

Supersizing your data?

OAM's the place with the space

BY KEVIN GOLDSMITH

That's right, now you can super-size your data in Object Access Method (OAM). Previously, the maximum object size was 256 MB, but now OAM can manage objects up to 2000 MB!

Overview

OAM is part of DFSMSdfp™, a base element of z/OS. Technically, OAM is an access method, but it is also much more. Do you:

- Need a place to store your data for access at some later time
- And have it automatically managed through a life cycle including creating multiple backups on removable media,
- And have it moved up and down a storage hierarchy that you create
- And the storage hierarchy can consist of any combination of disk, optical, and tape destinations depending upon your access requirements,
- And at the end of the life cycle have the data expired and automatically deleted?

It's a mouthful, but this is what OAM can do for you. An object is an unstructured data string and OAM does not care what it contains—it can be an x-ray, Adobe PDF, scanned image, your entire mp3 collection—whatever content you would like. OAM maintains metadata about each object in an IBM DB2 table that provides an object directory.

With the OSREQ application programming interface (API), you can use OAM to store objects, retrieve objects, query the metadata about objects, change the way the object is managed, or manually delete the object. You can write your own application program or use one of several solutions that already use the OSREQ API, including IBM DB2 Content Manager and IBM DB2 Content Manager OnDemand.

OAM is very flexible and exploits the power in the SMS storage group, storage

class and management class constructs, and associated ACS routines to conveniently manage objects throughout their life cycle according to the policy you have established based on your unique requirements.

About larger object sizes

There are several different ways that OAM applications provide a data entity to OAM:

- The data entity is provided in its entirety as a single object to OAM. For example, consider an architectural drawing that has been scanned in black and white and results in a file that is 250 MB in size.
- Multiple small data entities are bundled together and this conglomeration is provided as a single object to OAM. For example, consider check images, each about 4 KB in size, so roughly 65,536 4 KB check images are in one 256 MB object. The application might need to track where each individual entity resides within the OAM object and some applications store locator metadata in the object itself.
- The data entity is larger than the maximum OAM object size and must be separated into multiple OAM objects. For example, consider a 1000 MB digital x-ray image that is divided into four separate 256 MB objects. In this case, the application might need to track the individual OAM objects to later reconstitute the data entity from multiple OAM objects.

The amount of data in the world is increasing at a staggering pace. Your cell phone bill can enumerate every call you made and received, new medical procedures can provide detailed digital



imaging of your inner workings, and social networking sites can capture all the aspects of your life in words and pictures. OAM is ready to handle this growth in the amount and size of data entities.

Let's see how the new 2000 MB object size helps in each of the following cases discussed in [About larger object sizes](#):

- When a data entity is provided in its entirety, the size of the object can be increased several fold. That scan of the architectural drawing might have been constrained by the old maximum object size of 256 MB, limiting it to a black and white rendition, and now perhaps that same architectural drawing can be scanned in full color to more accurately preserve its original content.
- When multiple data entities are bundled together, now many more data entities can be included in each bundle, greatly reducing the number of OAM objects that are needed. With our 4 KB check image example, the number that can be bundled in a single 2000 MB object is more than half a million!
- For a large data entity, rather than having to separate it, it might now fit in its entirety within a single OAM object. If it must be separated, it will result in much fewer OAM objects. That 1000 MB digital x-ray image would now easily fit in a single OAM object.

Configuration and usage

Now that you know how using larger object sizes can make life easier, there are a few steps that you must take to configure OAM before you can use it.

OAM configuration

Figure 1 illustrates how you can configure OAM to use object sizes up to 2000 MB.

```

OAM Configuration

IEFSSNxx member of PARMLIB - OAM subsystem level configuration
:
:
SUBSYS SUBNAME(OAM1) INTRTN(CBRINT) INITPARM((TIME=GMT)[,MSG=x][,OTIS=x][,UPD=x][,MOS=nnnn][,LOB=x][,QB=x][,DP=x])
:
:

OAM member of PROCLIB - OAM address space configuration

//OAM PROC OSMC=YES,MAXS=2,UNLOAD=9999,OAM=xx,EJECT=LRW,REST=YES
//IEFPROC EXEC PGM=CBROAM,REGION=0M,step 3
//PARM=( 'OSMC=&OSMC,APLAN=CBROAM,MAXS=&MAXS,UNLOAD=&UNLOAD',
// 'OAM=&OAM,EJECT=&EJECT,RESTART=&REST')
//SYSABEND DD SYSOUT=A

```

Figure 1. OAM configuration

For step 1, specify the maximum object size on the SUBSYS statement for the OAM1 subsystem in the IEFSSNxx member of parmlib. The SUBSYS statement allows the initialization parameters to be specified for the OAM1 subsystem using the INITPARM keyword. For the maximum object size, you must specify a value for MOS=xxxx where xxxx is the number of megabytes for the largest object that can be stored in your implementation (up to 2000).

For step 2, for each object storage group in which objects greater than 256 MB will be used in the disk level of the OAM storage hierarchy, specify the DB2 tables for object storage. In z/OS V1R8 and later, you can optionally use OAM DB2 LOB tables to store objects, but if you are going to use objects greater than 256 MB, you must use DB2 LOB tables. You'll need to review your DB2 configuration to ensure that the LOB tables have been defined and OAM has been configured to use the LOB tables (including LOB=x on the SUBSYS statement).

Step 3 is required if objects greater than 256 MB will be used at the tape level of the OAM storage hierarchy. Processing of objects at the tape level of the OAM storage hierarchy is performed in the OAM address space using 64-bit addressable virtual storage so you must specify a MEMLIMIT value to provide adequate virtual storage above the 2 GB bar to the OAM address space. You can use several mechanisms for providing a MEMLIMIT value, but the easiest is to specify REGION=0M on the OAM procedure CBRAPROC. This is the default.

The following example for an object stored to disk illustrates an application address space that receives data from a client over a network. As each part of the overall object arrives in the application address space, it is provided on a STOREPRT invocation until the entire object has been received. Application processing is similar for an object stored to tape.

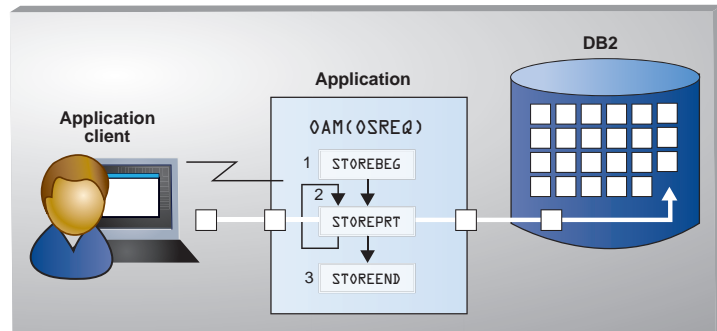


Figure 2. Application store of a large object in parts

One more related item, however, that simply cannot be overlooked is that because of the increased use of virtual storage, a robust auxiliary paging subsystem must be in place backed by DASD and sufficient real memory.

OSREQ application programming interface

When you have defined your configuration for objects up to 2000 MB in size for use, it is time to get busy on the application programming interface. One of the challenges of storing an object up to 2000 MB in size is the virtual storage demands in the application address space. Not to worry though because the OSREQ API now provides store sequence functions named STOREBEG, STOREPRT, and STOREEND that accommodate the application programming environment. For objects greater than 256 MB, the store sequence functions allow an application program to provide the object to OAM in “parts”. The store sequence is initiated with a STOREBEG invocation to begin the sequence, followed by one or more STOREPRT invocations to provide each part of the object followed by a STOREEND invocation to complete the storage of the object.

With this mechanism, the application address space never has to materialize the entire object at one time.

Availability

Now that you are ready to store even more, here's the fine print:

- In z/OS V1R10, OAM supports objects up to 2000 MB only at the disk level of the OAM storage hierarchy. This means no backups or transitions are possible for objects greater than 256 MB.
- In z/OS V1R11, OAM added support for objects up to 2000 MB at the tape level of the OAM storage hierarchy, facilitating full support including backups and transition for objects greater than 256 MB.
- Objects greater than 256 MB are not supported at the optical level of the OAM storage hierarchy.

Resources

- For details on specifying a MEMLIMIT value, see *z/OS MVS Extended Addressability*, SA22-7614.
- For the OAM configuration steps to enable objects up to 2000 MB in size, see *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426.
- For details on the OSREQ API, see *z/OS Object Access Method Application Programmer's Reference*, SC35-0425. ■