. is it compressed or not).
- HDF5 Groups and HDF5 Datasets can have HDF5 Attributes attached to specify some characteristic of the Group or Dataset.

In this specification, an informal diagram notation is used to represent the HDF5 concepts. The following figure shows that notation.



Informal HDF5 Diagram Notation

The above statements are incomplete but give the reader some hint of how HDF5 works. The following diagram shows these concepts and a convention used for diagrams in this specification.



HDF5 Concepts Diagram

A key resource for the understanding of the HDF5 technology and how it relates to this specification is the HDF5 Users Guide: Chapter 1 HDF5 Data Model. Implementors and reviewers are encouraged to read that document before continuing with this specification.

Another key resource is the HDF5 Tutorial [3].

4. EXPRESS-HDF5 Approach

A set of basic propositions underly the approach for representing EXPRESS-driven data using HDF5 in this specification. Those basic propositions follow.

- HDF5 software manages arrays well both in memory and on disk to satisfy the EXPRESS-driven data requirements.
- Maximizing the use of HDF 5 structures to enable the use of its optimizations is needed to satisfy performance and file size requirements.
- Only the datatypes used for the writing of the data on disk are specified, nothing is stated about the representation of that data in memory.
- It is preferable to use the HDF5 pre-defined data types where possible.
- It is preferable to allow flexibility of representation which puts somewhat more burden

on developers of software that reads HDF5 files.

- Cross-platform interoperability (e.g. write in Unix C and read in Windows Fortran) needs to be supported.

The general approach for representing EXPRESS-driven data using HDF5 is to treat instances of EXPRESS entity types and EXPRESS array instances in a similar manner. The set of instances of an EXPRESS entity type along with any non-array attribute values are treated as a dataset based on an HDF5 compound data type. EXPRESS Array-valued attributes are represented using an HDF5 reference to an HDF5 dataset for each array instance.

5. EXPRESS Data Represented as HDF5

This section describes the representation for EXPRESS-driven data represented using HDF5. This representation is specified by relating EXPRESS data concepts to HDF5 data concepts, not by specifying any particular application programming interface (API). HDF5 implementations are available in multiple programming languages. This specification does not specify the use of any particular HDF5 API.

A summary presentation of the mapping is also available : EXPRESS/HDF5 Mapping Summary Presentation.

5.1. HDF5 Link Names

HDF5 Groups, Datasets and Named Data Types used in this mapping are identified by an HDF5 Link. For HDF5 constructs mapped directly from EXPRESS constructs, the EXPRESS identifiers are used as part of the construction of the names of the HDF5 Links. The case of the EXPRESS identifiers is preserved. The forward slash ("/") character is used to separate HDF5 names that are part of the link.

Note:

The dot (".") character familiar to EXPRESS experts is reserved by HDF5 and so not used for this purpose.

From the HDF5 spec: It is important to note that, just like the UNIX file system, HDF5 objects do not have names, the names are associated with links. An object has an object identifier that is unique within the file, but a single object may have many names because there may be many links to the same object. An object can be renamed, or moved to another group, by adding and deleting links. In this case, the object itself never moves. For that matter, membership in a group has no implication for the physical location of the stored object.

This specification places no restrictions on additional links (i.e. names) being defined in the

file for any EXPRESS-driven HDF5 object.

If data is shared between EXPRESS-driven representations this specification does not preclude the same HDF5 object having links mapped from more than one EXPRESS schema.

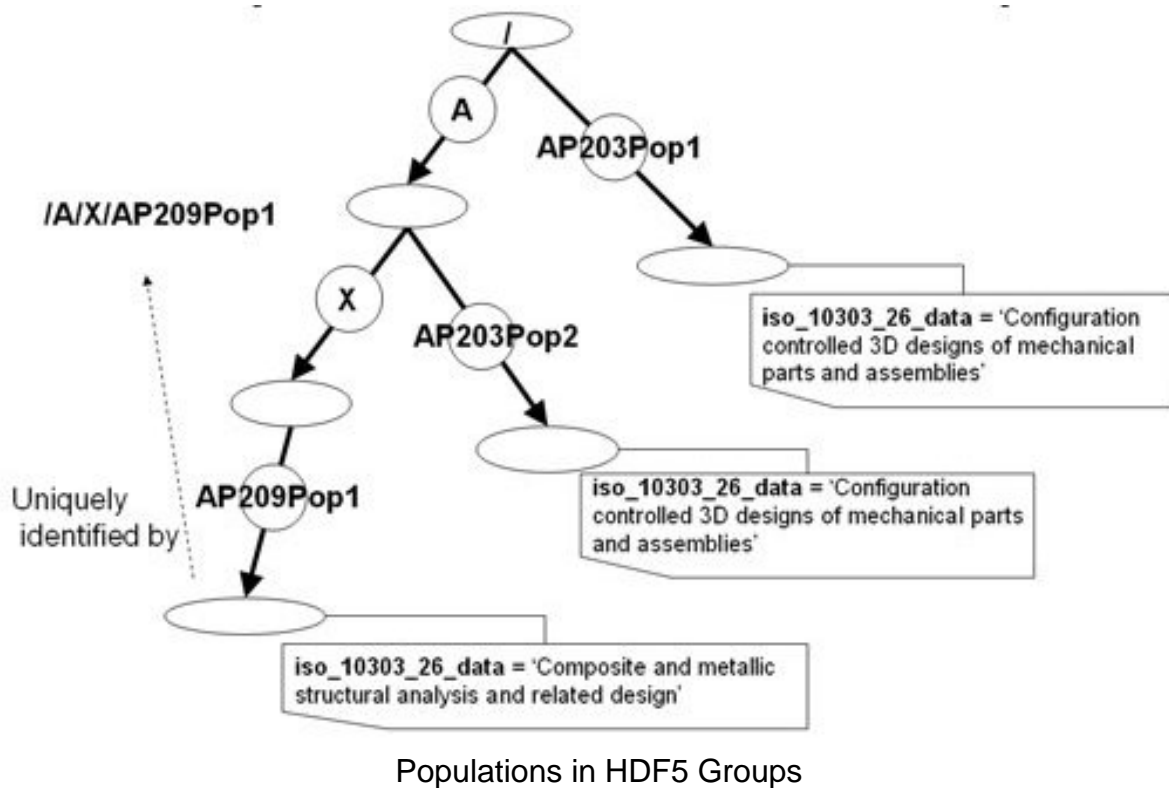
EXAMPLE: The following EXPRESS encoded in an HDF5 Group named "pets_encoding"

```
SCHEMA pets;  
  ENTITY Dogs;  
  END_ENTITY;  
END_SCHEMA;
```

could result in part of the HDF5 Link name for the related HDF5 Dataset being "pets_encoding/Dogs".

5.2. The Population of EXPRESS Schemas as HDF5

In this specification, EXPRESS schemas are considered to be defined for the purpose of constraining the validity of data populations. Multiple populations based on the same EXPRESS schema are possible and may be included in the same HDF5 file. As numerous ways to represent EXPRESS-driven data values based on the same schema in HDF5 are possible, no single representation is required for any EXPRESS schema. The same HDF5 file may contain populations of an EXPRESS schema that use different encodings. Information about the HDF5 constructs used to encoding the EXPRESS-driven data is stored in the HDF5 file along with the data itself. The following figure shows three HDF5 Groups containing data based on this specification.



Note:

The HDF5 File Interface and HDF5 Group Interface are used to create the HDF5 File and HDF5 Group.

Each population of an EXPRESS schema is represented as an HDF5 Group. That HDF5 Group shall have an HDF5 Attribute associated with it with the following name and description:

- `iso_10303_26_data` : which has a data value of the `<schema_id>`

The HDF5 Attribute named "iso_10303_26_data" is the indicator to a software application that the HDF5 Group contains data encoded based on this specification.

This specification also defines several optional HDF5 Attributes for the purpose of annotating HDF Objects that are based on this specification. They have the following names and descriptions:

- `iso_10303_26_description` : which has a data value of the `<user_defined_description>`
- `iso_10303_26_timestamp` : which has a data value of the corresponding to the

extended format for the complete calendar date as specified in 4.2.1.1 of ISO 8601 concatenated to the extended format for the time of the day as specified either in 4.3.1.1 or in 4.3.3 of ISO 8601. The date and time shall be separated by the capital letter "T" as specified in 4.4.1 of ISO 8601. The alternate formats of 4.3.1.1 and 4.3.3 permit the optional inclusion of a time zone specifier (e.g. 2005-04-12T15:27:46-05:00)

- `iso_10303_26_author` : which has a data value of the `<user>`
- `iso_10303_26_organization` : which has a data value of the `<user_organization>`
- `iso_10303_26_originating_system` : which has a data value of the `<software_system_name>`
- `iso_10303_26_preprocessor_version` : which has a data value of the `<software_application_and_version>`

Additional HDF5 attributes may, of course, also be associated with HDF5 objects.

Warning:

The "iso_10303_26" prefix assumes that this specification will eventually be standardized as ISO 10303-26. If that is not the case, then the prefix will change.

Note:

The HDF5 Attribute Interface is used to create these attributes.

5.3. EXPRESS Schema Information as HDF5

For any encoding of EXPRESS-driven data as HDF5, an EXPRESS schema defines the context, or universe of discourse, for the data. That EXPRESS schema may use other EXPRESS schemas as part of its definition.

Note:

The following changed significantly between 0.4 and 0.5 of the mapping.

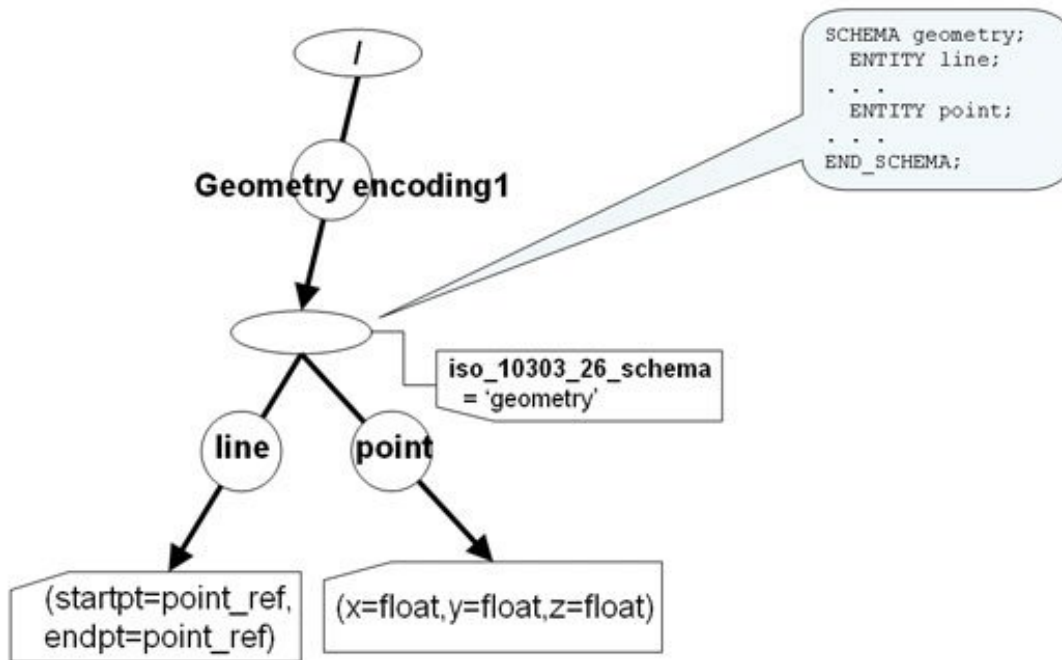
For all data that is written to HDF5 Files, HDF5 itself requires that the datatype(s) also be written to the HDF5 Files. Therefore, in order to precisely specify the HDF5 concepts used in the encoding of the EXPRESS population, aspects of the EXPRESS schema are also represented in the HDF5 File. The EXPRESS schema itself is not required and this specification does not require the encoded representation of the entire EXPRESS schema to be included in the HDF5 File.

Note:

The text, or any other representation, of the EXPRESS schema itself may be included in the HDF 5 file - as can many other types of data.

The aspects of the EXPRESS schema encoded in the HDF5 File shall be defined in an HDF5 Group. The location in the file for the HDF5 Group representing the EXPRESS schema is not constrained in this specification and that HDF5 Group may be shared between populations that are encoded in the same way (i.e. using the same HDF5 Datatypes, etc). The HDF5 Group that contains the representation of the EXPRESS schema shall have an HDF5 Attribute named "iso_10303_26_schema" with a value of <schema_id>. The same HDF5 Group may contain both the population and the datatypes derived from the EXPRESS schema.

EXAMPLE: The following figure shows an HDF5 Group containing the HDF5 Named Data Types related to the EXPRESS Schema for the population (i.e. the "Geometry_encoding1" HDF5 Group contains the definitions of the datatypes for data encoded based on the "geometry" schema).



Schema Information HDF5 Group

EXAMPLE : The Groups C Code shows the creation of HDF5 Groups and HDF5 Attributes representing the "geometry" schema and results in the h5dump output of the resulting HDF5 File.

5.4. EXPRESS Defined Type Information as HDF5

This section specifies the mapping of EXPRESS defined types that are not EXPRESS enumerations or SELECT types into HDF5.

The HDF5 Named Data Types that represent EXPRESS defined types are defined so that they have a parent which is the representation of the EXPRESS schema as specified in Schemas.

Warning:

The following naming changed between 0.4 and 0.5 of the mapping.

5.4.1. EXPRESS Simple Defined Types

For EXPRESS defined types that are specializations of EXPRESS simple datatypes, an HDF5 Named Data Type shall appear within the HDF5 Group representing the EXPRESS schema for the particular population being represented. The local name of the Named Data Type shall be as follows:

<user_defined_schema_group_name> + "/" + <type_id>

.

The HDF5 Datatype of the HDF5 Named Data Type is the HDF5 representation of the base type of the EXPRESS defined type as mapped in Datatypes.

EXAMPLE: The following EXPRESS defined type in an EXPRESS schema named "s" with a schema group named "s_encoding1"

```
SCHEMA s;  
  TYPE x = REAL;  
  END_TYPE;  
END_SCHEMA;
```

would map to an HDF5 Named Data Type with name = "s_encoding1/x" with an HDF5 base of HDF5 IEEE Floating Point 32 or 64 bits, Big- or Little-Endian.

The mapping for EXPRESS defined types that specialize other EXPRESS defined types works in a similar way. A new HDF5 Named Data Type is defined with an HDF5 base datatype of the other HDF5 Named Data Type.

EXMAPLE: The following EXPRESS

```
SCHEMA s;  
  TYPE positive_integer = INTEGER;
```

EXPRESS Data as HDF5 Mapping Specification Version 0.5

```
END_TYPE;  
TYPE big_positive_integer = positive_integer;  
END_TYPE;  
END_SCHEMA;
```

would map to an HDF5 Named Data Type with name = "s_encoding1/positive_integer" with an HDF5 base of HDF5 IEEE Floating Point 32 or 64 bits, Big- or Little-Endian and an HDF5 Named Data Type with name = "s_encoding1/big_positive_integer" with an HDF5 base datatype the Named Data Type with name = "s_encoding1/positive_integer".

Note:

The HDF5 Datatype Interface is used to create the HDF5 Named Datatypes.

EXAMPLE : The Defined/Enumeration C Code shows the creation of HDF5 Datatypes supporting simple EXPRESS Defined Type and Enumeration Types and results in the h5dump output of the resulting HDF5 File.

5.4.2. EXPRESS Array Defined Types

Warning:

The following type = array type was added between 0.4 and 0.5 of the mapping.

For EXPRESS defined types that are specializations of EXPRESS array datatypes (and only array data types to any level of nesting), an HDF5 Named Data Type shall appear within the HDF5 Group representing the EXPRESS schema for the particular population being represented. The local name of the Named Data Type shall be as follows:

<user_defined_schema_group_name> + "/" + <type_id>

.

EXAMPLE: The following EXPRESS defined type in an EXPRESS schema named "s" with a schema group named "s_encoding1"

```
SCHEMA s;  
TYPE context_dependent_measure = REAL;  
END_TYPE;  
TYPE anisotropic_symmetric_tensor2_2d =  
  ARRAY [1:3] OF context_dependent_measure;  
END_TYPE;  
END_SCHEMA;
```

would map to an HDF5 Named Data Type with name = "s_encoding1/anisotropic_symmetric_tensor2_2d" with an HDF5 base of HDF5_ARRAY of HDF5 IEEE Floating Point 32 or 64 bits, Big- or Little-Endian because "s_encoding1/context_dependent_measure" was mapped that way.

EXAMPLE : The Type Array C Code shows the creation of an HDF5 Array Datatype for an EXPRESS Defined Type and results in the h5dump output of the resulting HDF5 File.

5.4.3. EXPRESS Bag, List and Set Defined Types

Warning:

The following type = bag/set/list type was added between 0.4 and 0.5 of the mapping.

All EXPRESS aggregation types are represented using HDF5 Array type. For EXPRESS Arrays with fixed bounds in the schema, the HDF5 Array datatype can be defined directly. However, for all other EXPRESS aggregates, the dimension of the HDF5 Array depends on the actual population of the data and therefore is not derivable directly from the EXPRESS schema. The creation of the HDF5 Array used to store the data requires the size of the array to be specified.

For each instance of an aggregate EXPRESS defined type that includes a BAG, SET or LIST anywhere in its definition, an HDF5 Array datatype is used to store each instance of the EXPRESS BAG, SET or LIST. An HDF5 Named Data Type representing it, as is required for EXPRESS array defined types, need not be included in the HDF5 File.

EXAMPLE: The following EXPRESS defined type in an EXPRESS schema named "s" with a schema group named "s_encoding1"

```
SCHEMA s;  
  TYPE real_list = LIST [1:?] OF REAL;  
  END_TYPE;  
  ENTITY x;  
    x_real_list : real_list;  
  END_ENTITY;  
END_SCHEMA;
```

would map a set of HDF5 Array datatypes, not necessarily named although this specification does not stop them from being HDF5 Named Datatypes, with an HDF5 base of HDF5_ARRAY of HDF5 IEEE Floating Point 32 or 64 bits, Big- or Little-Endian where the dimension of the HDF5 Array would vary for each datatype depending on the number of elements list for each instance of the "x" EXPRESS entity type. This specification does not require that datatype to be defined within the "s_encoding1" HDF5 Group.

EXAMPLE : The Type List C Code shows the creation of a set of unnamed HDF5 Array Datatypes for an EXPRESS Defined Type and results in the h5dump output of the resulting HDF5 File.

5.5. EXPRESS Enumeration Types as HDF5

EXPRESS enumeration types are mapped in the same way as other simple EXPRESS defined types in that they are represented as an HDF5 Named Data Type. However, the HDF5 Datatype is specified directly as an HDF5 ENUM which is an <string>:<unsigned integer> pair where <string> represents the symbolic name that shall be assigned a value of <enum_name> + "/" + <enum_literal>. Because of EXPRESS Edition 2 Extensible Enumeration Types, there is no ordering that can be related to an EXPRESS enumeration literal. Therefore, references must be made using the symbolic name. For that reason, the integer values to be used are not standardized.

Warning:

The following naming changed between 0.4 and 0.5 of the mapping.

EXAMPLE: The following EXPRESS enumeration related to a schema representation "s_encoding1" would result in the HDF5 Named Data Type "s_encoding1/ahead_or_behind" being defined and pairs ("s_encoding1/ahead_or_behind/ahead", 0), ("s_encoding1/ahead_or_behind/exact", 1) and ("s_encoding1/ahead_or_behind/behind", 2) representing the EXPRESS enumeration items, remembering that the integer values 0, 1 and 2 should not be used as a reference.

```
SCHEMA s;  
  TYPE ahead_or_behind = ENUMERATION OF  
    (ahead, exact, behind);  
  END_TYPE;  
END_SCHEMA;
```

For each EXPRESS enumeration type that is extensible, each possible EXPRESS enumeration literal shall be mapped as if it were defined locally.

EXAMPLE: The following EXPRESS enumerations related to a schema representation "s_encoding1":

```
SCHEMA s;  
  TYPE x = EXTENSIBLE ENUMERATION OF (a,b);  
  END_TYPE;  
  TYPE y = ENUMERATION BASED_ON x WITH(c,d);  
  END_TYPE;  
  TYPE z = ENUMERATION BASED_ON x WITH(e,f);  
  END_TYPE;  
END_SCHEMA;
```

would result in the following being HDF5 definitions:

- the HDF5 Named Data Type "s_encoding1/x" being defined with six pairs ("s_encoding1/x/a", 0), ("s_encoding1/x/b", 1), ("s_encoding1/x/c", 2), ("s_encoding1/x/d", 3), ("s_encoding1/x/e", 4) and ("s_encoding1/x/f", 5) representing

the EXPRESS enumeration literals.

- the HDF5 Named Data Type "s_encoding1/y" being defined with four pairs ("s_encoding1/y/a", 0), ("s_encoding1/y/b", 1), ("s_encoding1/y/c", 2) and ("s_encoding1/y/d", 3) representing the EXPRESS enumeration literals.
- the HDF5 Named Data Type "s_encoding1/z" being defined with four pairs ("s_encoding1/z/a", 0), ("s_encoding1/z/b", 1), ("s_encoding1/z/e", 2) and ("s_encoding1/z/f", 3) representing the EXPRESS enumeration literals.

EXPRESS enumerations values for attributes are mapped in the section on Datatypes.

EXAMPLE : The Defined/Enumeration C Code shows the creation of the HDF5 Datatypes supporting the simple EXPRESS Defined Type and Enumeration Type and results in the h5dump output of the resulting HDF5 File.

5.6. EXPRESS Select Types as HDF5

EXPRESS select types are typically used in a variety of ways but the current mapping only supports the following uses:

1. a select type consisting of only selections of entity types and other select types that consist of only entity types (i.e. GENERIC ENTITY select types)
2. a select type consisting of only selections of defined data types that all resolve to the same underlying simple or aggregation data type (e.g.three TYPE = REAL definitions)

5.6.1. EXPRESS Select of Simple Types as HDF5

Warning:

The following naming changed between 0.4 and 0.5 of the mapping.

In the case where all items in the EXPRESS select type resolve to be based on a simple data type, then the mapping works as if it were an EXPRESS defined type based on that simple type. The mapping is applied whether the EXPRESS select type is extensible or not so long as all (extended) select items resolve to the same EXPRESS simple data type. An HDF5 Named Data Type shall appear within the HDF5 Group representing the EXPRESS schema for the particular population being represented. The local name of the Named Data Type shall be as follows:

<user_defined_schema_group_name> + "/" + <type_id>

.

The HDF5 Datatype of the HDF5 Named Data Type is the HDF5 representation of the base type of the EXPRESS defined type as mapped in Datatypes.

EXAMPLE: The following EXPRESS select type in an EXPRESS schema representation named "s_encoding1"

```
SCHEMA s;  
  TYPE x = REAL;  
  END_TYPE;  
  TYPE y = REAL;  
  END_TYPE;  
  TYPE z = SELECT( x, y );  
  END_TYPE;  
END_SCHEMA;
```

would map to three HDF5 Named Data Types named = "s_encoding1/x", "s_encoding1/y" and "s_encoding1/z" with an HDF5 base of HDF5 IEEE Floating Point 32 or 64 bits, Big- or Little-Endian.

EXAMPLE : The SELECT of Simple Defined C Code shows the creation of the HDF5 Datatypes supporting an EXPRESS SELECT Type of other Defined Types that all resolve to the same underlying simple datatype and results in the h5dump output of the resulting HDF5 File.

5.6.2. EXPRESS Select of Entity Types as HDF5

Warning:

The Select of entity types mapping has not yet been tested but is proposed to initiate discussion.

In the case where all items in the EXPRESS select type resolve to be EXPRESS entity data types, the EXPRESS select type is not really mapped to the HDF5 file. Instead, all EXPRESS attributes that have the EXPRESS select type as their domain are defined as having a generic HDF5 Region Reference as their type.

EXAMPLE: The following EXPRESS select type in an EXPRESS schema representation "s_encoding1"

```
SCHEMA s;  
  ENTITY x;  
    a: REAL;  
  END_ENTITY;  
  ENTITY y;  
    b : INTEGER;  
  END_TYPE;  
  TYPE z = SELECT( x, y );  
  END_TYPE;  
END_SCHEMA;
```

would map to two entity data types as specified elsewhere for "x" and "y" but no new

datatype is defined for "z".

5.7. EXPRESS Entity Type Information as HDF5

This section specifies the mapping of EXPRESS entity type information into HDF5.

Warning:

The following naming changed between 0.4 and 0.5 of the mapping.

For each EXPRESS entity type, an HDF5 Named Data Type that is an HDF5 Compound Type shall appear within the HDF5 Group representing the EXPRESS schema for the particular population being represented. The local name of the HDF5 Named Data Type shall be as follows:

<user_defined_schema_group_name> + "/" + <entity_id>

.

The EXPRESS entity type information being represented shall include information about all explicit attribute, including all inherited explicit attributes. The details of this representation are as follows.

- The type of each EXPRESS explicit attribute of the EXPRESS entity type, including inherited EXPRESS explicit attributes, is represented as a member (or HDF5 Field) of the HDF5 Compound Type.
- The name of each member of the HDF5 Compound Type is the name of the EXPRESS explicit attribute, preserving the case.
- The type of each member of the HDF5 Compound Type is an HDF5 Datatype corresponding to the EXPRESS type of the EXPRESS attribute as specified in the Datatypes, Arrays and Aggregates sections of this specification.

EXAMPLE: The following EXPRESS entity types related to a schema representation "s_encoding1":

```
SCHEMA s;  
  ENTITY x;  
    name : STRING;  
  END_ENTITY;  
  ENTITY y SUBTYPE OF (x);  
    age : INTEGER;  
  END_ENTITY;  
END_SCHEMA;
```

would result in the following being HDF5 definitions:

- the HDF5 Named Data Type "s_encoding1/x" being defined with a single HDF5 Field

- named "s_encoding1/x/name";
- the HDF5 Named Data Type "s_encoding1/y" being defined with two HDF5 Fields named "s_encoding1/y/name" and "s_encoding1/y/age".

5.7.1. EXPRESS Attribute Redeclaration

In EXPRESS, inherited attribute may be redeclared which allows several changes. Handling these changes is described as follows.

- EXPRESS explicit attributes that redeclare inherited attributes are mapped based on the attribute domain in the redeclaring subtype.
- EXPRESS derived attributes that are redeclarations of inherited explicit attributes do not appear in the HDF5 Compound Datatype representing the subtype in which they are redeclared.
- EXPRESS explicit attributes that rename a redeclared inherited attribute are mapped using the new name as specified in the redeclaring subtype.

5.7.2. Mapping EXPRESS Non-Mutually Exclusive Entity Types

In EXPRESS, the default relationship between subtypes of an entity type is that they are not mutually exclusive (i.e. ANDOR). EXPRESS entity instances that are based on a combination of EXPRESS entity types are possible. The mapping in that case is that the same HDF5 constructs are defined for each combination that appears in the instance data.

```
SCHEMA test; ENTITY a; name : STRING; ENTITY b SUBTYPE OF a; age :  
INTEGER; x : REAL; ENTITY c SUBTYPE OF a; height : REAL; x: BOOLEAN;  
Results in test/a test/a/name test/b test/b/name test/b/age test/c test/c/height  
test/b__c test/b__c/name test/b__c/age test/b__c/height test/b__c/b__x  
test/b__c/c__x
```

An HDF attribute of type BOOLEAN named "isComplex" to signify that it is a synthetic type shall be added to the b__c Group.

5.8. Simple EXPRESS Entity Instances as HDF5

Within a particular population of an EXPRESS schema, the complete set of EXPRESS entity instances (e.g. the entity extent) of the same EXPRESS entity type(s) is represented within an HDF5 Group.

The local name the HDF5 Group for the instances of each EXPRESS entity type shall be as follows.

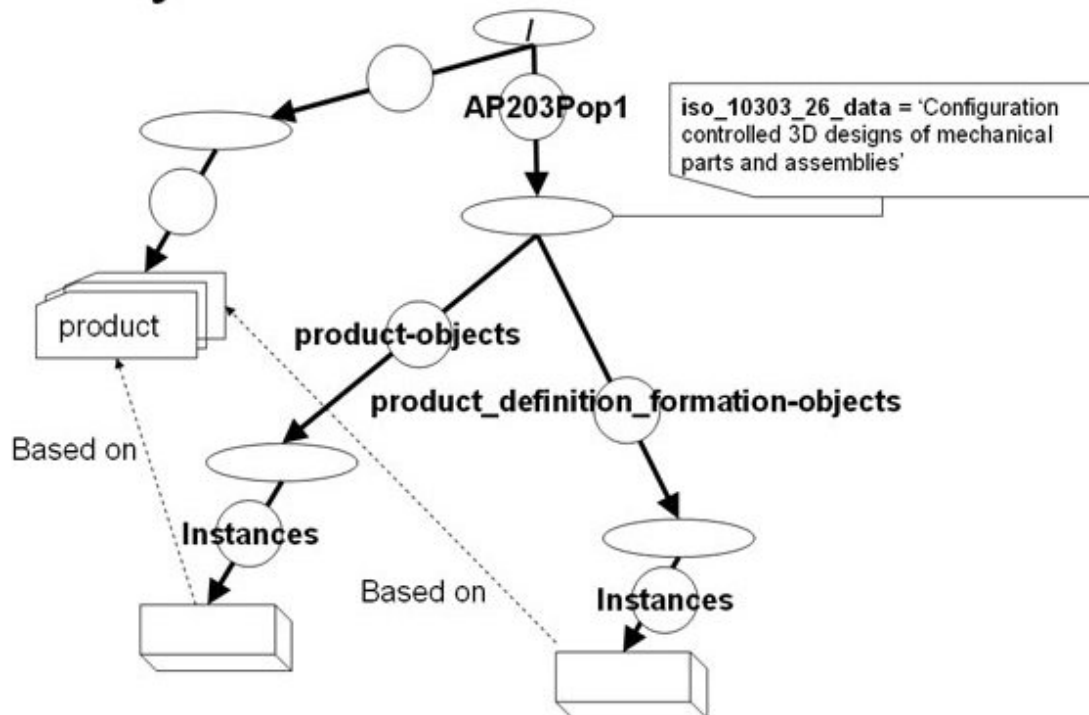
```
<entity_id> + "-objects"
```

The HDF5 Group related to the EXPRESS entity type contains an HDF5 Dataset that

contains every EXPRESS entity instances of the particular entity type. The local name the HDF5 Dataset containing the EXPRESS entity instances shall be as follows.

`<entity_id> + "-objects" + "/" + "Instances"`

The following figure is an overview, not showing the details, that shows two HDF5 Datasets, one for EXPRESS entity instances of the entity type "line" and the other for EXPRESS entity instances of the entity type "point". The details of the HDF5 Datatypes, HDF5 Dataspaces, and HDF5 Array are explained elsewhere in this specification.



Simple Entity Instance Datasets

The HDF5 Dataset that contains the EXPRESS entity instances is based on the HDF5 Named Data Type representing the EXPRESS entity type as specified in Entity Type Information as HDF5. That HDF5 Dataset must also have an associated HDF5 Dataspace to define its rank and dimension. The HDF5 rank is set to be 1 (the integer one), a single dimensional array contains the EXPRESS entity instances. The HDF5 dimension depends on the number of EXPRESS entity instances and on how the HDF Dataset is stored and so its value is not specified here.

For EXPRESS entity types that have no EXPRESS aggregate-valued attributes represented in

their own HDF5 Datasets, no further structures are required.

EXAMPLE : The Instance Entity Type point x,y,z = REAL C Code shows the creation of HDF5 Datatypes and data representing two instances of an EXPRESS entity type and results in the h5dump output of the resulting HDF5 File.

5.9. EXPRESS Entity Instances with related Aggregate Instances

Two approaches to mapping EXPRESS aggregate-valued attributes are defined in this specification.

1. For large aggregate instances, a separate HDF5 Dataset is used to represent the data.
2. For small aggregate instances, the same HDF5 Dataset that contains the EXPRESS entity extent is used to represent the data (i.e. they are embedded in the EXPRESS entity instance).

This means that software reading EXPRESS entity instances that have aggregate-valued attributes must examine the definition of the HDF5 Datatype representing the EXPRESS entity instance to see whether the aggregate data is defined in the HDF5 Compound Datatype as an HDF5 Object Reference (to an HDF5 Dataset as specified in EXPRESS Aggregate Instances Datasets) or as an embedded HDF5 Array Datatype.

The decision about which approach to use for EXPRESS aggregate-valued attributes must be made for the EXPRESS entity type - it is not made for each EXPRESS entity instance.

5.9.1. Embedded EXPRESS Aggregate Instances

For small aggregate instances, creating a separate HDF5 Dataset so that HDF5 can optimize access is not necessary. Embedding the EXPRESS aggregate instance in the representation of the EXPRESS entity instance saves the overhead of many small HDF Datasets being required.

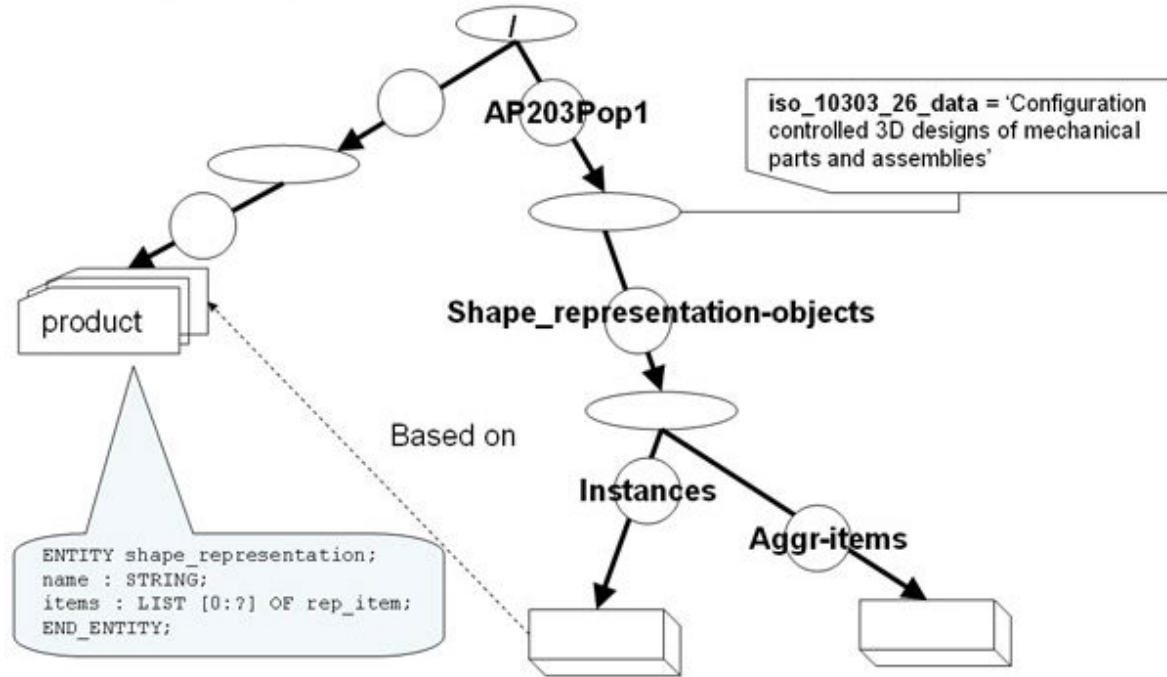
5.9.2. EXPRESS Aggregate Instances Datasets

The HDF5 Group related to the EXPRESS entity type also contains an HDF5 Dataset for each EXPRESS aggregate-valued attribute that is the value of an EXPRESS attribute (see Array Datatype Values).

The local name the HDF5 Dataset containing the EXPRESS aggregate instances shall be as follows.

```
<user_defined_schema_group_name> + "/" + <entity_id> +  
"-objects" + "/" + "Aggr-" + <attribute_id>
```

The following figure shows three HDF5 Datasets, one for EXPRESS entity instances of the entity type "line", the other for EXPRESS entity instances of the entity type "point" and one for the EXPRESS aggregate instances that are the value of the EXPRESS attribute "line.pts".



Entity Instance and Aggregate Instance Datasets

As each HDF5 Dataset represents a single EXPRESS aggregate instance, the HDF5 Dataset identifier itself is sufficient for representing the EXPRESS aggregate identifier.

5.10. EXPRESS Entity Instance Identifiers

There have been two proposals for Entity Instance Identifiers: 1) HDF5 Region Reference and 2) Tuple of Integers (HDF5 Object Identifier, HDF5 Array Row Index). Both are described below.

Region Reference Approach

The EXPRESS entity instance identifier used to refer to an EXPRESS entity instance as a data value is represented as an HDF5 Region Reference that includes a selection with the containing HDF5 dataset.

EXAMPLE: The following C code uses built-in HDF5 constructs for Region References :

hdset_reg_ref_t and H5T_STD_REF_DSETREG.

```
typedef struct line_t {
    hdset_reg_ref_t start_point;
    hdset_reg_ref_t end_point;
} line_t;
line_t line[2]; /* hold two instances of line */

...

line_tid = H5Tcreate (H5T_COMPOUND, sizeof(line_t));
H5Tinsert(line_tid, "start_point", HOFFSET(line_t, start_point),
H5T_STD_REF_DSETREG);
H5Tinsert(line_tid, "end_point", HOFFSET(line_t, end_point),
H5T_STD_REF_DSETREG);
status = H5Tcommit (schema_group, "line", line_tid);
```

For more on dataset regions, please see HDF Tutorial section "References to Dataset Regions" at <http://www.hdfgroup.com/HDF5/Tutor/reftoreg.html> and the related C language example at http://www.hdfgroup.com/HDF5/Tutor/examples/C/h5_ref2reg.c.

EXAMPLE : The Instance Entity Type line start_point, end_point = point and point x,y,z = REAL C Code shows the creation of HDF5 Datatypes, Selections, Region References and data representing two kinds of entity instances where one entity instance points to another. The EXPRESS entity type line referencing thpoints results in the h5dump output of the resulting HDF5 File.

Tuple of Integer Approach

5.11. EXPRESS Datatype Values as HDF5

The following table specifies the representation of data values for EXPRESS simple and enumeration datatype values using HDF5 (see HDF5 Datasets).

| EXPRESS Datatype Value | HDF5 Representation | Unset Value |
|------------------------|---|-------------|
| INTEGER | HDF5 Standard 8, 16, 32 or 64 bits, Signed or Unsigned, Big- or Little-Endian | -2147483647 |
| REAL | HDF5 IEEE Floating Point 32 or 64 bits, Big- or Little-Endian | 1.79E308 |
| BOOLEAN | HDF5 ENUM of the <string>:<integer> pairs BOOLEAN-TRUE:1, BOOLEAN-FALSE:0 | 2147483647 |
| LOGICAL | HDF5 ENUM of the <string>:<integer> pairs | 2147483647 |

| | | |
|-------------|---|---------------|
| | LOGICAL-TRUE:1, LOGICAL-FALSE:0, and LOGICAL-UNKNOWN:-1 | |
| STRING | HDF5 STRING if a width is specified, otherwise Variable-length datatype with base type HDF5 STRING (see Issues) | to be decided |
| BINARY | HDF5 OPAQUE if FIXED, otherwise HDF5 VARYING OPAQUE | to be decided |
| NUMBER | same as EXPRESS REAL representation | 1.79E308 |
| ENUMERATION | HDF5 ENUM of <string>:<integer> pairs where string is <user_defined_schema_group_name> + "/" + <enum_name> + "/" + <enum_literal> | 2147483647 |

Table 1: Summary of Mapping of EXPRESS Datatype to HDF5

5.12. EXPRESS Array Datatype Values as HDF5

A array that is the value of an EXPRESS attribute is represented as an HDF5 Dataset containing an HDF5 Array that contains the actual data. In HDF5 all elements of an Array must be of the same HDF5 type. The number of dimensions of the Array must also be specified (the HDF5 documents call this the "rank") and the dimensions themselves must be specified.

The HDF5 rank shall be an integer specifying the the nested-ness of the Array and the HDF5 dimensions are the number of elements of the array.

The EXPRESS array is referenced using an HDF5 Dataset Reference. This HDF5 Dataset Reference is the value stored for the EXPRESS attribute value, it must be dereferenced in order to read the elements of the array.

Warning:

It is left to the application reading and writing HDF5 encoded data to handle the differences between the HDF5 Array index and the EXPRESS ARRAY index.

EXAMPLE: The following EXPRESS snippet

ARRAY[1:2] OF ARRAY[1:3] OF INTEGER

would have an HDF5 rank of 2 and dimensions 2 and 3.

The following table specified the representation of N-dimensional array values of the same EXPRESS datatype.

| EXPRESS N-Dimensional Array Value Datatype | HDF5 Representation |
|--|--|
| INTEGER | H5T_ARRAY of base type for EXPRESS INTEGER |
| REAL | H5T_ARRAY of base type for EXPRESS REAL |
| BOOLEAN | same as for INTEGER |
| LOGICAL | same as for INTEGER |
| STRING | H5T_ARRAY of base type for EXPRESS STRING |
| BINARY | H5T_ARRAY of base type for EXPRESS STRING |
| NUMBER | same as for REAL |
| ENUMERATION | same as for INTEGER |

Table 1: Summary of Mapping of EXPRESS Array to HDF5

5.13. EXPRESS Aggregate Datatype Values as HDF5

FIXME ():

I think we need to name the Aggregate Instance Datasets.

FIXME ():

State that C ordering is how the dimensions of the array instance is to be interpreted.

Single-dimensional EXPRESS LIST, BAG and SET Datatypes that have a basic type that maps to the same HDF5 Datatype are mapped identically to single-dimensional EXPRESS ARRAYs.

Warning:

It is left to the application reading and writing HDF5 encoded data to handle the differences between the HDF5 Array index and the EXPRESS LIST, SET and BAG list position.

As the dimensions of many SETs, LISTs and BAGs are defined by the content of the population of data, not the schema, the dimensions of the HDF5 Array cannot be specified here. Instead, those dimensions must be set based on the data itself.

EXAMPLE: The following EXPRESS snippet

```
LIST[2:?] OF LIST[2:?] OF INTEGER
```

would have an HDF5 rank of 2 but its dimensions cannot be set from the information in the schema. If the data included two lists each containing three integers then the dimensions would be set to 2 and 3.

N-dimensional EXPRESS LIST, BAG and SET types are mapped to an HDF5 1-dimensional Array of Variable Length, Variable Length {,Variable Length} >HDF5 Datatype representation<.

EXAMPLE: The following EXPRESS snippet

```
LIST[0:?] OF LIST[0:?] OF point
```

would be represented by a 1-D Array of Variable Length, Variable Length Region References that refer to the representation of the EXPRESS "point" entity type.

6. Examples

Warning:

This section is incomplete for V0.3 so a V0.3a will be published containing the examples in the near future.

Providing test data for a binary data representation is difficult. For that purpose, this specification provides several simple test datasets using an XML format and related HDF5 DTD.

The details of the DataFromFile XML element are not specified in any detail in the DTD. In the sample datasets provided with this specification the following rules are followed.

1. CompoundType data is enclosed in "{" and "}".
2. ArrayType data values are enclosed in "[" and "]"

Example EXPRESS schemas and schema fragments, EXPRESS-driven data samples, and data samples encoded as HDF5 XML using the HDF5 DTD[2] follow.

A very small first example follows.

```
SCHEMA test0_line;  
ENTITY point;  
  x,y,z : REAL;  
END_ENTITY;  
ENTITY line;
```

```
points : ARRAY[1:2] OF point;  
END_ENTITY;  
END_SCHEMA;  
#1=POINT(1.0,2.0,3.0);  
#2=POINT(2.0,3.0,4.0);  
#3=POINT(5.0,6.0,7.0);  
#4=POINT(9.0,6.5,7.1);  
#5=LINE((#1,#2));  
#6=LINE((#2,#3));  
#7=LINE((#3,#4));
```

An XML representation of the HDF5 File structure and small dataset is available in the linked file: [test0_line sample HDF5 XML data](#).

7. EXPRESS-HDF5 Mapping Issues

This section describes the issues in mapping between ISO EXPRESS and HDF5.

7.1. Issues Against V 0.4

1. Variable length string : (1) HPdK thinks that these cannot be put in a HDF Compound Datatype. Georg found where it the UG seems to say this is allowed 7.1 Complex combinations of datatypes. Maybe it's a limitation of pyTables? (2)The current mapping says use Variable length datatypes but it's not clear if that's allowed in a Compound Datatype. (3) We may have to use the general purpose object id capability and have a dataset somewhere containing varying length strings (or find another solution). It does look like you may have to specify the maximum length of the varying length strings. (DEFER TO EMAIL WITH HDF)
2. Instance identifiers : (1) Every hdf5 link and hdf5 dataset has an hdf5 object id that is an unsigned 32/64 bit integer Issue : Is there a problem with using 64 bit integer as part of entity instance ids on a 32 bit platform (i.e. does this place a limit on file size or interoperabilty?) (2) H-P thinks the object ids are managed inside a hash table in HDF (3) Also thinks the object id is not exposed in the hdf API everywhere that we need it (4))Proposal is to use a tuple of integers that can be used for both an entity instance id and a pointer into the aggregates (hdf object id, row index)
3. We may need to add some HDF attributes to the Groups and Datasets when they are written to help readers (e.g. number of instances of an entity type that were written) C Tables API uses this approach so we should look at that to see if we can learn anything for our use.
4. We need to have more discussion about whether to allow or require writing inverse attribute values into the file, nothing is done there now. For read-only files inverses could be a nice optimization. Would we need to allow this to be configured? If so, how? What about the unnamed inverse that EXPRESS says exists?

7.2. Issues Against V 0.2

1. How to handle n-D LISTS? Proposed resolution : For LIST OF LIST OF point use 1-D array of variable length, variable length point references (VL OF VL OF point is the HDF5 Datatype).
2. What is the primary use case? The mapping depends on the answer to that question. For example, insert into a LIST stored as an HDF5 Array is very expensive.

7.3. Issues Against V 0.1

An issues log against this V 0.1 mapping will be created and documented here so if you have issues, please send them to the team.

1. How do I know which HDF5 Groups in the file contain EXPRESS-driven data?
RESOLVED : Use iso_10303_26_data HDF5 attribute.
2. How are EXPRESS entity instance identifiers represented? Proposed Resolution : use HDF5 Region References. Note that a second approach of using a compound identifier containing the HDF5 Dataset name and an integer index into that Dataset is worth investigation. RESOLVED : HDF5 Region References.
3. How is optionality addressed? Open Issue

7.4. Initial Issues from V 0.1 development

1. The handling of STRING need further study. NATIVE types may make data not interoperable across platforms.
2. What exactly does HDF5 BITFIELD mean and is it OK for EXPRESS binary?
RESOLVED : Use HDF5 OPAQUE
3. We need to allow including raster files, etc.
4. In STEP there are references to external files so how would we make that reference if the external file was another dataset inside the same HDF5 file?
5. Do we want to base this work on SDAI or be based on native HDF5? : RESOLVED : Native HDF5 and Schema-specific if required.

8. Relationship to ISO 10303-21 Part 21 Clear text encoding

This section briefly explains how Part 21 and this specification are related.

| Part 21 Concept | HDF5 Concept |
|---------------------------|--|
| Data Section | The HDF5 Group containing a data population based on a single schema (see the section on populations). |
| Header Section Attributes | HDF5 Attributes associated with the HDF5 Group containing a data population based on a |

| | |
|--|---|
| | single schema (see the section on populations). |
| Part 21 object Identifier #<n> | HDF5 Region Reference (see the section on EXPRESS Entity Instances and EXPRESS Entity Instance Identifiers). |
| Part 21 aggregate groups (...) and ((...), (...)) | Aggregates have identifiers in HDF5. An HDF5 Dataset contains the N-dimensional aggregate values (see the section on array values). |

Table 1: Part 21 Concepts Mapped to HDF5 Concepts

9. Bibliography

[1] Introduction to HDF5, November 2005 [cited 2006-10-01]. Available from World Wide Web: <http://hdfgroup.com/HDF5/doc/H5.intro.html>.

[2] The HDF5 XML Information Page, March 2005 [cited 2006-10-01]. Available from World Wide Web: <http://hdfgroup.com/HDF5/XML/>.

[3] HDF5 Tutorial [cited 2006-10-01]. Available from World Wide Web: <http://hdfgroup.com/HDF5/Tutor/index.html>.