SNIA OSD TWG document
cross posted to T10

T10/03-280r1

Sami Iren
Erik Riedel
Seagate Technology

Mikhail Zelikov
EMC

David Nagle
Garth Gibson
Panasas

Julian Satran
Richard Golding
IBM

Aug 20th , 2003

OSD Grouping and Attributes

Version 0.6

0 Revision History

Version	Date	Changes
0.1	6/11/2003	Initial draft
0.2	7/29/2003	Made changes based on the new "Collections" document and feedback on attributes
0.3	8/5/2003	Expanded the section on grouping and cleaned up rest of the document to be consistent with this section. Added a section on reference pages. Incorporated other feedback from the document review.
0.4	8/13/2003	A completely new approach on attribute templates and references was introduced in Section 9.1. A new section on "ideas for next version of the standard" was introduced in Section 9. Minor editorial changes were made.
0.5	8/14/2003	Attribute templates and references were delayed until next version of the standard. Therefore, related section is moved to the end of the document under "ideas for future version of the standard" section. Editorial changes were made throughout the document.
0.6	8/20/2003	Clarifications were put in for collections

1 Introduction

The purpose of this document is to summarize the consensus that was reached by the committee on OSD grouping and attributes. It is intended to provide a summary of the key points and how it differs from the current version of the T10 spec[1]. It does not fully define how grouping and attributes should be implemented. For a complete understanding of the issues, the following documents must be read as a prerequisite to this document:

- T10 Spec [1]
- ObS-Grouping [2]
- ObS-Identifying Objects [3]
- ObS-Collections [4] (This document should be read to find out more about the history of collections, not as the reference for standards and implementation. The document you are currently reading (OSD Grouping and Attributes) contains the most up to date information on the state of collections and how they should be implemented).

Future versions of the T10 Spec should be revised based on this document.

2 OSD Grouping

The T10[1] spec defines the purpose of groups as to contain a list of user objects that share some common attributes. It also specifies one type of group called "object groups" each one of which is represented by a group object. An object_group_id uniquely identifies each object group.

Further discussions in the applications subgroup pointed to a need for a two layer grouping initially termed as "hard" and "soft" groups. This concept was materialized in [2] in the form of partitions (hard groups, this corresponds to the current definition of the term "group" in the T10 spec [1]) and collections (soft groups, this is a new concept and is not currently defined in the T10 spec[1]). We will adopt this new terminology in this document and avoid using the term group. Instead we will use the terms partitions and collections as defined below.

2.1 Partitions

A *partition* is a division of storage device space for creating distinct management domains (e.g., for naming, security, space). Each object lives in exactly one partition. Partitions are represented by partition objects and a partition_object_id uniquely identifies each partition object. OSD implementations must support partitions.

The following commands are defined for partitions:

- CREATE OBJECT PARTITION: Creates a new partition. This command replaces the CREATE OBJECT GROUP command that currently exists in the T10 spec[1].
- REMOVE OBJECT PARTITION: Removes a partition. If there are user objects living in the partition, this operation will fail and return an error message. This

command replaces the REMOVE OBJECT GROUP command that currently exists in the T10 spec[1].

- LIST: Returns the list of all the objects that live in a partition.

2.2 Collections

While partitions are implemented for security, quotas, and naming, collections are implemented mainly for fast indexing and multi-object operations. An object can be a member of zero or more collections at the same time. A collection is wholly contained within exactly one partition. A partition may contain zero or more collections. OSD implementations may or may not support collections.

2.2.1 Representation

Collections are represented as objects. A "collection_object_id" uniquely identifies each collection. Collections live in the same name space as other user objects. That is, a given object id will uniquely identify a single object be it either a user object or a collection object. Since both user objects and collection objects live in the same name space, an "object-type" attribute separates collection objects from other user objects.

The content of the collection object is the ids of the objects (64-bit object id only, partition id is not needed) that belong to this collection. There is a back pointer on each object (in the form of an attribute) to the collection objects that this user object belongs to. A new attribute page called Collections Page is defined. The attributes in this attribute page will be ids of the collection objects (64-bit object id only, partition id is not needed) this user object belongs to. There are 2^32 possible attributes in this page. However, OSD implementations may limit this number (that is, the number of possible collections an object can be a member of at the same time) by setting an attribute on the root information page.

Membership to a collection will be accomplished by using SET ATTRIBUTE command on the collection page of the user object. Note that, before doing setattr on an attribute of the collections page, the client has to make sure it does not override a previous value by mistake and the OSD has to make sure that no duplicate entries exist on this page. Removal from a collection is again accomplished with a setattr on the attribute containing the collection id with a value of zero.

A delete on a collection object with members might result in dangling references in the collections attribute page. One way of preventing this is to reject the delete operation on collection objects with members. This will force the client to remove the user objects from the collection first (by doing setattr on the collections page of each member object). However, an option should also be provided if the client wants to remove this collection object regardless. Therefore, a "force" flag in the CDB will result in collection object deletion regardless of the existing members. The client is responsible for fixing the collections page attributes of the member objects later. When this flag is not set, the delete operation will be rejected.

2.2.2 Operations

The following new commands are introduced for operation on collection objects:

- CREATE OBJECT COLLECTION: Given the partition id (and optionally object id if user-defined ids are being used), this command creates a new collection.
- REMOVE OBJECT COLLECTION: Given the partition id and the object id, this command removes a collection. Note that this command will fail if there are member objects (i.e., user objects living in this collection). However, if the "force" flag is set, this command will succeed regardless of the member objects.
- LIST OBJECT COLLECTION: Given the partition id, this command returns a list of all the collection object in a given partition. If the collection object id is also provided, it lists the member objects of the collection.

The following commands will be used on user objects for membership to collections:

- SET ATTRIBUTE: When executed on the collections page of a user object, this command is used to add/delete an object to/from a collection.
- GET ATTRIBUTE: When executed on the collections page of a user object, this command lists the collections this object is a member of.

3 Attributes Terminology

Three types of attributes are assumed throughout this document:

1. **Internal attributes**: These are the attributes that OSD maintains for internal management, for example, object size, last access time, etc. These attributes can be accessed by the client via "GET ATTRIBUTES" command, but cannot be set (e.g., via "SET ATTRIBUTES" command). Internal attributes are defined by the T10 spec.
2. **Opaque attributes**: These are the attributes that are attached to the object and interpreted by the client only. They have no meaning for OSD itself. Clients can GET and SET these attributes.
3. **Shared attributes**: These are the attributes that are interpreted by both the client and OSD. A typical example for this category of attributes is quality of service (performance) related attributes that the OSD device may take into account when storing objects. Clients can GET and SET these attributes.

4 Attributes Overview

OSD object attributes allow the association of metadata with any OSD object (e.g., root, partition, user objects). Object attributes may be used to describe specific characteristics of an object including the total amount of bytes occupied by the object, logical size, the time object was created, and the time it was last accessed.

Attribute values can be accessed (get/set) via the GET ATTRIBUTE and SET ATTRIBUTE commands. Also, any OSD service action on an object may get/set attributes of the object it is accessing via the "get attributes parameters" and "set attributes parameters" fields in the CDB. A client cannot get/set attributes of objects other than the object it is currently accessing.

4.1 Attribute Pages

Object attributes are organized in pages for identification and easy reference. An attribute page is a grouping of logically related attributes. For example, the timestamps attribute page contains all the attributes related with timestamps in a single page. The attributes within a page are referenced with an attribute number. Each attribute page is associated with one of the following objects:

1. The root object
2. A partition object
3. A user object
4. A collection object
5. Any other object type that might be introduced in future versions of the T10 spec.

Identifying numbers are assigned to object attribute pages with ranges of page numbers indicating the type of object with which an object attributes page is associated as shown in Table 1.

Page Numbers	OSD Object type with which the object attributes page is associated
0000 0000h – 2FFF FFFh	User object
3000 0000h – 5FFF FFFFh	Partition object
6000 0000h – 8FFF FFFFh	Collection object
9000 0000h – BFFF FFFFh	Root object
C000 0000h – FFFF FFFEh	Reserved
FFFF FFFFh	All attributes pages

Table 1 - Object Attribute Page Numbers

Table 2 shows how the page number space for user object type is allocated between standards bodies, manufacturers, vendors, and applications. The number space for other object types (e.g., root, partition, etc.) are divided in the same manner.

Page Number	Description	Type
0000 0000h – 0000 007Fh	Defined by this standard	Fixed
0000 8000h – 0000 EFFFh	Defined by other standards	Fixed
0000 F000h – 0000 FFFFh	Defined by OSD manufacturer product specifications	Fixed
0001 0000h – 1FFF FFFFh	Dynamically defined by applications	Dynamic
2000 0000h – 2FFF FFFFh	Vendor specific	N/A

Table 2 - Attribute Page Number Allocations

4.2 Attribute Type (How are Attributes Defined?)

There are two types of attribute page structures. The first, standard pages, are defined by the standards bodies (e.g., SCSI OSD by T10) or the vendor of the OSD. These pages,

the attributes within these pages, and their addresses are not changeable by the OSD or the client of the OSD.

The second type, dynamic pages, are intended for use by the client of the OSD. Unlike standard pages, dynamic pages are not pre-defined and all the possible pages in the range specified in Table 2 are assumed to exist. Since these pages are not pre-defined, client has to specify the attribute length with every SET operation and OSD should return this value with every GET operation. For these dynamic pages, the list entry format specified in Table 98 (section 7.1.3.5) of the T10 Spec[1] will be used. In summary, the definition of these pages change dynamically (hence the name Dynamic Pages) with client commands (SET operations) and OSD has to keep track of these definitions. If a GET is issued on an attribute that has not been set before, an attribute length of zero will be returned by OSD indicating the attribute has not been set before.

5 Attribute and Attribute Page Operations

5.1 Operations to be Removed from the Spec

The following two operations that are currently defined in the T10 spec will be removed from the spec since we assume dynamic attribute pages exist at all times and there is no need to explicitly create them:

- CREATE ATTRIBUTES PAGE
- REMOVE ATTRIBUTES PAGE

5.2 Operations to be Kept in the Spec with Modifications

The following operations are kept as they are defined in the spec with slight modifications as described below:

- GET ATTRIBUTES
- SET ATTRIBUTES

The modifications on these commands are:

- They can operate only a single object. That is, they cannot get/set attributes of multiple objects with a single command
- The CDB parameters have been changed as described in 5.4. Now the parameters include either a single attributes page (versus two attributes pages) or an attributes list (versus an attributes page and an attributes list). A single attribute page is used for standards defined attribute pages. A list can be used for both standards defined and dynamic attribute pages.

5.3 Combining Data and Attribute Operations

The “Get attributes parameters” and “set attributes parameters” fields of the CDB can be used to get/set attributes of the object currently referenced with any service action.

Users should be aware of the fact that co-locating data and attribute commands might cause performance hits. However, implementations must not incur any performance hits when clients are not using this feature.

When "get attributes parameters" field of the CDB is used in a data command, the returned attribute values will reflect the values *after* the data operation and "set attributes" operation, if any, were performed. For example, if "last modified time" attribute is requested in the "get attributes parameters" field in a WRITE operation, the returned value will reflect the last modified time *after* the WRITE operation. Specifically, the following order will be followed when operating on attributes:

1. They will be updated as a result of any set attributes parameters field (if any)
2. They will be updated as a result of any data operation (e.g., WRITE) (if applicable)
3. They will be read for get attributes parameters field (if any)

An exception to the above rule is the REMOVE , REMOVE OBJECT PARTITION, AND REMOVE OBJECT COLLECTION operations. Since we cannot do any attribute operations after the object was removed, the attribute operations are performed first (setattr first, then getattr) and then the object is removed.

5.4 CDB Parameters for Attributes

Attributes can be accessed either by specifying the attribute page numbers (the whole page is accessed) or by constructing an attribute list (individual attributes from different pages can be accessed). An attribute list acts on one or more individual attribute values using attribute page and attribute number tuples. It is a mechanism to access multiple attributes in a single command. Attribute lists can only retrieve attributes from a single object at one time.

Note: [SI] This is different than what is specified in T10. The current version in T10 allows access to attributes from multiple objects in a single command. This should change.

The current spec [1] uses a CDB parameter definition that allows either two attribute pages or one attribute page and an attribute list. We are changing this definition to be either a single attribute page or an attribute list.

6 Partitions, Collections, and Inheritance

A range of attribute page numbers are reserved for partitions/collections. Several attribute pages are defined for partitions in the T10 spec and more can be defined by future versions of this standard, other standards bodies, and vendors. These attributes are used for the partition/collection only and not for the objects in that partition/collection. Hence, they are not dynamically inherited to the objects in that partition (i.e., **no inheritance**). However, they can be used to initialize attributes of the user objects created under that partition if stated so in the definition of the attributes page.

Since collection is a new concept, the spec does not define any attribute pages for it yet. An appropriate set of attribute pages and attributes for collections needs to be added to the T10 spec [1].

7 Object Type Attribute

Collection objects share the same name space as other user objects. A mechanism is needed to distinguish this type of special objects (and any special future objects) from the regular user objects. Towards this end, a new attribute is defined called "object type". The current possible values for this attribute are:

- 0 – reserved
- 1 – for user objects
- 2 – for collection objects
- 3-255 – reserved

Note that this is a read-only attribute and updated only by OSD. For example, a CREATE OBJECT COLLECTION command creates an object and sets this attribute to 2, whereas a CREATE command creates an object and sets this attribute to 1. When a LIST command is executed on a partition, the device will return the list of user objects (object type = 1). For other object types special LIST commands are defined (e.g., LIST OBJECTS COLLECTION).

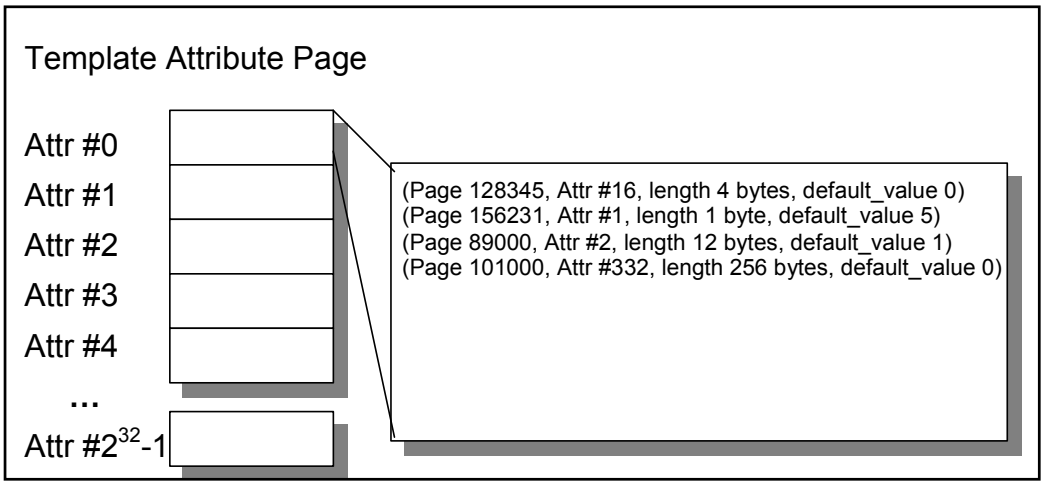
8 Additional Changes to Definitions of OSD Attribute Pages and Attribute Parameters

The following additional changes are suggested in the T10 spec:

- The used capacity attribute of the root information attributes page should contain the number of bytes used by all the objects on the device including the user objects. Currently it only keeps the bytes used by the root object.
- Group information attributes page should have a new attribute called "OSD security level" just like the root information attributes page does. This is a user-settable attribute and will determine the security level of the partition.
- Group information/directory/resources/timestamps pages should be renamed as Partition information/directory/resources/timestamps.

9 Ideas for Next Version of the Standard: Temporary Objects, Garbage Collection, Expiration Date, Secure Erase, etc.

This section lists a couple of ideas that we think are useful and should be part of the standard but not included in the current version (version 1) of the standard due to lack of time for a through review. These ideas will be revisited for version 2 of the standard.



9.1 Attribute Templates and References

To optimize the performance of attribute storage and retrieval, the T10 OSD Spec includes two new concepts, Attribute References and Attribute Templates. An Attribute Reference describes a set of attributes that can be fetched (i.e., GetAttr) or set (i.e., SetAttr) by a single Attribute Reference, allowing for a concise representation in a command CDB. An Attribute Template describes the set of attributes that are commonly accessed, allowing the OSD to efficiently organize the most commonly accessed dynamic attributes.

9.1.1 Attribute Templates

While the virtual attribute namespace is very large (2^32 attribute pages; 2^32 attributes per page), only a few attributes are accessed frequently. For efficient implementations, it is important to disclose to the OSD which attributes are frequently accessed, allowing the OSD to optimize attribute layout. Therefore, attribute templates are essentially a hint to the OSD about which attributes will be frequently accessed.

An Attribute Template is a set-of-{Attribute Page, Attribute Number, Attribute Size, Default Value} tuple. Each Attribute Template is stored as a single attribute on a special attribute page called the Attribute Template Page (see below). An Attribute Template is created using the ***Create Attribute Template*** command. The CREATE and CREATE OBJECT PARTITION commands that utilize these templates reference an Attribute Template using the {Attribute Template Page #, Attribute Number} tuple.

9.1.1.1 Attribute Template Page

The Attribute Template Page should be a T10 Standard defined page associated with every Root- and Partition-objects. An Attribute Template Page Entry will be a List-entry format description with the following fields: page number, attribute number, length, and default value. This provides the user with space for $2^{32}-1$ Attribute Templates per Root- or Partition-object. Attribute template page is only accessible via the special commands

described below. A setattr/getattr on this page will return an error message. Also note that if an attribute is defined in the template that was used at the creation of the object, definition (size) of that attribute cannot be changed later via setattr command.

9.1.1.2 Attribute Template Commands

The following commands are defined for attribute templates. These are the only ways to access the attribute template page:

- CREATE ATTRIBUTE TEMPLATE: The ***Create Attribute Template*** command is a new command that provides the OSD with the list of attributes associated with the template. The attribute list is sent in a List-entry format, which is the {Attribute Page, Attribute Number, Attribute Size, Default Value} tuple. The Create Attribute Template command specifies the partition id and the Attribute Number on the Attribute Template Page where the new Attribute Template should be stored. If a template already exists at the corresponding Attribute Number, the old template is overwritten with the new template.
- REMOVE ATTRIBUTE TEMPLATE: Given the partition id and an attribute number on the Attribute Template Page, the ***Delete Attribute Template command*** deletes the associated Attribute Template is deleted from the specified OSD root-, or partition-object.
- LIST ATTRIBUTE TEMPLATE: Given the partition id and an attribute number on the Attribute Template Page, this command returns the template definition in the List-entry format.

9.1.1.3 Using Templates

The CREATE and CREATE OBJECT PARTITION commands specify an attribute from the Attribute Template Page that the OSD will use to determine the most-commonly accessed attributes that should be optimized for efficient access. The Current Command Attribute Page can be used to specify what template to use. For partition-objects, the CREATE OBJECT PARTITION Command will specify a Root-object Attribute Template Page Attribute that holds the appropriate template; for user-objects the CREATE command will specify a Partition-object Attribute Template Page Attribute that holds the appropriate template.

9.1.2 Attribute References

Recall that attributes are organized into separate pages, based on the type of attribute information (e.g., timestamps are located on the timestamp attribute page, security information on a security attribute page, capacity used on the resource attribute page, etc.). A single command, however, often needs to set or get information from many different attribute pages (e.g., last-time-accessed, capacity-used), requiring the command to enumerate all attributes using the long-form list entry format {attribute page #, attribute #, attribute length, attribute value}. Unfortunately, enumerating every attribute using the list-entry format can create very large Capabilities that exceed the size of the SCSI CDB. An Attribute Reference provides a very concise encoding by effectively

storing the list-entry format on the OSD, allowing the command to concisely specify the attribute reference page and any associated values.

The definition of reference pages is kept as it is in T10 spec except that the commands used to create and delete them are changed. The following commands will replace CREATE ATTRIBUTES PAGE and REMOVE ATTRIBUTES PAGE commands that are currently used for creating/removing reference pages:

- CREATE REFERENCE PAGE
- REMOVE REFERENCE PAGE

9.2 Temporary Objects and Garbage Collection

Applications might benefit from having OSD to perform garbage collection on objects not properly recorded or not recorded at all. To distinguish a temporary object from a permanent one, a new attribute called "isCommitted" might be introduced. The default value of this attribute is "yes" (i.e., by default the object is permanent). A new operation called GARBAGE COLLECT is defined to perform the task. Given the partition id, this command clears all the non-committed objects in that partition.

9.3 Expiration Date

A new attribute called "expiration date" can be defined for objects to limit the lifetime of an object. Those objects with an expired lifetime would be inaccessible and an explicit command would remove them from the device permanently.

9.4 Secure Erase

Secure erase can be implemented as an attribute of the object or attribute of the command (via current command attributes page).

10 References

[1] SCSI Object-Based Storage Device Commands (OSD), T10 Working Draft, Revision 07, 10 June 2003.
[2] ObS-Grouping, G. Ericsson and M. Zelikov, Version 0.3, May 2003.
[3] ObS-Identifying Objects, J. Satran et all. Version 0.7, June 2003.
[4] ObS-Collections, G. Ericsson et all, Version 0.3, July 16, 2003.