IAM: Improving Performance for VSAM Applications

Time. It’s all about time. How to get things done quicker. IAM can be of significant benefit in getting your VSAM processing work completed faster. Not just on time, but frequently ahead of schedule. IAM has been helping businesses in a variety of industries do just that for over 40 years, and is doing it even better today. IAM is a proven, reliable, and high performance alternative to VSAM. No programming changes are required. Generally no JCL changes other than an operand in the VSAM define process to indicate that IAM will handle the data set. IAM is being used daily to handle an incredible amount of I/O activity from online systems, batch systems, using in-house written software as well as software products provided by other vendors.

This presentation will provide insight as to how IAM is able to accomplish this feat, along with some user results that demonstrate IAM’s time saving results.

Click here for SLIDE only view.
IAM: Improving Performance for VSAM Applications

To preview, there will be an explanation of what the IAM product is, then continue with an overview of the IAM files structure followed by a discussion that explains why one would want to use the IAM product. While IAM offers a variety of functions and features not found in the VSAM access method, the primary benefit of IAM is the performance that it provides which exceeds the performance of VSAM, in terms of CPU time, elapsed time, and reduced physical I/O activity. These benefits occur because of the intentional design and programming techniques that are used to achieve these results. We’ll explore the strategies that IAM uses to achieve this outstanding performance, and provide examples from actual user experiences with IAM.

We’ll wrap up with showing how easy it is to implement IAM, discuss some of the IAM advanced functions, and have a quick review of the enhancements in the current release, IAM Version 9.2.
IAM Evolution

IAM continues to be an evolving product. IAM has a team of experienced developers dedicated to enhancing and improving the IAM product. This includes making use of new technologies, hardware and software, as applicable to the IAM product. Thanks to our partnership with IBM, we are able to test the IAM product and make any necessary revisions prior to general availability of new operating system releases, new releases of CICS, even recently with the new COBOL z/OS Version 5.

Since 2010, with the IAM V9.x releases, IAM has added support for these technologies:

- **EAV** Super large volumes with >64K Cylinders
- **XTIOT** TIOT entries for IAM files above the line, enables more datasets to be allocated to an address space
- **31-bit DSAB** Reduces 24-bit virtual storage use
- **64-bit virtual storage** For the index, enables more files to be opened and larger files
- **Sysplex XCF** Enables data set sharing across a sysplex
- **System Logger** Used by IAM/RLS and IAM/PLEX for journaling
- **Uncaptured UCB’s** Reduces 24-bit virtual storage use
- **Hiper-dispatch** Redesign of IAM/RLS and IAM/PLEX work flow handling to efficiently work with hiper-dispatch.
- **z/HPF support** Enables reduced channel utilization for increased capacity
- **64-bit virtual storage** Buffers in 64-bit virtual alleviate above the line storage constraints and enable use of more buffers

Among the potential future enhancements for IAM, Innovation is looking at the following:

1. Utilization of SYSPLEX Coupling Facility for IAM/PLEX
2. Support for IBM GDPS Active-Active function
3. PE Block reuse

The list above is intended to provide insight as to enhancements that are under serious consideration at this time.
VSAM API

Application programs using VSAM have no required program changes to use IAM files for the functions supported by IAM. Presuming IAM is in the link list, in general there are no JCL changes required for application programs that are using IAM files. There is no need to remove VSAM specific parameters, such as those specified by the AMP= parameter, or use of Batch LSR. Parameters not relevant to IAM will be ignored. Types of JCL changes that might be needed include changes for the REGION or MEMLIMIT parameters. Some JCL changes may be recommended for application programs such as an IAMOVRID DD card with associated control cards, or an IAMINFO DD card to obtain the IAM report on file activity.

Functions not supported by IAM's VSAM API are:

- KSDS – No CI or RBA processing
- ESDS – No updating by CI
- RRDS – CI access limited to files with 1 record per CI

IDCAMS Functions Supported by IAM include:

- Delete
- Define
- Bldindex
- Repro
- Print
- Listcat
- Verify
- Alter Newname

Some IAM file attributes can be altered via Define Recatalog. (See Section 10.84 of IAM Manual)
Fixed Length Blocks

IAM uses fixed length data blocks to store data records. IAM is different from VSAM in that the data blocks do not have a limited set of block sizes (CI sizes) that can be used, rather the blocks can be any size. IAM will automatically select a block size that will maximize the amount of data that can be stored on each track based on the device architecture. A typical IAM file will have 4 blocks of the maximum size per track, which on 3390 type device is 13,682 bytes. Different block sizes can be used, such as 2 blocks per track at 27,968 bytes.

Non-VSAM Data Set

An IAM data set resides on DASD as a non-VSAM sequential (DSORG=PS) type of data set. IAM data sets do not have separate data components and index components, all the data, index, and other information about the data set reside in a single data set on DASD that is in the catalog and VTOC using the cluster name.

Use of DSNTYPE=LARGE is recommended for best performance with least restrictions. This can be set as the default in the IAM Global Options with ENABLE=LARGE.

SMS Extended Format data sets are supported by IAM but are not recommended. Issues are performance and potential increase in DASD space requirements with this format. Note that block sizes used for SMS Extended Format are 32-bytes smaller due to the 32-byte suffix added to each block.

EAV’s

Extended Address Volumes are DASD volumes that can have greater than 64K cylinders. IAM data sets can reside on these super large volumes. All IAM datasets can be made eligible for EAV’s by setting the IAM Global Option ENABLE=EAV. This specification does not require the data set to reside on an EAV volume, they just make them eligible to reside on an EAV volume.

If an installation wants to limit EAV usage to only specific data sets, then EAV can be specified by the DEFINE CLUSTER keyword of EATTR(OPT). Alternatively, they can be placed in a Data Class that has the EATTR value set to OPT.
IAM Concepts

- IAM File Structure

<table>
<thead>
<tr>
<th>Control Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prime Data Area</td>
</tr>
<tr>
<td>Index</td>
</tr>
<tr>
<td>Extended Area</td>
</tr>
</tbody>
</table>

IAM File Structure

The IAM file structure is contained within a non-VSAM DSORG=PS type of data set (including DSNTYPE=LARGE and SMS Extended format) with four distinct areas. The first couple of blocks in the data set contain the information that VSAM keeps in the catalog about the data set. That includes maximum record size, key length, key offset, whether the index and / or data areas are compressed. They also contain the information about where the other areas are in the physical data set, such as the data area blocks, the index, and the Extended Area. The location information is not device dependent, allowing the data set to be freely moved to other volumes or even device types.

The Prime Data Area contains the data records that were written when the data set was last loaded, plus any imbedded free space based on the amount of control interval free space when the file was defined. The size of the prime area is fixed after the file is loaded or reorganized.

The Index Area contains the Prime Index, that is the index to the prime data blocks. For most files there is a compressed format of the index that is used to reduce the amount of storage that the index requires. This index is read into virtual storage when the data set is opened. If the data set is using hardware compression, the compression dictionary is appended to the end of the index blocks.

The Extended Area contains the Overflow blocks, the PE (Prime Extension) blocks, and a few “index” or “directory” blocks describing the content of the subsequent blocks. The Overflow blocks contain records that have been inserted or updated and could not fit in the appropriate block in the Prime Data Area. The PE blocks contain records that were appended to the logical end of the file. The Overflow and PE blocks are intermixed within the Extended Area and are obtained as needed. The Extended Area is expandable by obtaining additional extents when needed, and can expand across multiple volumes.
IAM Performance Strategies

- Index in Virtual Storage
- Real Time Tuning Buffer Management
- Insert Strategy
  - Record Based Overflow
  - Prime Related Overflow
  - Prime Extension
- Data Compression
- Dynamic Tabling
- Overflow Caching

IAM provides performance by design. Beyond just utilizing efficient coding techniques, the design includes the implementation of key strategies to achieve a high level of performance. We'll explore the key strategies that IAM employs to achieve high level of performance. These strategies include having the index in virtual storage, Real Time Tuning, record insert strategies, data compression, and data in virtual capabilities.
Index in Virtual Storage

When an IAM data set is opened, IAM reads the index of the file into an area of virtual storage associated with the address space that has opened the file. Based on the IAM Global Options and storage availability, the virtual storage can be in 64-bit virtual storage, a z/OS data space, or in above the line storage in the callers address space. The area selected will contain the entire index for the file. The index is typically kept in a compressed format to reduce the amount of storage required. The key benefit is the elimination of the I/O VSAM performs to read the index as the file is being used. For VSAM applications this reduces that amount of physical I/O that is necessary to process the file with VSAM.

When the index is placed in a data space or 64-bit virtual, the IAMINFO report contains the amount of that storage used for this particular data set.

For IAM files that have a million or more records in overflow, the PRO (Prime Related Overflow) structure will significantly reduce the overflow index storage size. This feature will be discussed in more detail in subsequent pages.
Real Time Tuning

Real Time Tuning is a concept that IAM uses to dynamically manage the buffering for IAM files in response to the processing requirements. This capability allows IAM to adjust the number of buffers in real time being used for any particular file, and dynamically select appropriate buffer management and I/O techniques. This innovative unique buffering technique has proven itself to provide extraordinary improvements in elapsed time to process batch jobs and reductions in online transaction response times. The strategies of IAM Real Time Tuning include:

- Least recently used (or referenced) management of buffers used for randomly accessed data to keep the most frequently referenced blocks in storage. This is automatic with IAM, requires use of LSR for VSAM.
- Automatic deferred writes for batch updates done with Share Option 1 or 2 files, must be requested for VSAM.
- Immediate reuse of buffers with sequentially processed blocks when reaching the end of the data block.
- Anticipatory buffering for long streams of sequential I/O requests
- Read and write multiple blocks per physical I/O, up to one cylinder worth of blocks.
- Dynamically balance buffer utilization even in mixed random and sequential environments such as can occur with ACCESS IS DYNAMIC applications. This is not possible with VSAM without closing and opening file.
IAM Performance Strategies

- Real Time Tuning (continued)
  - Dynamically adjusts number of buffers
    - Works within a range of minimum/maximum number of buffers
    - Periodically evaluates buffer usage and adjusts as necessary
    - Provides indication if larger maximum would reduce I/O
    - Maximum buffer defaults (installation modifiable)
      - 65,536K (64M) buffer space for Batch/TSO
      - 1,024K (1M) buffer space for CICS
      - Defaults should yield excellent performance for most datasets
    - Increase maximum by using BUFND or BUFSP
    - Can use IAM Override facility to override buffering value
      - BUFSP, MINBUFNO, MAXBUNO
  - Turbo mode increases responsiveness

Real Time Tuning – 2

IAM Real Time Tuning works with a variable number of buffers within a minimum and maximum range. IAM acquires buffer storage when it decides more buffers would be beneficial, and releases it when the buffers are no longer needed. IAM mode, will acquire buffers as a need is observed, which is done very quickly in TURBO mode. Reductions in the number of buffers is done less frequently to minimize storage management overhead.

IAM generally starts out during open processing with sufficient buffers to read the index with as few EXCP’s as possible, but no more than the number of blocks in a cylinder plus a track. For small files, IAM will not get more buffers than there are data blocks in a file. Under CICS, the number of buffers obtained during OPEN processing will be reduced to a minimal amount after the index has been read into storage to avoid excessive buffer allocations for files that may have very low activity.

There is a default buffer space value when the file is opened by a batch job or TSO user, and a smaller buffer space default for CICS regions due to usual memory constraints under CICS.

Default values can be changed through JCL or ACB specification of BUFND or BUFSP. There are also IAM Overrides that can be used to alter the values.

The TURBO mode is invoked at initial OPEN and during periods of heavy physical I/O in short time periods to accelerate buffer acquisition to meet the processing needs. When TURBO mode is not enabled, buffer acquisition occurs over a longer period of time if needed.
IAM Performance Strategies

- **Real Time Tuning (continued)**
  - Uses 31-bit virtual storage for all buffers
    - If a buffer is acquired in 24-bit storage, it will be released
  - Option to use 64-bit virtual for buffers
  - Does not connect buffers to place holders (strings)
    - Eliminates CI lockout-exclusive control problems
  - **Simplified Manual Tuning**
    - IAM will display the IAM368 message in the IAMINFO report for a dataset if it detects that more buffers would have been beneficial
  - **Bottom line**
    - Typical results are a 30% to 80% reduction in elapsed time

**Real Time Tuning – 3**

IAM will never give an application program the address of an actual data buffer. The advantages of this are that IAM can use buffers in storage areas that the application program may not have addressability to. It also prevents inadvertent corruption of the data that is in buffers. By default, IAM buffers are always in above the line (31-bit addressable) storage areas. If a buffer is acquired in 24-bit storage, unless it is the only buffer IAM has, it will be released. To prevent potential for storage fragmentation, the storage acquired for a buffer is always rounded up to a page size boundary.

IAM can also utilize above the bar (64-bit virtual) storage for buffers, either optionally or can be set as the default. The 64-bit virtual buffers can be used even when the application program is not running in an AMODE of 64. Due to z/OS managing 64-bit virtual in megabyte size portions. IAM will acquire a megabyte at a time and divide that up into whatever number of buffers will fit into a megabyte for that particular data set. So IAM’s management of the buffers is handled at the megabyte level rather than the individual buffer level.

Contention is reduced when using IAM instead of VSAM because IAM does not create a connection between a buffer and the place holder (VSAM string). This eliminates the CI lockout problem with IAM. Users no longer need to use small control interval sizes to minimize this problem. CICS users can set the SIT parameter CILOCK=YES when using predominately IAM files to avoid additional under the covers overhead added by CICS to avoid the CI lockout problem.

With IAM, there is no need for spending long hours doing I/O performance analysis of your VSAM file to obtain optimal performance. Through the IAMINFO report, IAM will inform users when more buffers could have helped reduce response time through the display of the IAM368 message. In fact when using the IAMSMF program to printout the IAMINFO report, with the ATTRIBUTE=MOREBUFFER keyword, IAMSMF will only printout the IAMINFO reports that indicate that more buffers will be beneficial.
How can IAM be so much better than VSAM?

As we stated in the beginning it is all about time. We recently received an email from a user asking how can IAM be so much faster than VSAM. They had an application that was using VSAM to process a particular data set, and had a job that was running for over two hours. One of the programmers knew they had the IAM product and decided to try that process with the file as an IAM file. The programmer was so amazed when the job ran in 13 minutes, that they ran it again just to make sure and it ran in 10 minutes the second time! This was accomplished with no changes to the application, just converting the file from real VSAM to IAM. The IAM file ran with all of the IAM default values.

Our analysis proceeded as follows. We asked the user to provide the IAMINFO report from the job so that the statistical information on how the file was being accessed could be analyzed. The IAMINFO report is automatically produced by IAM to SYSOUT and / or written out as an SMF record. The IAMINFO report contains information about the file being accessed and run time statistics on the different type of logical I/O requests processed, the physical I/O, virtual storage usage, and information on the buffering. The user was able to retrieve the record from the SMF data.

From the IAMINFO report, we determined that ACCESS IS DYNAMIC was being used to perform random reads. What happens under the covers is that a random read request with ACCESS IS DYNAMIC as implemented by LE as a POINT followed by a sequential GET. There were 2 million POINT’s being issued, and there were some 4 million sequential GET’s issued. About half of the GET’s are being done for random reads. The other half could be coming from either:

1. Following the random read with sequentially reading some of the following records, or
2. There could be a phase of the program where it does a sequential read of the entire file.

If it was #1, you can get some improvement with VSAM by using some form of LSR buffering.

If it was #2 which seems more likely based on the stats, then the improvement you can get with VSAM will not be as much as for case #1. The reason is that with VSAM, the job can use either NSR (non-shared resource) buffering which with a good number of buffers will help the sequential side or LSR buffering which is better for the random side. Even with SMS System Managed Buffering (SMB), SMB decides what buffering technique will be used when the dataset is opened based on the ACB MACRF or the ACCBIAS parameter. With ACCESS IS DYNAMIC, SMB will use NSR buffering unless user specifies differently.
There are various strategies that IAM utilizes to enhance VSAM application performance. The first key strategy that helped improve performance for this application is that IAM reads the index into virtual storage in a compressed format during open processing to avoid any index component I/O while processing the data set.

The second strategy was with IAM’s innovative Real Time Tuning process. This process automatically adjusts the buffer management algorithm and the number of buffers used based on what it sees the application program doing in real time. IAM can adjust buffer management and the number of buffers very quickly if a program changes from predominately sequential to predominately random or the other way around, or adjust buffer management to handle the mix of random and sequential. This enables IAM to obtain very significant I/O and elapsed time savings, particularly when an application is using ACCESS IS DYNAMIC type of processing.

The third strategy that IAM used to reduce I/O in this circumstance is that IAM defers writing out updated data blocks until the buffer is needed for another block. IAM will automatically defer writes for batch programs on files defined with Share Option 1 or 2. With some half a million updates being done, depending on how they are being done, the job with real VSAM may be doing significantly more write I/O than IAM is doing. With VSAM for other than sequential updates with NSR, a user has to request deferred writes, it is not automatic.
IAM Performance Strategies

- **Real Time Tuning: File Load Buffering**
  - Sequential output process
  - Defaults to obtaining enough buffers for 2 cylinders of data
  - Uses 31-bit or 64-bit virtual storage for buffers
  - Controlled by CRBUFOPT Override or Global Option
  - When 1/2 buffers are filled, issues EXCP to write that set of buffers
  - Application can concurrently fill up rest of buffers
  - Uses Data Space to hold index while writing data
  - *Note: For SMS Extended Format datasets BSAM is used, so IAM does not have direct control on number blocks written per physical I/O*

**Real Time Tuning – File Load Buffering**

A file load is a sequential output process, so IAM automatically sets up buffering to optimize the I/O performance for that process. IAM acquires two sets of buffers, with each set by default being the amount of blocks that will fit on a cylinder. When the first set is full, IAM writes out in one EXCP that set of buffers, and enables the application to continue processing with the second set of buffers. Processing continues alternating between the 2 sets of buffers with one set being physically written while the other set is being filled with user data. While that process is going on, IAM keeps the high key for each data block in a data space. When the CLOSE Is issued, IAM will then create and write out the index using the data from the data space.

The IAM CRBUFOPT Global Option and override keyword is used to control the amount of buffers that will be utilized for the file load process. There generally is no need to change this value, except if the use of 64-bit virtual for the buffers is desired. That would be specified by CRBUFOPT=64BIT. Note that 64-bit buffers do increase CPU time due to z/OS overhead of page fixing 64-bit virtual storage.

For IAM files that are SMS Extended Format data sets, IAM uses BSAM to load the file instead of EXCP due to the IBM restrictions on that type of data set. This limits the control on the number of blocks written per physical I/O, and can result in less than optimal performance. Innovation recommends the use of DSNTYPE=LARGE in place of SMS Extended Format data sets to avoid the potential performance issue.
High Performance FICON: z/HPF

High performance FICON is a combination of an architecture, hardware, and software that result in more efficient utilization of FICON channels that increases the channel capacity for transferring data. While it initially had very limited functionality, over the years IBM has enhanced the architecture with increased functionality to the extent where z/HPF can now handle a set of functions used by channel programs executed using EXCP. The callers of EXCP processing are required to change the supplied channel program to a z/HPF format, which includes a TCW and its associated data areas. The architecture is still limited to a specific set of functions, and requires a very exact set of specifications that are different than with the CCW format.

From testing with IAM, we’ve seen an average of 26% reduction in connect time, based on SMF data. The elapsed time savings varies from run to run, we’ve seen up to a 22% reduction in elapsed time with z/HPF, however typically it is somewhat less, in the 5% to 10% range.

- The processor hardware requirements are for a z/196 GA2, z/114 or EC12 processor or higher.
- The software requirements are z/OS 2.1, or z/OS 1.12 and z/OS 1.13 with APAR OA38185, and IAM Version 9.2.
- Dasd Storage:
  - **IBM:** DS8700 or DS8800 at 6.2 level and above
  - **HDS:** Hitachi Virtual Storage Platform G1000 at microcode level 80-02-01 or higher is appropriate for IAM zHPF support
  - **EMC:** Customers should contact their local EMC representative regarding information on an appropriate VMAX microcode level for IAM zHPF support.

IAM Version 9.2 will default to using z/HPF when all of the requirements are met.
IAM Insert Strategies

IAM has two alternative insert strategies. The original record based overflow area still provides the best performance for most IAM files. The alternative insert strategy called PRO (Prime Related Overflow) is primarily for those few files that have a tremendous quantity of insert activity where the overflow area may contain approximately a million or more records. The Record Based Overflow will be discussed on this slide, and PRO format on the next slide.

Record Based Overflow

When a record is being inserted into an IAM data set, and there is insufficient space for it in the block that the index indicates the record should logically be in, then that record is placed in the extended overflow area. The new or changed record can be placed anywhere within the overflow area that has sufficient space for it to fit. If there is no space available, a new overflow block will be assigned and the record placed in the new block. Additional extents will be taken as needed to acquire additional overflow blocks. The key of the record is placed in the overflow index in virtual storage in a compressed format along with the block identifier of the block containing the record.

Records that are deleted from the extended overflow area have that space made immediately available for reuse along with any other unused space in the overflow block for any other record regardless of the key value.

This strategy has consistently provided better performance than the VSAM CI and CA split strategy. It uses less I/O, less CPU time, and less DASD space than VSAM, and works exceptionally well for the vast majority of data sets. This IAM insert processing is ideal for VSAM files that have a lot of insert activity.

In very rare circumstances with exceptionally high insert rates there can be some performance issues which are easily resolved by the Prime Related Overflow structure.
Prime Related Overflow (PRO)

Prime Related Overflow Insert is designed to be used for those files that do not perform well with the record based overflow area. In general it is recommended for use for data sets that have approximately one million or more records in the extended overflow area of the data set, although it can be used for any IAM data set. The advantages are reduced virtual storage for the index, reduced data set reorganization requirements, decreased open time, and potentially better sequential read performance than the record based overflow.

Insert Strategy

The insert strategy is based on key value utilizing a block split technique, which is conceptually similar to a VSAM CI split. All records in the overflow block are related to the same Prime Data Block such that the overflow index only needs to contain one key per block rather than the one key per record used by the standard overflow area. This reduces virtual storage used by the overflow index. The overflow block key is included in the entry for the overflow block in the extended area index / directory entry eliminating the need to read through the overflow blocks during open processing.

When a PRO extended overflow block has all of the records deleted, the block is immediately available to be reused by records associated with any prime block and any key.

Unlike VSAM, IAM files have no similar concept to a CA (Control Area), so the very expensive CA splits do not occur in an IAM file.

The PRO strategy is not recommended as the default for two reasons. One, because it may use more disk space than the record based overflow strategy, and two, the PRO file structure is not as sharable as the record based overflow structure. Note that for any IAM file, full data integrity when sharing is only available with IAM/RLS or IAM/PLEX.
Prime Extension

The IAM Extended Prime Extension (Extended PE) area is used for records that are added to the end of the file, that is with key values higher than the previous high key in the file. The records are kept in ascending key sequence and are indexed by the high key in each block. Available extended area blocks are used as additional PE blocks when required, and will be intermixed with the extended overflow blocks. As with overflow, the Extended Area is expanded by additional DASD extents as needed. On disk, the extended PE area index is contained in the Extended index / directory blocks, and read into virtual storage when the file is opened.

The benefits of the Extended PE area over the overflow area include less index storage, good sequential performance, and reduced need for the file to be reorganized.

The disadvantage of the Extended PE area is that if records are added then subsequently deleted, the space in the extended PE blocks is not reusable. For such a circumstance, users are encouraged to force overflow use instead of PE. PE reuse is currently under consideration as a future enhancement.
Data Compression

IAM offers record level data compression on all IAM Enhanced Format files, including KSDS, ESDS, and RRDS type of files. The benefits of using data compression include increasing the effective amount of data transferred per I/O, a reduction in EXCP counts, and reduction in data set size. The disadvantage of using data compression is the increase in CPU time over non-compressed IAM files to perform the data compression and decompression. For many circumstances, IAM will still use less CPU time data compressed then VSAM uncompressed. However when your key objective in utilizing IAM is for CPU time savings, then compression should be turned off and used only for selected very large data sets.

IAM currently offers two forms of compression. The first is a high performance proprietary software algorithm that does not use a dictionary. Typical results are 20% to 50% compression. In general this form of compression uses less CPU time than hardware compression to compress the data, but slightly more CPU time than hardware to decompress the data.

IAM also offers support for the z/Architecture hardware compression instruction which does require a compression dictionary. IAM will dynamically build a dictionary as part of the file load process which is generally intended to provide compression results comparable to VSAM. Users can optionally create compression dictionaries with the IBM provided CSRBDICT REXX exec and provide those dictionaries to IAM to use. Depending on the data patterns in the data set the dictionaries created by CSRBDICT have achieved 90% or more compression for some data sets.

Users can set a global default for which compression technique, if any, they desire to use. The technique can be changed by IAM overrides for specific files as desired. The amount of compression achieved is dependent on the data patterns contained in the file. For some files software will do better, for others the hardware will do better. Users generally just pick the type of compression they desire to use, and use that technique for all of the files they convert to IAM.

z Enterprise Data Compression (zEDC) is oriented to compressing large portions of data, such as a full track of data rather than the record oriented compression that is used on smaller amounts of data. VSAM (and IAM) type of data sets tend to be a mix of record level random or record level sequential processing type of processing which is more efficiently and effectively handled using the IAM software compression or the hardware data compression with a dictionary.
IAM Performance Strategies

- IAM’s Dynamic Data Space
  - Record based cache in virtual storage
  - Used for randomly read records
  - May significantly reduce I/O and buffer needs
  - Records stored in segments, less unused storage for variable length records
  - Dynamic LRU management of records in table
  - Statistics provided in IAMINFO reports

Dynamic Data Space

The Dynamic Data Space is a feature that can be used for selected IAM data sets that can provide significant I/O savings for files processed in online environments. This feature will keep the frequently randomly read records in a data space as a type of cache for those records. When activated, IAM will check the dynamic data space to see if the record is contained in it, and if so it will return the record to the caller without any I/O. If it is not in the table, it will then read the record from the buffer if available or from DASD, and put the record in the table. The casting out of the least recently used record when another record is inserted means that all records that are currently being frequently read will stay in the table.

To determine which files may benefit from this function, examine the IAMINFO reports from the online environment for files that have a high number of GET RANDOM requests and that have a relatively very low value for the number of PUT UPDATE and ERASE requests. If any such files are found, then the Dynamic Data Space can be tried through an IAM ACCESS Override for those specific files with the keyword DYND=nnnn where nnnn is the amount of storage in megabytes that you want to use for the data space cache. You can vary the amount of storage used to determine what works best. Statistics on the use of the Dynamic Data Space in the IAMINFO report on the line DYNAMIC TABLE RETRIEVALS= 0 - DYNAMIC TABLE RECORDS= 0, which tells you the number of records retrieved from the Dynamic Table, and the number of records that were contained in the Dynamic Table.
Overflow Area Cache

For files with large record based overflow areas that are being sequentially processed, IAM has a feature that uses z/OS 64-bit virtual storage. This capability can help improve run time for sequential processing, including backups and reorganizations of Enhanced Format IAM files with a large number of records in overflow. During OPEN processing, IAM reads all of the overflow area blocks to build the record level index. With this new caching feature enabled, the overflow blocks will be copied to an above the bar virtual storage area for later reference during file access. When one of these blocks is needed for a data record, IAM will copy the data block from the above-the-bar storage area into a normal I/O buffer, rather than performing a physical I/O.

The benefits for installations with sufficient real storage are reduction in physical I/Os and elapsed times to read the file sequentially. Installations without sufficient real storage will likely find that the time spent in paging will be detrimental to job performance. To use this new feature, simply specify the IAM ACCESS Override of CACHE64. IAM will calculate the amount of above-the-bar storage required to hold the overflow blocks, and acquire it. Users may need to specify the MEMLIMIT= parameter on their execute cards to make sure that adequate above the bar storage is available to the job step, for example MEMLIMIT=4G for 4 gigabytes of above the bar storage. To determine the amount of storage needed, multiply the number of Extended Overflow Blocks from either a recent IAMINFO report or an IAMPRINT report by the block size.

Use of this function is intended for sequential batch applications and file reorganization, it is not recommended for online systems, or for jobs that are doing a lot of updates.
IAM Performance Strategies

- **Installation Selectable Defaults**
  - Buffering
  - Data Compression
  - SMF Records
  - Use of 64-bit Virtual for the Index Space
  - Use of 64-bit Virtual for buffers
  - Can be easily changed with provided program: IAMZAPOP

**Installation Selectable Defaults (IAM Global Options)**

This section identifies the IAM Global Options, which are installation selectable default values, that impact the performance of the IAM product overall.

**Buffering related Global Options:**

**BUFSP** Specifies a default, in kilobytes, for the maximum amount of storage that IAM is to use for buffers when accessing a file in environments other than CICS. IAM divides this value by the file's block size to determine the number of buffers that will fit. The default is 65,536K.

**CICSBUFSP** Specifies a default, in kilobytes, for the maximum amount of storage that IAM is to use for buffers when accessing a file under CICS. IAM divides this value by the file's block size to determine the number of buffers that will fit. For example: The default is 1024K.

**MAXBUFNO** Specifies the default maximum number of buffers IAM is permitted to acquire during file processing. The default is 5.

*Note: IAM will use the higher of either MAXBUFNO or the appropriate BUFSP value.*

**Data Compression Global Options:**

**COMPRESSTYPE** Specifies the default compression algorithm to use.

**DATACOMPRESS** Specifies the smallest size IAM file in tracks that will be considered for automatic data compression.

**SMF Global Options**

**RECTYPE** Specifies the SMF 'user' record type to be written if SMF recording is requested for IAM files.

**SMF** Specifies whether IAM is to write an IAM SMF user record when an IAM file is closed.
Index Space Use

INDEXSPACE= Specifies if IAM is to use 64-bit Virtual, a data space, or user region for the index.

Use 64-bit virtual for buffers

CRBUFOPT=64BIT Use 64-bit virtual for file load I/O
ENABLE= BUF64 Use 64-bit virtual storage for regular file access or update I/O.
Performance Summary

IAM improves VSAM Application Performance

- Index in virtual storage
  - Eliminates index component I/O and buffers
- Dynamic Real Time Tuning
  - IAM dynamically selects best buffer management technique
  - IAM dynamically decides on number of buffers
- Record based overflow
  - Eliminates I/O overhead of CI and CA splits
- Data Compression
  - Increases effective data transfer per I/O
  - Reduces EXCP counts

IAM uses multiple capabilities to provide better performance for VSAM Applications, including keeping the index of open files in virtual storage to avoid index component I/O, the Dynamic Real Time Tuning that dynamically controls buffering based on actual application needs, a Record based overflow area for faster insert processing, and Data Compression to reduce data set size and physical I/O.
User Experience

A large VSAM user was experiencing very high I/O rates to their VSAM files that was resulting in serious performance issues. They decided to try IAM to see if it could help alleviate the performance problems they were having. This and the following charts show the results the achieved just by converting these files to IAM.

Their TRMS data base was experiencing greater than 80 million I/O’s per week, and getting close to 100 million I/O’s on occasional weeks. When the converted the data base to IAM, the physical I/O activity was reduced by approximately 79%, to less than 20 million I/O’s per week!
User Experience – 2

That wasn’t the only application that benefitted. They had a Part’s Master file with greater than 100 million I/O’s per month, exceeding 120 million and even 140 million I/O’s per month occasionally. With IAM, the file’s physical I/O’s dropped to less than 40 million a month, a savings of 68%.
User Experience – 3

The same user converted the VSAM files used by the Lawson Application product to IAM. The performance results are shown on this and the following 2 charts. In this chart, we see that with IAM, their average CPU time dropped by 40.6%!
User Experience – 4

In addition to the CPU time savings, their elapsed time to run the job stream was reduced by 68.8%!
User Experience – 5

IAM also decreased their physical I/O for this VSAM application by an amazing 99%. While this is certainly an exceptional case, it is not unique. Other users have had some applications experience similar I/O savings. Certainly with some time consuming analysis effort, and trying out different tuning approaches, they could have reduced the VSAM I/O to some extent. IAM’s Real Time Tuning eliminates the need for that analysis, and along with other IAM performance enhancing features, IAM achieved results that were most likely unachievable with VSAM.
Typical Results

The results that any specific application achieves with IAM will vary, as there are many factors that can influence performance. Based on our own internal benchmarks and reports from our users, the typical results by converting VSAM files to IAM will achieve the following savings:

- Elapsed time savings for batch work of from 20% up to 60%.
- Reduction in EXCP's (Physical I/O) of 40% up to 80%.
- Reduction in CPU time of 20% to 40%. (These savings may be less if using IAM data compression).
- When data compression is used, reduction in DASD space of 20% up to 50%.
Can IAM Help?

Innovation offers at no cost an SMF data analysis program for VSAM data set activity, accompanied by our expert technical review of that data, the analysis program can provide answers to these questions:

1. Do we have enough VSAM activity to benefit from IAM?
2. What VSAM data sets will benefit the most from the IAM product?
3. How should we implement utilization of the IAM product?
SMF Analysis Program Output

Shown is an example of a portion of the SMF analysis report. The first thing we focus in on is the amount of dasd space being used by VSAM, and then the amount of EXCP’s issued. As approximately 30% of the I/O is VSAM, it does appear that there is sufficient VSAM utilization to do further analysis.

The SMF analysis program produces three different reports, including the VSAM EXCP Report which has the top 100 clusters based on the number of EXCP’s, a portion of which is shown above. The second report is a data set summary report, with additional information about each data set that was found in the output, sorted by data set name. The third report is the VSAM SIZE REPORT which has the top 100 clusters based on space allocation which can benefit from IAM by data compression which can reduce the amount of DASD space required for the data set.
How to IAM a Data Set: Initial Steps

1. Install the IAM Product
   - Refer to Section 90 of the IAM Manual for Instructions
   - Recommend placement in Link List to avoid need for STEPLIB

2. Activate the IAM VSAM Interface
   - Run the IAMSTART procedure

3. Define an IAM Data Set with IDCAMS
   - Add OWNER($IAM) to DEFINE CLUSTER Statement

4. Use as you do any standard VSAM Data Set
   - No JCL changes necessary if in Link List
   - Otherwise add a STEPLIB to the IAM Load Library

How to IAM a Data Set: Initial Steps

The first step is to download and install the product. Complete instructions are in Section 90 of the IAM Manual, and are also included as one goes through the process starting with the email received to acknowledge order of the product. Included in the process is a JAVA applet and an ISPF application to facilitate proper installation of the product. The process is quick and easy, SMPE is not used to install the product, it is just a few copies of some PDS type of files.

Even if this is just a trial product install, it is recommended that you place the final product load library in Link List. By having the IAM load library in the Link List you eliminate the need for adding STEPLIB’s to all of the jobs, TSO users, and CICS regions that you will be using the IAM files with. This can be done using the SETPROG operator commands to avoid having to do an IPL.

Once everything is all set, you need to start the IAM VSAM interface on each system / LPAR that will be using the IAM files. It is a single step started task (IAMSTART) that installs the IAM interfaces into logical LPA (using actual CSA storage), and then terminates. The IAM VSAM interface is easily shut down by running the stop started task (IAMPARE).

Once the IAM VSAM Interface is active, you can start defining and using IAM files instead of standard VSAM files.
How to IAM a VSAM Data Set: Example of IDCAMS DEFINE

```
DEFINE CLUSTER(NAME(my.vsam.ksds) -
    VOLUME(*) CYL(10 1) -
ADD       OWNER($IAM) ) -
    DATA(NAME(my.vsam.ksds.data) -
    RECORDSIZE(200 256) KEYS(16 0) -
    CISIZE(4096) FREESPACE(10 10)) -
    INDEX(NAME(my.vsam.ksds.index) -
    CISIZE(1024))
```

How to IAM a Data Set: Example of IDCAMS DEFINE

An IAM data set is defined by a standard IDCAMS DEFINE CLUSTER command. The only change is to add the keyword of OWNER($IAM) on the CLUSTER level. If the IAM load library is not in link list, then a STEPLIB will need to be added with the IAM load library. Once the defines is done, you can do an IDCAMS LISTCAT ALL on the data set, and you should get an IAMPRINT report with the file attributes. The IAM data set will appear in the IDCAMS SYSPRINT as a non-VSAM type of data set.

The data set can then be used just like you would a VSAM type of data set. It can be loaded by an IDCAMS REPRO, or as SORTOUT by the SORT, or by any other application program that normally loads VSAM datasets. It can then continue to be used as you would any other VSAM dataset, by other application programs or CICS. In general no changes other than adding the OWNER($IAM) to the DEFINE statement are needed to use IAM in place of an otherwise identically defined VSAM data set.

It is really just that easy!

The use of the OWNER($IAM) is the most common method used to define an IAM data set. There are a couple of other ways which include:

- Having the literal $IAM in the data set name
- Having the literal $IAM as part of the data class or storage class name.

Using a data class name for example DC$IAM could provide a method where not even the DEFINE CLUSTER has to be changed. The DC$IAM would need to be created as an actual SMS data class, then in the data class ACS routine, when a VSAM data set is being defined, code could be added to decide what data sets should be IAM, for example by a list of data set names, and then assign the DC $IAM to the data set in the ACS routine. The IAM define process would see that data class name, and then define the data set as an IAM data set.
IAMINFO Reports

IAM will optionally provide a one page run time report on dataset usage, which is produced each time an IAM dataset is closed providing that the job has an IAMINFO DD card. IAM will also optionally produce an SMF record with the same information, if so enabled by the IAM Global Options table. These reports contain information on the dataset attributes, various statistics about the content of the dataset, statistics on the resource usage, and statistics on the different types of I/O requests processed. The IAM Overrides used for each dataset are also included in the report, adjacent to the statistic that would be most influenced by the override. The IAMINFO reports are the primary information source for what is going on with each dataset. Many of the questions about the resource usage or performance on any particular IAM dataset can be answered with the IAMINFO reports.

The IAMINFO reports are easily obtained by providing an IAMINFO DD card in each job step that processes an IAM dataset. Normally this DD statement indicates a SYSOUT file, but can indicate an actual sequential dataset if desired. The only additional overhead when using IAMINFO is the overhead of actually formatting and writing out the report. IAM always maintains the various statistics that are reported on with IAMINFO, regardless of whether or not the DD statement is in the JCL.

Due to the potential large amount of output, you may not want to have an IAMINFO DD card in long running tasks, such as CICS, IAM/RLS, or IAM/PLEX. Use the IAM SMF records instead and the data will be available for reports if you need them.
IAM SMF Records

IAM can also optionally produce SMF records that contain all the information that appears on the IAMINFO reports. To enable this facility, the IAM Global Option SMF=YES must be specified, along with indicating the SMF record type to use by specifying RECTYPE=nnn. IAM provides the IAMSMF program that can produce IAMINFO reports from the SMF data and also produce a CSV file of the IAMINFO data for use in a spreadsheet. There is also the IAMSMFVS program, which produces more of a summarization set of reports from the IAM SMF data. Additionally, customers can provide their own programs to report on the IAM SMF data.

Important Messages that may appear on IAMINFO reports:

IAM368 SPECIFYING A BUFNO VALUE GREATER THAN nnnn MAY IMPROVE PERFORMANCE

Reason: IAM's Real Time Tuning buffer management found that, for this mix of data and file processing commands, if additional buffers had been available they would have been acquired. Additional buffers were not acquired because it would have exceeded the maximum buffers allowed for this job.

IAM373 REORGANIZATION OF THIS FILE IS RECOMMENDED

Reason: During Open and/or Close processing, IAM noticed that the size of the extended overflow area was such that it could cause a performance degradation. As a preventative measure, it is recommended, but not required, that the file be reorganized to move records out of the overflow area. This IAMINFO message should be accompanied with an IAMW22 message, which will indicate the reason that a reorganization is being recommended.
IAMINFO Reports - 2

The top portion of the report is primarily information about the data set itself, including basics such as the record length (maximum), the key size, key offset, and the defined control interval size. Included is the information on total records in the data set, number of records updated, deleted, and added to the file since it was last loaded. The time stamps for when the data set was defined, loaded, and last updated.
IAMINFO Reports – 2

The bottom two sections of the report consist of primarily statistics on resources used, including storage, buffers, physical I/O requests, and counts of the various logical I/O requests that were processed.
How to IAM a VSAM Data Set: IAM Override Facility

- Control IAM functions for which there are no JCL parameters
- Change parameters for IAM without affecting VSAM
- Specified by //IAMOVRID DD card
  - Card image (LRECL=80) data set
  - Can be DD *
  - Can be sequential dataset or PDS Member
- CREATE control card for define and file load
- ACCESS control card for other IAM dataset processing
- Described in Section 30 of the IAM Manual

How to IAM a Data Set: IAM Override Facility

The IAM Override Facility provides a mechanism through JCL to request various IAM functions, or specify various IAM parameters which do not have any corresponding JCL facility to do so. A second reason that people will use the IAM overrides is to alter various parameter values for IAM which could be changed otherwise, however they want to preserve the values that were used when the file was real VSAM in case they should need to use VSAM in the future. Values specified through IAM overrides take precedence over default values, or values specified through other mechanisms.

The two basic override control cards are a CREATE override for use when defining or loading an IAM data set, and the ACCESS override for use when accessing the IAM file for input or update. On each control card you specify the DD or DSN of the dataset(s) that the override values are applied to. Specifying a DD=&ALLDD applies the override values to any datasets that are not otherwise explicitly overridden.

The IAM Override control information is passed through a file defined by the IAMOVRID DD card, and can be either a SYSIN type of data set (DD *), a sequential data set, or a PDS member. For started tasks such as CICS or IAMRLS, or for use in procs the IAMOVRID DD would need to specify a sequential or PDS member.

The IAMINFO report for each file will include on the far right-hand column information about any override values that have been specified.
How to IAM a Data Set: Best Practices

While none of these practices are mandatory, they are recommended to avoid problems, and to have information should a problem of some type occur.

1. Keeping the current IAM load library in link list will eliminate need to use STEPLIB’s or JOBLIB’s to access your IAM files, and avoids problems if the STEPLIB is not used.

2. Activating the IAM VSAM Interfaces automatically as part of the IPL procedures ensures that IAM is always active so that there is always access to the IAM data sets.

3. Collecting the IAM SMF records enables IAMINFO reports to be generated from the SMF data using the program IAMSMF. These can be very useful in understanding or resolving issues or problems that can occur. It will also provide a way to get summary information of IAM utilization through the IAMSMFVFS program, and allow other user programs access to IAM statistical information.

4. While IAM does support the use of non-striped SMS Extended Format sequential data sets for IAM files, they may not perform as well, may require more dasd space, and seem to encounter more problems than regular DSNTYPE=LARGE data sets. Extended format adds a 32-byte suffix to each data block that causes IAM data blocks to be 32 bytes smaller, and for certain functions IAM needs to use BSAM rather than EXCP due to SMS restrictions.

5. There can be some problems when copying or restoring a multi-volume IAM data sets using a previously allocated data set, and sometimes the space usage on the multiple volumes may not make the best use of the space, particularly after a reorganization type of process. To be perfectly safe, it is always best to delete and re-define these data sets when they are reorganized or being reloaded.

6. Having LISTCAT data for newly defined and newly loaded IAM files is beneficial, particularly if some problem subsequently occurs with the data set. It can many times be helpful in some circumstances just to have it available.

7. This is less important if you are collecting the IAM SMF records, but even so it is sometimes handy to have the IAM statistics and data set information readily available if something happens with the job that may be causing problems.
How to IAM a Data Set: Best Practices – Global Options

1. Keep DSORG=PS set
2. Keep BELOWPOOL=YES set
3. Set SMF=YES and RECTYPE=nnn to an unused SMF user record type.
4. Set ENABLE=LARGE
5. Set ESDSINTEGRATED=5 if using IAM ESDS files and
6. Set ENABLE=XESDS or ENABLE=PSEUDO if using ESDS
7. Set ENABLE=EAV if IAM files will be on EAV volumes
8. Set COMPRESSTYPE=HW For Hardware Instruction

How to IAM a Data Set: Best Practices – Global Options

Except as recommended here or elsewhere in the manual, such as Section 90.10, Innovation highly recommends that users keep the default values. If there is a desire to change beyond what is recommended, please contact technical support for advice. A couple of default values are in the list because we have seen users change them and encounter possible problems by so doing.

1. Please keep DSORG=PS set. Use of DSORG=DA has caused problems with various data set management utilities and is therefore not recommended. Support is provided by FDREPORT to find IAM datasets on your dasd volumes.

2. Please keep BELOWPOOL=YES set. It has been used for several years now without problems, and is essential particularly for keeping below the line storage at a minimum under CICS.

3. Setting SMF=YES and RECTYPE=nnn (such as 201) or another number if that one is in use, is highly recommended. This enables collection of the IAM SMF records so that IAMINFO reports can be produced from the SMF data if needed without having to include an IAMINFO DD card in your JCL.

4. Innovation recommends setting ENABLE=LARGE to cause the default IAM data set format to be a large sequential. The primary reason is to help avoid the use of SMS Extended Format data sets, which while supported are not recommended.

5. If using ESDS files with IAM Data Compression that will be updated, Innovation recommends providing some free space in case an updated compressed record increases in size.

6. Setting either ENABLE=XESDS or ENABLE=PSEUDO is recommended if you are using ESDS files to avoid the 4 gigabyte limit on the amount of data in an ESDS caused by using the standard 4-byte RBA value.

7. If you have EAV (Extended Attribute Volumes – the super large volumes) then it is recommended to set ENABLE=EAV so that they will automatically be eligible for use without having to code EATTR(OPT) on the DEFINE CLUSTER.

8. If you prefer to use the IBM Hardware Compression instruction over the IAM software compression, then change set the COMPRESSTYPE=HW as your default.
The Global Options MAXOVERFLOW, OCOREO%, OCOREX%, PE, and RECFM deal only with IAM Compatible format files that most IAM users are not using and are not recommended for use.

You can change the following Global Options, although you are encouraged to consult with technical support for guidance on changing these from the default values: BUFSP, CICSBUFSP, CICSTXID, CRBUFOPT, DATACOMPRESS, IAMW22LIM, IMMEDWRITE, INDEXSPACE, MULTIVOLUME, RELEASE, RLS, RLSID, ROUTECODE, STORCLASS, VSAMBLOCKF; and any of the WORK options.
How to IAM a Data Set: Best Practices – Global Options Continued

9. To turn off automatic IAM data compression set the DATACOMPRESS value to eight 9's.
10. INDEXSPACE=64BIT or ALL
11. ENABLE=BUF64 and CRBUFOPT=64BIT for 64-bit buffers
12. Keep set to DATASPACE=2048
13. Be careful of using RLS=(REQUIRED, TABLE).

How to IAM a Data Set: Best Practices – Global Options Continued

9. To turn off automatic IAM data compression set the DATACOMPRESS value to eight 9's.
10. By keeping INDEXSPACE to either 64BIT or ALL helps prevent all address spaces from running out of storage due to the size of the indexes for the IAM files being processed.
11. To use 64-bit buffering as the default, set ENABLE=BUF64 and CRBUFOPT=64BIT. At this time (as of z/OS 2.1) there is some increase in CPU time caused by the page fix process of 64-bit virtual storage. If your use of IAM is primarily for reducing CPU time then we do not recommend setting those values. You may want to use the 64-bit buffering for address spaces such as CICS FOR’s or IAM/RLS or IAM/PLEX which can be done via the IAM Overrides.
12. Please keep DATASPACE=2048 set to avoid problems that some users have encountered with smaller values on running out of space when trying to load very large datasets.
13. Using RLS=(REQUIRED, TABLE) will require that the IAM/RLS or IAM/PLEX dataset name tables be loaded by starting either an IAMRLS or IAMPLEX address space before any IAM file can be opened, even if they do not require IAMRLS or IAMPLEX.
IAM Advanced Functions

- Alternate Index (AIX) and Relative Record (RRDS) Support
- IAMRLS – Single System Record Level Sharing
- IAM/PLEX – Record Level Sharing on a SYSPLEX

IAM Advanced Functions

The IAM product has various functions beyond just KSDS and ESDS type of data set support, some of which are optional and are additional cost functions. The advanced functions include Alternate Index Support along with RRDS support included as an optional additional cost feature. The IAMRLS Single System Record Level Sharing is an advanced feature that is included with the base product. The IAM/PLEX SYSPLEX Record Level Sharing is a second optional and additional cost function.
IAM Alternate Index Support

IAM Advanced Functions Alternate Index Support (AIX)

- IAM Performance to Alternate Index Processing
- Defined and Functions like VSAM Alternate Index, only faster
  - Define Base Cluster as IAM (OWNER($IAM))
  - Define Alternate Index – Automatically becomes IAM AIX
    - Unique or Non-Unique Keys
    - Upgrade or NoUpgrade
  - Define Path – Automatically becomes IAM Path
    - Update or NoUpdate
- No Application Program or CICS Changes
- Additional Cost Option to IAM Product
- Includes Support for VSAM RRDS type data sets

IAM Alternate Index Support

The IAM Alternate Index Support brings IAM performance and benefits to VSAM alternate index processing. Just define the base cluster as an IAM data set. If the base cluster is an IAM file then the alternate index and paths will automatically become IAM data sets, each residing on DASD in an IAM format. Note that if the base cluster is real VSAM, then the alternate index and paths remain as real VSAM. The PATH, rather than being an alias type of catalog entry, resides on dasd as a one-track data set. All of the connections are kept within the data sets themselves rather than in the catalog structure. There are no application program changes or CICS changes. There may be some changes to utility functions that backup, restore, copy, and rename alternate index spheres.

Available as an additional cost option to the IAM product. Feature also includes support for RRDS type of data sets.
IAM Advanced Functions
Record Level Sharing

- Enables shared access to IAM files with data integrity
- IAM/RLS
  - Sharing with multiple address spaces on single LPAR
    - Supports CICS, Batch, TSO, other address spaces
    - Included with base IAM product
    - All I/O for shared data set handled by IAMRLS address space
- IAM/PLEX
  - Sharing with Multiple Systems in a SYSPLEX
    - Supports CICS, Batch, TSO, other address spaces
    - All I/O to each shared data set routed to owning IAMPLEX
    - Additional Cost Option to base IAM product

IAM Advanced Functions: Record Level Sharing

IAM’s Record Level Sharing provides a mechanism to share IAM files with data integrity between multiple address spaces. With the IAM/RLS function, sharing is provided for multiple address spaces running on the same system / LPAR. IAM/PLEX provides that service for multiple address spaces that can be running on multiple systems / LPARS that are part of the same SYSPLEX.
IAM Advanced Functions: Record Level Sharing - 2

IAM/RLS and IAM/Plex support CICS with no application program changes required. For proper record lock management and coordination with Unit of Work events, IAM provides some CICS TRUE and GLUE exits. These exits are necessary for proper IAM/RLS and IAM/Plex function. The IAM files are defined to CICS as non-RLS VSAM data sets. With Version 9.2, IAM provides support for the CICS VSAM/RLS only operands of UNCOMMITTED, CONSISTENT, REPEATALE, and NOSUSPEND.

IAM/RLS and IAM/Plex processing do have a fault tolerance capability, and to the extent possible will keep running in the event of many types of errors. However in the event of an outage of an IAM/RLS or IAM/Plex address space, IAM provides an automatic recovery process under CICS to automate the appropriate handling of both the outage itself and recovery when the IAM/RLS or IAM/Plex address space is once again available. To be clear, such an outage would result in the temporary loss of access to the IAM files handled by the IAM/RLS or IAM/Plex address spaces.
IAM Advanced Functions: Record Level Sharing – 3

IAM/RLS and IAM/PLEX feature journal and recovery capabilities. Both provide support for use of the SYSplex System Logger facility. IAM/RLS can alternatively use a set of sequential DASD data sets for the journal process. CICS handles its own recovery processing which IAM/RLS and IAM/PLEX are aware of and fully compatible. The IAM recovery capabilities provided include an optional automatic backout process when a job abends, and utility programs IAMJREST and IAMJUTIL in support of running a manual recovery process if needed.

To help maintain data integrity in the event of a failure, IAM/RLS and IAM/PLEX do offer a persistent record locking capability that will re-establish the record locks that were held at failure when the IAM/RLS or IAM/PLEX address spaces are restarted.

Batch applications that perform a large volume of updates on recoverable files will need to use the IAM record level sharing syncpoint process to prevent an excessive number of record locks from being held. While an automatic syncpoint capability is provided, users do need to consider their recovery needs in the event of failure while the batch jobs are running and determine whether the automatic syncpoint process is sufficient or if they need to make batch program changes to appropriately handle a syncpoint and restart process.
IAM Advanced Functions: Record Level Sharing – 4

Both IAM/RLS and IAM/PLEX use the concept of an RLSID, which is a 4-character name for each IAM/RLS and IAM/PLEX address space. The IAM datasets that are using IAM/RLS or IAM/PLEX services are assigned to a specific IAM/RLS or IAM/PLEX address space through the RLSID. The linkage to a particular address space can be established in one of a few different ways:

1. By use of a common data set name table, which indicates the names of the IAM datasets to be handled by an IAM record level sharing process, and the RLSID of the address space to perform the I/O services for that data set. Most of our users use this facility. This is activated by specification of the IAM Global Option RLS=(TABLE,...) and specification of that table to each IAM/RLS or IAM/PLEX address space.

2. A dataset that will use IAM record level sharing can be defined with an RLSID when it is defined with the IAM CREATE override. The RLSID value can be changed or removed by performing a DEFINE RECATALOG with the appropriate IAM overrides.

3. Can be specified via an IAM ACCESS Override on the job steps that need to utilize the IAM record level sharing capability.

4. It can be assigned a default RLSID through the IAM Global Options.
IAM Advanced Services
Record Level Sharing: IAMPLEX

- **RLSGROUP**: Refers to a group of related IAM/PLEX address spaces that form an XCF group that are able to directly communicate with each other
  - IAM/PLEX address spaces assigned to an RLSGROUP by startup parameter RLSGROUP
  - Any particular application address space (CICS region, batch job) can only access those IAM datasets that are being handled by an IAM/PLEX address space within a single RLSGROUP
  - All member IAM/PLEX address spaces in an RLSGROUP will share the same System Logger
  - XCF services are used to communicate to each IAM/PLEX address space within the RLSGROUP

IAM Advanced Functions: Record Level Sharing – 5

The IAM/PLEX services uses an RLSGROUP name to link together the different IAM/PLEX address spaces running on each system / LPAR. This is used to facilitate the XCF communications within the SYSPLEX for IAM/PLEX processing. All of the IAM data sets that will be processed by any job or CICS address space that are being handled by IAM/PLEX must belong to an IAM/PLEX address space that is in the same RLSGROUP.
IAM Advanced Functions: Record Level Sharing – 5

The diagram is of a basic IAM/PLEX implementation, and is intended to show how everything is connected. An important thing to notice in the diagram is that any particular IAM data set is directly processed by only one IAM/PLEX address space. All I/O requests related to a specific data set are communicated to the owning IAM/PLEX address space through the SYSPLEX XCF communications facility. Therefore good performance in the IAM/PLEX environment will need an adequate XCF configuration to handle the expected volume of requests.
IAM Version 9.2 Enhancements

IAM Version 9.2 includes the following enhancements:

- **z/HPF I/O Architecture Support**
- **64-bit Virtual I/O Buffers**
- **Enhanced I/O Error Information**
  - Includes full 32-byte sense data when available
- **IAM WTO Message Enhancements**
  - Use of Multi-Line WTO messages for automated operations
  - Split IAMW22 reasons into separate message numbers
- **IAM/RLS and IAM/PLEX Support for VSAM RLS functions**
- **Enhanced IAMSMFVS Report**
- **GA: February 2015**
IAM in Summary

- Transparently improves VSAM application performance
- Uses a simpler file structure, dynamic buffer management and caching
- Reduces physical I/O (EXCP’s) by 40% to 80%
- Cuts CPU time by 20% to 40%
- Reduces elapsed processing times 20% to 60%
- Data Compression can save DASD space by 20% to 50%
- Provide Record Level Sharing across a SYSPLEX

IAM in Summary
The IAM product is proving itself everyday to provide better VSAM application performance than native VSAM does itself. It is easily implemented requiring no program changes, and generally no JCL changes other than on the DEFINE step to indicate that the data set is to be an IAM data set. Even that step can be eliminated if desired through appropriate specifications in the ACS Data Class routine. IAM reduces the physical I/O, CPU time, which results in elapsed time savings. Additionally IAM can help reduce DASD space requirements through data compression. IAM also has record level sharing facilities that ensure data integrity.
Vendor Products that Use IAM

Provided is a list of several of the VSAM application products our customers are using IAM files instead of VSAM for better performance.
Vendor Products that work with IAM

The list on this and the following page are examples of the many different products that work with IAM files, some of which have explicit support added for IAM. As can be seen, IAM is supported by many of the commonly used products. We want IAM to be your high performance alternative to VSAM and to help accomplish that we strive to minimize the need for any changes or other special considerations when using the IAM product.

<table>
<thead>
<tr>
<th>DASD MANAGEMENT SOFTWARE</th>
<th>JOURNALING AND RECOVERY</th>
<th>PERFORMANCE MONITORS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDR/ABR (INNOVATION DATA PROCESSING)</td>
<td>FILE SAVE (CA)</td>
<td>OMEGAMON (TIVOLI)</td>
</tr>
<tr>
<td>FDCREPORT (INNOVATION DATA PROCESSING)</td>
<td>DRS (BMC)</td>
<td>THE MONITOR TMON (ASG)</td>
</tr>
<tr>
<td>FDRRERORG (INNOVATION DATA PROCESSING)</td>
<td>AR/CTL (BMC)</td>
<td>STROBE (COMPUWARE)</td>
</tr>
<tr>
<td>DF/SMF (IBM)</td>
<td>CICS/VR (IBM)</td>
<td></td>
</tr>
<tr>
<td>DF/HSM (IBM)</td>
<td>RRDF - Remote Recovery Data Facility (E-Net)</td>
<td></td>
</tr>
<tr>
<td>DF/DSS (IBM)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA ALLOCATE (formerly VAM) (CA)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMS/OS (CA)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POOLDASD (BOOLE &amp; BABBAGE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAINVIEW SRM STOPX37/II (BMC)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© Copyright 2015 INNOVATION Data Processing. All rights reserved.
## Vendor Products that work with IAM

<table>
<thead>
<tr>
<th>MISCELLANEOUS PRODUCTS:</th>
<th>PROGRAMMING LANGUAGES:</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE-AID (COMPUWARE)</td>
<td>VS/COBOL (IBM)</td>
</tr>
<tr>
<td>File Manager (IBM)</td>
<td>COBOL II (IBM)</td>
</tr>
<tr>
<td>ABEND-AID (COMPUWARE)</td>
<td>FORTRAN (IBM)</td>
</tr>
<tr>
<td>CICS (IBM)</td>
<td>PL/1 (IBM)</td>
</tr>
<tr>
<td>CONNECT; DIRECT (IBM-Sterling)</td>
<td>BAL (IBM)</td>
</tr>
<tr>
<td>ISPF (IBM)</td>
<td>CA/OPTIMIZER (CA)</td>
</tr>
<tr>
<td>NETWORK DATA MOVER (CA)</td>
<td>SAS (SAS INSTITUTE)</td>
</tr>
<tr>
<td>(NDM)</td>
<td></td>
</tr>
<tr>
<td>SELCOPY and SELCOPY/i (CBL Compute (Bridgend) Ltd)</td>
<td>SHARING PACKAGES:</td>
</tr>
<tr>
<td>SHRINK (CA)</td>
<td>IAM/PLEX (IDP)</td>
</tr>
<tr>
<td></td>
<td>SHARE OPTION 5 (CA)</td>
</tr>
<tr>
<td></td>
<td>SYSB (H &amp; W)</td>
</tr>
<tr>
<td>SECURITY PRODUCTS:</td>
<td></td>
</tr>
<tr>
<td>RACF (IBM)</td>
<td>SORT PRODUCTS:</td>
</tr>
<tr>
<td>ACF/2 (CA)</td>
<td>SYNSORT (SYNSORT)</td>
</tr>
<tr>
<td>TOPSECRET (CA)</td>
<td>DF/SORT (IBM)</td>
</tr>
<tr>
<td></td>
<td>CA/SORT (CA)</td>
</tr>
</tbody>
</table>

© 2012, Innovation Data Processing
Additional Resources

- Innovation Website:  http://www.fdr.com
- FREE Trial  http://www.fdr.com/riskfreetrial
- Request Latest Version  http://www.fdr.com/upgrade
- Support Email:  support@fdrinnovation.com
Closing: IAM
Improving Performance for VSAM Applications

Richard Morse
Innovation Data Processing
rmorse@fdrinnovation.com