Hitting the Right Note: Tuning the SMS Run-Time Options

BY STEVE PRYOR

Last month, I discussed the SMS control datasets and SYS1.PARMLIB members that must exist in order for the storage management subsystem to control the creation and movement of data in a system-managed environment. This month, I'll examine in more detail IGDSMSxx, the member that contains the runtime options for SMS.

Like other members of SYS1.PARMLIB, IGDSMSxx consists of 80-character records that specify keyword parameters that control the operation of the subsystem. The values in IGDSMSxx are used when the SMS subsystem is started at IPL time or when the SET SMS=xx console command is used (SET is usually abbreviated as "T" to avoid confusion with the SETSMS command). The T SMS=xx command is used to specify a particular member of SYS1.PARMLIB for SMS operation. In contrast, the SETSMS command is used to override the settings of individual SMS run-time options that were specified in the IGDSMSxx member. SETSMS is also used to save the current contents of the SMS control datasets or to activate new control datasets.

The only required parameters of IGDSMSxx, which must be the same across all MVS systems in the SMS complex, are the ACDS and COMMDS names. As we saw last month, if the SETSMS command is used to change the name of the currently active ACDS or COMMDS, then the new control dataset names are stored in the COMMDS. These dataset names are then used instead of the values in IGDSMSxx when the system is started again. This is done to ensure that only the most recent copy of the ACDS and COMMDS are in use, and that all systems use the same configuration. It can be very confusing, however, when the system is started and an apparently "old" ACDS and COMMDS are used.

A number of parameters are related to RACF processing. Setting ACSDEFAULTS(YES) allows SMS to make an extra call to RACF for each dataset processed in order to determine the dataset's application identifier and the SMS class names specified in the RACF user or group profiles. The result of the RACF call is used to set the values of the &APPLIC, &DEF_DATACLAS, &DEF_STORCLAS, and &DEF_MGMTCLAS variables that can then be used in the ACS routines to control placement of the dataset. The REVERIFY parameter indicates when SMS is to check if the storage and management classes actually assigned to the dataset are valid for the user – either at job initiation time or at both job initiation time and at the time the dataset is allocated. The user ID that is used in these checks may be either that of the user allocating the dataset or that of the owner of the dataset, depending upon the setting of the USE_RESOWNER parameter. The default is YES; that is, the resource owner of the dataset is used for authorization checking.

The other run-time options that may be set in IGDSMSxx or changed via the SETSMS command fall into several categories, including, for example, OAM keywords. In prior releases of DFSMS, the Object Access Method (OAM) keywords DB2SSID, OAMPROC, and OAMTASK were included in the IEFSSNxx member of SYS1.PARMLIB. Beginning with DFSMS version 1.1, these keywords have been moved to IGDSMSxx. This actually solved a problem, documented by APAR OY53388, in which the length of the SMS record in IEFSSNxx could exceed the 80-byte limit if all of the optional parameters were coded. Several other changes to the OAM keywords for DFSMS version 1.1 are documented in the informational APAR II07429.

ADDITIONAL RUN-TIME OPTIONS

Other run-time options are of interest to the storage administrator because they affect allocation performance. The additional RACF call that is performed when ACSDEFAULTS(YES) is specified adds slightly to allocation overhead. A more important source of performance problems is the TRACE parameter. If the tracing facility is active, each SMS module writes a trace record to an in-storage buffer as it is entered. The default is to trace all SMS functions for error events for all jobs (i.e., SELECT(ALL) TRACE(ERROR) JOBNAME(*)). Tracing only ERROR events limits the amount of overhead, as compared to TRACE(ALL), the use of which can seriously degrade performance. However, unless problems are being experienced for which trace records might prove useful, it is probably better to eliminate the additional overhead altogether by specifying TRACE(OFF). The current state of the TRACE option can be determined by issuing the DISPLAY SMS, TRACE command at the operator console. Additional information on all of the options that can be set in IGDSMSxx can be displayed via the DISPLAY SMS, OPTIONS command.

Other performance and reliability-related parameters can also be found in IGDSMSxx. The INTERVAL parameter specifies the time period that must elapse before the COMMDS is refreshed with any changes to the SMS configuration and with new statistics for the volumes in the system. The DINTERVAL parameter specifies the length of time between reads of device statistics from the 3990 control unit. The DEADLOCK_DETECTION interval determines the time in seconds that the system uses for finding deadlocks within and across systems, and it is used with VSAM record-level sharing introduced in DFSMS version 1.3.

Since PDSEs must reside on SMS-managed volumes, IGDSMSxx contains, as one might expect, a number of parameters that regulate the use of PDSEs. These include the DSNTYPE, PDSESHARING, and HSPSIZE parameters. Specifying DSNTYPE(LIBRARY) allows PDSEs to be created as a default whenever a dataset is allocated with directory space specified instead of an ordinary partitioned dataset. PDSESHARING may be either NORMAL or EXTENDED. NORMAL sharing allows PDSEs to be shared across systems at the dataset level. EXTENDED sharing uses XCF coupling to allow the sharing of individual PDSE members across systems. EXTENDED sharing requires the use of a cross-system enqueue facility that can pass the SYSZIGW0 and SYSZIGW1 queue names across systems. HSPSIZE specifies the number of megabytes that are set aside for hiperspace caching of PDSE members.

NEW RUN-TIME OPTIONS

As DFSMS matures, its capabilities continue to be extended. Each new DFSMS release introduces new run-time options into IGDSMSxx to support the enhancements. DFSMS version 1.3 includes VSAM record-level sharing, in which SMS-managed VSAM datasets can be shared across systems and concurrent updates made to individual records without loss of integrity. This involves a new address space, the SMSVSAM address space, and several related parameters in IGDSMSxx. One such parameter, RLSINIT, is used to start the SMSVSAM address space at IPL. DFSMS version 2.4 (included in OS/390 version 2.4) adds "tailored" compression in which the compression dictionary for each individual dataset is unique to the dataset and is stored with it. This facility is controlled by the COMPRESS-(TAILORED/GENERIC) parameter.

Another option of concern to the storage administrator is the OVRD_EXPDT parameter. When set to YES, this option allows SMS-managed DASD datasets to be deleted even if their expiration date has not yet passed. This option is designed to be used with the Tape Mount Management (TMM) methodology, which uses the ACS routines to eliminate tape mounts by redirecting small tape datasets to a DASD buffer. Since the JCL for tape datasets often includes an expiration date, the DASD buffer could not be periodically swept clean if many of the datasets contained an expiration date.

UNDERSTANDING OPTIONS KEY TO EFFICIENT USE OF RESOURCES

Storage management in the modern data processing environment is a complex task that requires an understanding of how the storage management subsystem automates the use of DASD resources. In a DFSMS environment, this means that the storage administrator must understand the control structures that determine how the SMS subsystem creates and manages datasets. Understanding the SMS run-time options specified in the IGDSMSxx member is an important key to ensuring that expensive system resources are used as efficiently and effectively as possible. ts

NaSPA member Steve Pryor has more than 15 years of experience in storage management, disaster recovery, software development, and technical support. Steve can be contacted via the Internet at pryor@atlanta.com.