IBM Content Manager OnDemand

Best Practices Guide

March 2015

IBM Software Group
Enterprise Content Management
IBM Content Manager OnDemand Development Team

Trademarks

IBM Content Manager OnDemand Best Practices Guide, March 2015.

Table of Contents

The team that wrote this document

This document was produced by a team of specialists from the CMOD Development team.

Ben Boltz – Ben joined IBM in 1994 with 10 years of experience as an IT professional. He started out installing CMOD for our very first customers as a service offering. He joined the CMOD development team a year later, and is now the Team Lead.

Chris Lewis – Chris joined IBM in 1992 as a co-op for the Advanced Technology Group while attending graduate school at the University of Colorado, Boulder. He was then hired full-time to work on OnDemand for Multiplatforms, where he continues to work as a software developer.

Darrell Bryant – After joining IBM, Darrell worked in Process Engineering. A transfer to the local branch office led to 10 years as a Systems Engineer specializing in S/36 and AS/400 systems. His next assignment was with ISSC as a part of their AS/400 systems team. In 2000 Darrell joined the OnDemand team. He has performed a mix of activities including services, education, support, and testing. Darrell is now the lead tester for OnDemand for i. He also develops and teaches workshops to customers and partners, and is the editor and distributor of the OnDemand Newsletter.

Hassan (Al) Shazly – Hassan (Al) is a Development, QA and Performance Manager with Enterprise Content Management OnDemand and has been with IBM since 1996. He has over 35 years of software management and development experience in various business and scientific applications. He has written over 20 publications and has presented at multiple technical conferences. Hassan holds a Ph.D. in Remote Sensing and Image Processing from the University of South Carolina.

Nancy O’Brien – Nancy started at IBM as an applications programmer. She later transferred to a branch office as a Systems Engineer, where she performed her first OnDemand (then known as R/DARS) implementation. After many more implementation service engagements, she joined the OnDemand development team and continued to do implementation services, training, support, testing, and technical writing. She currently focuses primarily on technical writing and testing.

Vanessa Stonesifer – Vanessa joined the IBM Software Group in 1999 as a Technical Leader and Consulting Services Specialist in the Enterprise Content Management Software Group Services division – Content Manager OnDemand. Vanessa has over 24 years of experience in Information Technology and with over 20 years of experience with OnDemand. Vanessa was the Technical Team Lead for Content Management Lab Services for more than 10 years, providing and performing implementations, performance optimizations, and migrations for OnDemand. Vanessa has since taken on a new role in OnDemand Development – QA and Performance Development. Vanessa has also presented at multiple technical conferences and Content Manager OnDemand User Groups and has co-authored a Content Manager OnDemand Redbook.

Many thanks to the following people who have shared their knowledge and contributed material for this document:

Bernie Wolf , Brian Hoyt, Dan Harrison, Debbie Matamoros, Paula Muir, Sharon Weinberg, Trang Doung
IBM Software Group, Content Manager OnDemand Development

1 Introduction

IBM® Content Manager OnDemand (CMOD) is the premier data archiving and retrieval system on the global market. CMOD is installed at thousands of customer locations worldwide serving a user base in the millions of users on a daily basis. The CMOD product is extremely scalable and flexible, allowing customers to create customized solutions that best meet their requirements. CMOD provides extremely high performance for loading, retrieving, and expiring of data.

This document describes the best practice recommendations for CMOD. The intent is to provide CMOD solution designers, system administrators, and system programmers an overview of the factors that impact the performance, scalability, and reliability of CMOD applications running on all platforms. This document generically describes implementation considerations for the various operating systems on which CMOD can be installed. At a high level, the supported operating systems are z/OS, IBM i, and Multiplatforms (AIX, Linux, Linux on system z, Sun Solaris, and Windows). Sections in the document that apply to a specific operating system environment are identified.

This document is not intended to be a replacement for any platform specific or release-specific CMOD publication or for any information contained in the CMOD Knowledge Center. It is intended to be a complement to the CMOD release-specific documentation. For more detailed information regarding CMOD, refer to the CMOD Knowledge Center and/or publications that are specific to your platform and software version. Links are provided at the end of this document.

Important: The information in this document concerning non-IBM products was obtained from the suppliers of those products. IBM has not tested such products and cannot confirm the accuracy of the performance, compatibility or any other claims related to non-IBM products. Questions about the capabilities of non-IBM products should be addressed to the suppliers of those products.

1.1 Planning and implementation

"IF YOU FAIL TO PLAN YOU PLAN TO FAIL." Anonymous

When planning for your CMOD system, there are many aspects that need to be taken into consideration. These are listed below. This document discusses the server configuration and architecture components of the planning and implementation process.

Planning the CMOD installation

- Define the CMOD data hierarchy, including application groups, applications and folders
- Define users, user groups and security settings
- Define load procedures and related exits (if any)
- Define the retrieval exits (if any)
- Allocate personnel and assign job roles/responsibilities
- Train - onsite and/or offsite as appropriate

Software considerations

- Obtain the latest CMOD code maintenance level. You can read about the updates available in the latest release of CMOD for your platform in the "What's New" information available on the web at www.ibm.com/support/docview.wss?uid=swg27021523 or at ibm.com using '7021523' as your search criteria.
- Ensure that the clients and server are at compatible code levels. The "Compatibility matrix for the Content Manager OnDemand clients and servers" is available on the web at
- www.ibm.com/support/docview.wss?uid=swg21392275 or at ibm.com using '1392275' as your search criteria.

- If you are using library and object servers (on one or more platforms), then ensure that they are all at the same code level.

Planning for performance, scalability and high availability

- Perform initial capacity planning and sizing estimates
- Consider your future growth requirements
- Examine various system architecture implementations
- Document the current and projected workload
- Define measurable performance and scalability objectives

Solution design choices and performance tradeoffs

- Understand topology choices
- Define server(s) (multiple logical partitions (LPARs), multiple systems and multiple CMOD instances)
- Determine client selection (types of client, two-tier and/or three-tier)
- Assess subsystem impact (ODF, SAPI, ODWEK, etc.)
- Determine feature choices (for example, exits, high availability, text search, etc.) and their performance implications

Initial system installation and tuning

- Set up a sandbox to test the design
- Test functionality, performance and scalability

Monitoring and maintaining performance

- Maintain a performance profile of the workload metrics and resource utilizations on the CMOD servers
- Observe trends so as to be able to prevent problems from arising

Note: IBM's CMOD Lab Services offers support for all of the above functions (as well as many others).

1.2 General factors affecting performance

Reliability and availability are considered to be two of the most important aspects of a systems solution. The scalability of CMOD allows you to create systems that perform at the level required by your organization. Performance tuning has to be done on a case-by-case basis, because every CMOD implementation has different characteristics based on customer data and usage requirements. But there are various items that need to be taken into consideration in all implementations. These considerations and requirements can be divided into two general categories:

CMOD considerations

CMOD usage requirements (at a very high level) are characterized by:

- Data type (AFP, Line, SCS, PDF and so forth)
- Quantity of data, report size, document size, report to document ratio, data density (white space)
- Indexer, number of indexes, number and location of triggers and key fields
- Exits, pre-load and pre-view requirements
- Data conversions, compression, encryption

- Data loading patterns (scheduling of load jobs, parallel loads)
- Data retrieval patterns (total number of users, number of concurrent users, automated retrieval processes)

Environmental considerations

The computing environment includes items such as:

- Hardware capacity (CPU, memory, disk, channels, and so forth)
- Software (operating system level, database, network, applied PTFs)
- Network usage (local, intranet, Internet)
- Other work performed on the system (competing for hardware, software, and network resources)
- System and workload manager settings (which affect hardware or the network or both)

This document discusses both sets of considerations at a high level, focusing mainly on the server components of the CMOD product.

1.3 CMOD system overview

A CMOD instance consists of a single logical server. This single logical server can exist as either

1) a single physical server that includes both a logical library server and a logical object server (IBM i is packaged this way), or
2) a single physical CMOD library server and one or more physical CMOD object servers.

Both of these implementations are available on Multiplatforms and z/OS systems.

These physical servers can exist on a single system image or logical partition (LPAR), on multiple LPARs, or on different systems (running the same or different operating systems and connected via TCP/IP) as shown in the figure below. Also included in the figure are the load process (by means of which data is ingested into CMOD), and the CMOD clients (the most popular of which include ICN, the Windows client, and the Java (ODWEK) APIs). The figure below shows the general layout of a CMOD instance (logical server):

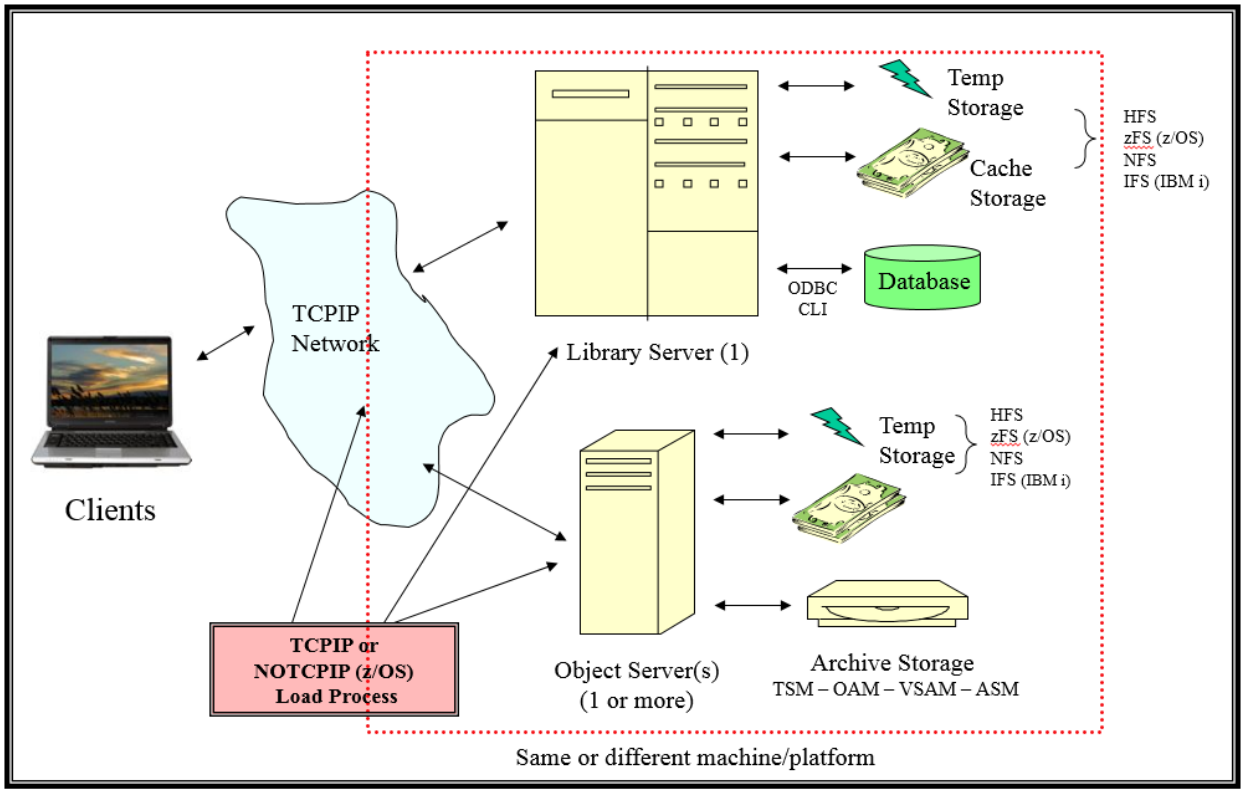


Figure 1. A Content Manager OnDemand Instance

The library and object servers contain the following subcomponents and functions:

The CMOD library server

There is a single library server per CMOD instance which:

- contains the DB2 database (other databases supported include Oracle and SQL server), which contains
  - CMOD system tables. These tables contain information for setup, configuration, and user accounts
  - CMOD metadata (application group data) tables. These tables contain the document indexes and pointers to the stored document location
- performs all database searches for both the system and the metadata tables
- performs user authentication
- performs logging

The CMOD object server

There are one or more object servers (on one or more systems) per CMOD instance. (For IBM i, there is a single object server per CMOD instance.) The object server(s):

- stores and retrieves objects (containing documents) in storage nodes. Storage nodes can be defined as:
  - Cache-only storage node (all platforms)
    - In a hierarchical file system (for example zFS, HFS, IFS, NSF…)
  - Tivoli® Storage Manager (TSM) storage node (MP only)
    - With or without cache
  - Archive Storage Manager (ASM) (IBM i only)
    - With or without cache
  - IBM DFSMSdfp Object Access Method (OAM) storage node (z/OS only)
    - With or without cache
  - Virtual Storage Access Method (VSAM) storage node (z/OS only)

            - Note 1: VSAM storage nodes are provided for historical data compatibility
            - Note 2: on z/OS the IBM recommended method for scalability and performance is OAM.
        - With or without cache
- transmits “document” data directly from the object server to the requesting client using TCP/IP

Legacy System Compatibility

- **z/OS only**: If there is a requirement to retrieve data captured using version 2 of CMOD for z/OS or IAFC, then the ARSOD, ARSMDT, and ARSODIND (migration only) tables must be created on (or at least be accessible by) the CMOD object server. Refer to the Content Manager OnDemand Migration Guide (SC19-1216) for further details on V2 to V8 migration.

- **IBM i only:** If there is a requirement to retrieve data captured using CMOD Spool File Archive on IBM i, refer to the CMOD Spool File Archive to Common Server Migration Reference for details. The Migration Reference can be found on the web at www.ibm.com using ‘7021252’ for your search criteria.

Regardless of the physical setup, all communications between the server and the clients (regardless of client type) execute the same proprietary CMOD communication protocol over a TCP/IP network.

The server supports multiple client types:

- Two-tier (OnDemand end-user client, OnDemand Administrator client, CICS client)
- Three-tier (IBM Content Navigator (ICN), structured APIs)
- The ODWEK Java APIs are two-tier components that might be used in two- or three-tier implementations

The workload/processing distribution between the clients and the CMOD server will differ based on the client type(s) selected.

1.4 Choosing a platform

A CMOD server can run on many different operating system and hardware environments. It can be setup to run on a workstation and can scale up to a zSeries complex. The following is a list of some of the factors that enter into the decision to implement CMOD on one platform vs. another:

- Existing hardware platforms
- Projected future hardware requirements (standardization, consolidation…)
- Existing personnel and skill set
- Current workload (number of users, quantity of data…)
- Projected future workload (number of users, quantity of data…)
- Interfacing with other systems (software and/or data)
- Vendor’s ability to support the environment (hardware, software and users over any geographic extent)

2 Performance, scalability, reliability, and availability architectures

CMOD is designed to be a “lightweight” process, meaning that the CMOD code itself does not require extensive system resources in order to perform the functions required of it. Typically, CMOD installations handle both large quantities of data and large numbers of users. The complexity and frequency of the database queries and the total quantity of data being stored, retrieved or expired at any one time are the main contributors to the resource consumption on the server.

2.1 Performance, scalability, reliability, and availability defined

- **Performance -**There are two components to performance: throughput and response time. They are defined as follows:

  Throughput - The number of transactions (CMOD requests) that can be satisfied per unit of time. The more transactions executed per unit of time, the higher the throughput. Higher throughput implies that more users can be served concurrently and more load jobs can be run in parallel. If the throughput values are low, then the system might not be able to support the required number of users.

  Response time - The amount of time it takes to service a single transaction (CMOD request). Faster response times imply that the users are able to retrieve their data faster from the archive, which in turn leads to happier users. If the response time is slow then users will be dissatisfied with the system.

  A high performance system such as CMOD provides both a high throughput and a short response time.

- **Scalability -** The ability of the CMOD system to handle a growing amount of work with no degradation in performance. A CMOD system’s performance will improve with the addition of hardware and network resources and is thus deemed to be a **scalable system**. There are two type of scalability:

  Horizontal scalability (or scale out) - This is achieved by adding more nodes/systems/LPARs to the CMOD instance. For example, adding more object servers to the CMOD instance is an example of horizontal scalability.

  Vertical scalability (or scale up) - This is achieved by adding more resources to a single node in a CMOD instance. Typically this involves additional CPUs, memory, disk, or networking hardware.

  CMOD is both horizontally and vertically scalable.

- **Reliability -** The ability of CMOD to perform and maintain functionality during regular workloads as well as during peak workloads. Peak workloads might occur on a regular basis (for example when everyone signs on at 9:00 am) or periodically (at the end of the month when more processing than usual occurs) or sporadically (for example when a special event occurs, such as a sales drive that results in more users using the system).

- **Availability -** A measure of the time that a CMOD server or process is functioning normally, as well as a measure of the time the recovery process requires after a component failure. It is the downtime (unavailability) that defines system availability. Availability is the amount of system uptime when the system is fully functional and accessible by all users.

  Availability requires that the system provide some degree of redundancy in order to eliminate single points of failure (SPOF). The greater the redundancy provided, the higher the availability of the system. A single physical machine is still a single point of failure. For this reason, a high availability system topology typically involves horizontal scaling and redundancy across multiple machines.

  High Availability - A highly-available system would have an availability limit of at least 99% which would allow for an average of 15 minutes per day to perform maintenance tasks (during which period the system would not be accessible to users). In certain environments (for example. z/OS parallel sysplex), it is possible to configure CMOD such that it is continuously available, (24 x 7 x 365). The degree of high availability achieved is a function of the amount of redundancy within the system and the degree to which this redundancy is automatically enabled. There are basically two redundancy techniques:

Passive redundancy (active-passive) - This is achieved by including enough excess capacity in the design to accommodate a performance decline, such as two CMOD servers ((known as ARSSOCKD on z/OS and Multiplatforms) accessing the same system tables and archive. If one server fails, then the other server is available to take on the workload.

Active redundancy (active-active) - Used to achieve high availability with no performance decline. For example, if the peak workload requires 1.5 CMOD Servers, then 3 CMOD servers are configured to work in parallel. If one of the servers fails, then the other two servers can take on the full workload with no performance degradation. This is typically implemented as a parallel sysplex on z/OS.

Systems typically become unavailable due to the lack of one or more of the following:

1. Change control procedures (a failure to implement the appropriate procedures from installation verification through performance testing before placing the system into production)
2. Monitoring of production system components (including total system workload, network issues)
3. Implementing high availability solutions (redundant systems and network connections)
4. A comprehensive backup (and restore) process that is tested on a routine basis

Note: There is a cost to implementing highly-available high performance systems. This cost needs to be weighed against the cost of not implementing such systems.

The following sections provide more details on example system implementations that allow for high performance, scalability, reliability and availability.

2.2 Scalability - all platforms

A CMOD instance can be scaled from a single system image that performs all of the required tasks (data loading, library storage, and object storage) to a multiple system/multiple logical partition (LPAR) configuration, allowing for higher levels of performance and availability. When CMOD is distributed among multiple systems these systems might be:

- **Single technology systems**: the CMOD instance consists of systems that are of the same architecture. For example, all systems might be AIX systems, or
- **Multiple technology systems**: The CMOD instance might consist of systems of different architectures. For example, the library server and an object server might be on a z/OS system, two other object servers might be on AIX systems, and another object server might be on a Windows system.

In both of the above scenarios, the configuration results in a single CMOD instance from both the administrative and user perspectives.

This flexibility and scalability allows customers to configure their CMOD systems to meet both their workload and their operational requirements.

Examples of these configurations are conceptually illustrated in the following diagrams. Note that these are only a sample of the possible configurations used to illustrate the basic scalability features.

The figure below illustrates a single CMOD instance. In this figure, the CMOD server supports the library server, one or more object servers, and one or more load processes. The following sections provide examples of how the CMOD server can be scaled both vertically and horizontally.

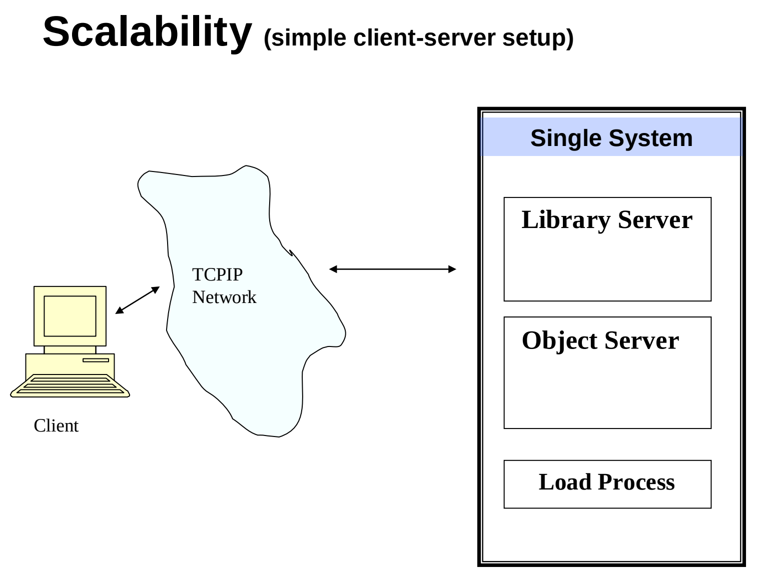


Figure 2. **Scalability (simple client-server setup)**

2.2.1 Vertical scalability

Growing the system - CMOD is vertically scalable to the extent that the system that it is running on is scalable. Vertical scalability is achieved by adding more hardware to the system. This might be in the form of CPUs, memory, disk, I/O, or network. The limit to the amount of vertical scalability possible is the architectural hardware constraints of the system. For example, if the system only supports 24 GB of memory then that memory limitation can only be overcome by buying a larger system.

Using a larger system - With CMOD, this can be achieved in one of two ways:

1. Installing a larger system within the same family/architecture. For example, moving from an entry level AIX system to an enterprise level AIX system.
2. Installing a larger system from a different architecture/family. For example, moving a CMOD server from a Windows system to an AIX system.

Application design - Modern computer systems contain multiple cores and are capable of multithreaded processing. Modern operating systems are also designed to allow for parallelism in operations. To take advantage of these hardware and software features, an application must be designed such that it can execute in parallel at multiple levels. CMOD is designed to take advantage of both.

At the process level, the CMOD server executes multiple processes. They are:

1) database management (on the library server)
2) cache and archive storage management (on one or more object servers)
3) data loading
4) expiration

At the thread level:

1) The library server is designed such that it is multithreaded and can service multiple incoming data requests on different threads and perform multiple database queries in parallel.
2) The object server is also multithreaded. This allows for multiple users to concurrently retrieve data from the CMOD archive.

Archive access parallelization - When accessing the TSM or OAM archives, a store or retrieve request is sent to the archive manager. The archive manager then either stores or retrieves the data and returns the result to the CMOD server. If this process is conducted in a serial fashion, then the archive access mechanism becomes a bottleneck. To overcome this potential bottleneck, CMOD has implemented connection pooling to the archives.

CMOD maintains a pool of connections to the archive. When an archive store or retrieve request is received, an available connection from the pool is selected to perform the request. This allows for both faster access to the archive (by eliminating the startup process each time a connection is requested) and for the parallel execution of the store or retrieve operations.

On IBM i, when accessing the ASM archives, connection pooling is not required for store requests. When a store request is made, ASM opens a connection and keeps it open until the data store request is complete. In addition, ASM allows aggregation of objects, sending significantly fewer objects to storage media than would otherwise be sent without aggregation.

On Multiplatforms and z/OS, it is also possible to aggregate documents loaded from ODWEK before storing them in the archive. The document is “stored” to cache where it is appended to the storage object until the document reaches the 10 MB (defined storage object size) at which point it is migrated to a storage manager such as TSM. More information can be found on the web at www.ibm.com using ‘1587507’ for your search criteria.

2.2.2 Horizontal scalability – library server

Even though the CMOD allows for a single library server per instance, this library server can be scaled horizontally. The library server is scaled horizontally using one or both of the following methods:

1) The database tables (both the system and the application group) can be placed in different databases (z/OS) or different tablespaces (Multiplatforms and z/OS) at the table level. Thus each of these tables can scale to the maximum practical size supported by the database in terms of maintenance and performance. There is no CMOD imposed limitation.
2) The application group data table design is such that:
   a. Any number of application groups can be created to support the required data to be archived.
   b. Each application group can be segmented into multiple tables where the table segmentation is based on size.
   c. Each of these application group data tables can be placed in a separate database (z/OS) or tablespace (Multiplatforms and z/OS)

2.3 Scalability – Multiplatforms and z/OS

2.3.1 Horizontal scalability – multiple object servers – scaling data

In this example, the CMOD system is horizontally scaled by placing the library server, object server(s) and load process(es) on multiple systems.

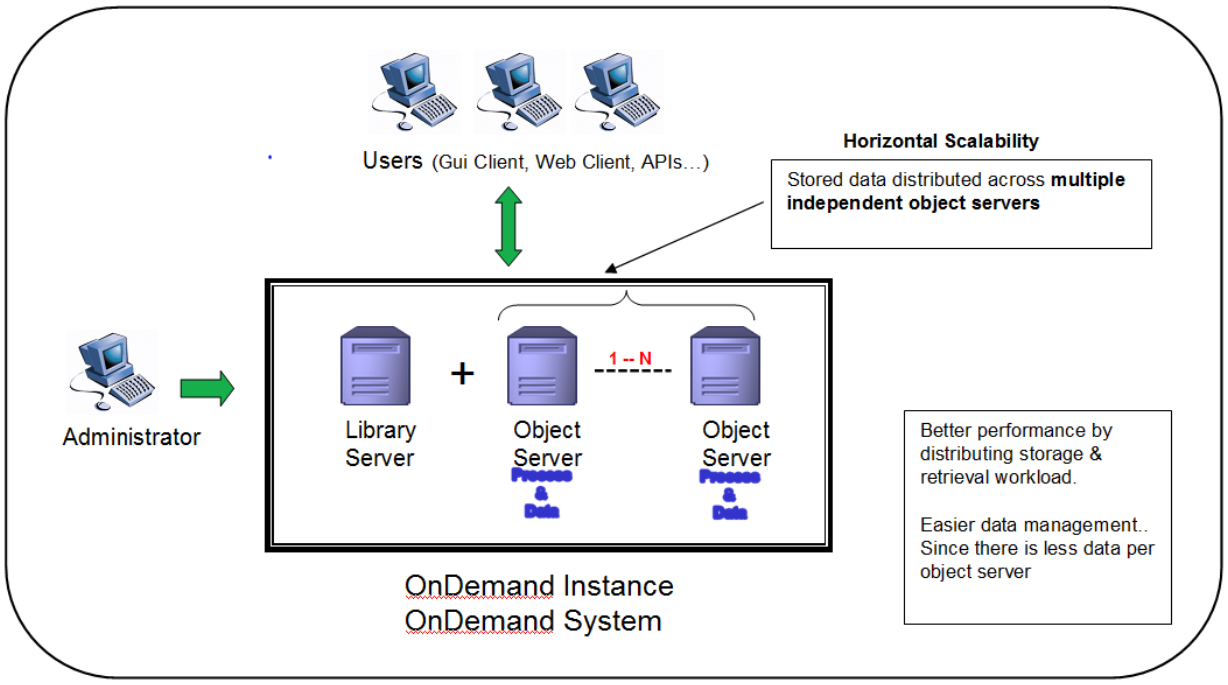

Users (Gui Client, Web Client, APIs…)
Horizontal Scalability
Stored data distributed across multiple independent object servers
Administrator
Library Server
+
Object Server
Process & Data
1 -- N
Object Server
Process & Data
OnDemand Instance
OnDemand System
Better performance by distributing storage & retrieval workload.
Easier data management.. Since there is less data per object server

Figure 3. Horizontal scalability - 1 library server and multiple object servers (z/OS and Multiplatforms)

This form of horizontal scalability provides better performance, reliability and scalability by distributing the storage and retrieval workload over multiple systems. Data management is also simplified since there is less data per object server.

From a CMOD perspective, there is no limit on the number of object and load process servers. Each of the servers can execute to its maximum capacity. Operational limitations will be imposed by the TCP network bandwidth that connects all the servers and available datacenter floor space. Both of these constraints can be reduced by placing multiple servers in a rack mounted configuration.

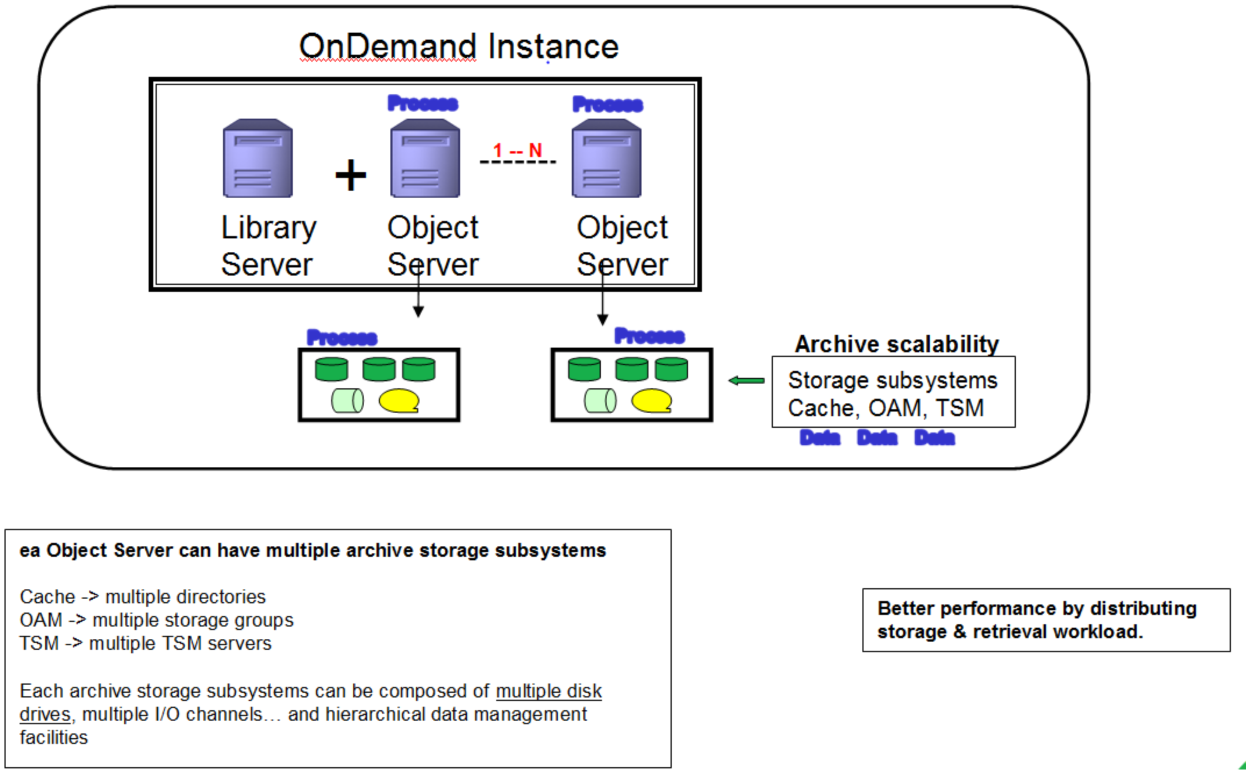
It is worth noting that, in this example and in all of the following examples, from an external perspective this is a single CMOD instance. The fact that the system is composed of multiple distributed systems is totally transparent to both:

- the CMOD administrator, who continues to administer the system through the OnDemand Administrator client exactly as if he/she were administrating a truly single system, and
- the CMOD users who continue to access the whole system through a single IP address (that of the library server) and, from their perspective, see only a single system.

2.3.2 Horizontal and vertical scalability – storage manager – scaling data even further

This form of horizontal scalability provides better performance, reliability and scalability by distributing the storage and retrieval workload over multiple storage subsystems within each object server.

An object server is designed to control the storage and retrieval of the archived data. The archived data is stored in a storage subsystem. The number and architecture of these subsystems can be scaled to the limitations of the subsystem. In other words, each object server can support one or more storage subsystems and each storage subsystem can be composed of multiple storage devices. This is illustrated in the figure below.



ea Object Server can have multiple archive storage subsystems

Cache -> multiple directories
OAM -> multiple storage groups
TSM -> multiple TSM servers

Each archive storage subsystems can be composed of multiple disk drives, multiple I/O channels… and hierarchical data management facilities

Better performance by distributing storage & retrieval workload.

Figure 4. Horizontal and vertical scalability - 1 library server and multiple object servers (z/OS and Multiplatforms) - scaling vertically

Each object server can have multiple storage subsystems of different types. The subsystem types are:

- **Cache -** The cache storage subsystem is controlled directly by the object server. Data is written and read directly from cache. Cache consists of one or more cache file systems. Each cache file system can be mounted on a different device in its own directory. Each device can be placed on its own independent I/O interface/channel. There is no CMOD imposed limit on the number of devices.
- **Tivoli Storage Manager (TSM) -** TSM is an archive storage subsystem. The CMOD object server sends data to and requests data from TSM. Each TSM server can be installed on its own system (for example an AIX server). The CMOD object server allows for the connection of multiple TSM servers. So, for example, if the CMOD object server is an AIX system and the data managed by that object server is stored in three TSM archives (all of which are AIX systems), then the total processing capacity for that object server is four AIX systems. Each of the AIX systems can be configured with as much CPU, memory, disk, and I/O as is needed up to its architectural limitation. If more capacity is needed, then more TSM servers and/or object servers can be added. **Note:** for better performance the TSM server should be installed on a system or LPAR that is separate from the CMOD server
- **Object Access Method (OAM) -** OAM is a z/OS-only archive storage subsystem. By design, there is only one OAM archive per system. Scalability within the archive is achieved by increasing the number of storage groups. A z/OS system can be grown by increasing the number of CPUs, memory, disk, and/or I/O. If more capacity is needed than can be provided by a single system, then z/OS allows for multiple systems to be connected in a “parallel sysplex.” All of these systems would then be able to access the same OAM subsystem, thus providing unparalleled scalability, reliability, availability and performance.

Both TSM and OAM provide hierarchical data management facilities. This allows data to be stored on different devices based on the age or predicted frequency of data access. For example, frequently accessed data might be placed on high speed disk (or Flash disk, or Solid State Disk) and infrequently accessed data might be placed on tape. When the data is requested by a user, the location of the data is totally transparent to the user. The only perceived difference from a user perspective is the response time which is a factor of the type of device on which the data is stored. In this example, tape access would be slower than disk access.

In summary, better performance is achieved by distributing storage and retrieval workload over multiple systems and multiple devices.

2.3.3 Horizontal scalability - multiple LPARs, multiple systems

This scenario is very similar to the multiple object server scenarios where each object server is running on a separate system. In this case, the library server and one or more object servers are installed in separate LPARs (logical partitions) on one or more physical systems.

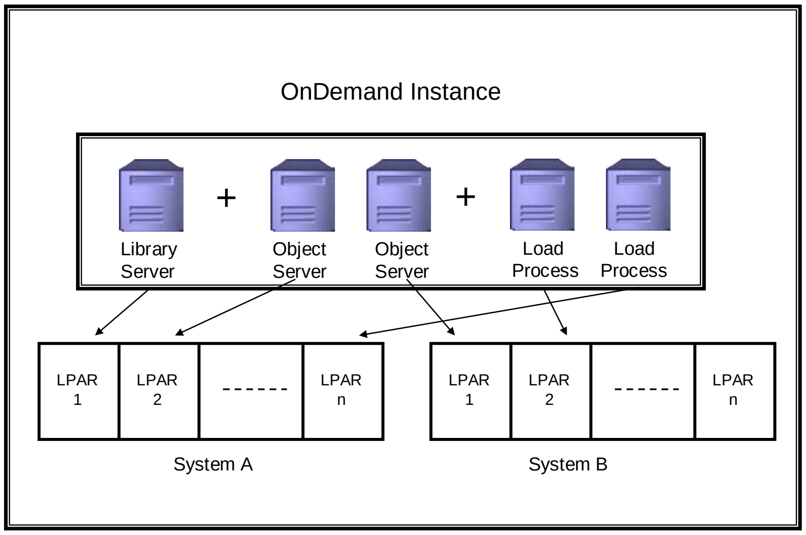


Figure 5. Horizontal & vertical scalability - multiple LPARs

This scenario is found in organizations that have large systems installed (such as AIX or z/OS) and have enough capacity available to support the organization's CMOD workload requirement. One of the advantages of this configuration is that it is possible to control the priority of work and computer resource distribution to each of the LPARs, such as the number of CPUs or processing priority (depending on the computer system/operating system architecture) allocated to each of the LPARs. So, for example, load jobs could be assigned a low priority during the day when the focus is on data retrieval and a high priority during the night when the focus is on data loading.

This setup supports horizontal scalability with the ability to use multiple technologies as appropriate. The only constraint is that clients must have access to all systems through TCP/IP.

2.3.4 Multiple server configuration rules

The following are a set of generalized rules to follow when configuring multiple CMOD servers. In all cases, you should refer to the appropriate CMOD documentation and/or contact CMOD Lab Services for additional guidance.

- Each CMOD server has its own set of configuration files.
- The parameters in all configuration files must be set such that all the servers are part of the same instance.
- The CMOD clients connect to the IP address listening port of the CMOD server (library server module).
- The documents are retrieved from the various object servers based on location information returned by the library server. This is transparent to the client systems.

The illustration below depicts this configuration type:

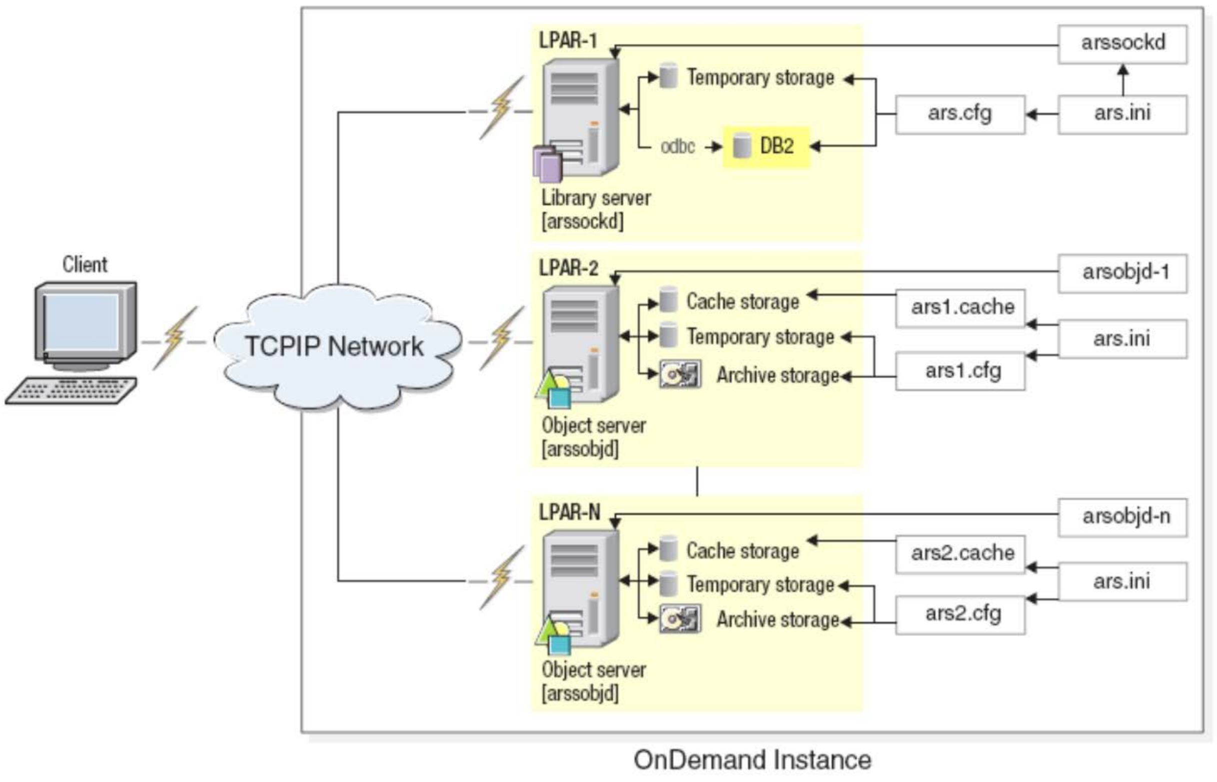


Figure 6. Multiple server configuration

2.4 High Availability

The concept of high availability roughly equates to a system and its data being available (accessible by users) almost all the time, 24 hours a day, 7 days a week, and 365 days a year. In reality, based on cost constraints, 100% availability is not a cost-effective reality today for the large majority of implementations; rather, it is a goal. The goal is to design and build systems that are highly available by minimizing both planned and unplanned outages that can be caused by single points of failure.

2.4.1 Redundant systems - all platforms

There are a variety of techniques employed on all platforms that can achieve near high availability. These techniques are based on creating as much redundancy as possible within the system and the data they include:

- **Preventing data loss -** employing various levels of RAID to store the data on disk
- **Duplicating the data -** by creating near real time copies of the data on "backup" devices that replace the online devices if they fail
- **Duplicate systems -** a duplicate system (hardware, software and data) is maintained (either locally or remotely) and when the main system fails users are automatically directed to the duplicates system
- **Network redundancy -** creating multiple paths through the network such that if one path (or router) fails the network continues to function

All of these techniques work well and provide various levels of near real-time high availability based on the degree to which the redundant systems are created and are kept in active-standby mode.

2.4.2 Multiple LPAR sysplex - z/OS

The z/OS operating system has high availability architecture built into it. A z/OS parallel sysplex is a tightly-coupled cluster of independent z/OS systems connected via a TCP/IP network. A cluster consists of from 2 to 32 independent systems that are locally or geographically dispersed. Communication between the z/OS systems in the sysplex is handled through the cross-system Coupling Facility (XCF). A z/OS parallel sysplex implementation provides the highest level of high availability in the industry. The figure below illustrates a CMOD implementation of a two-system highly available z/OS sysplex system.

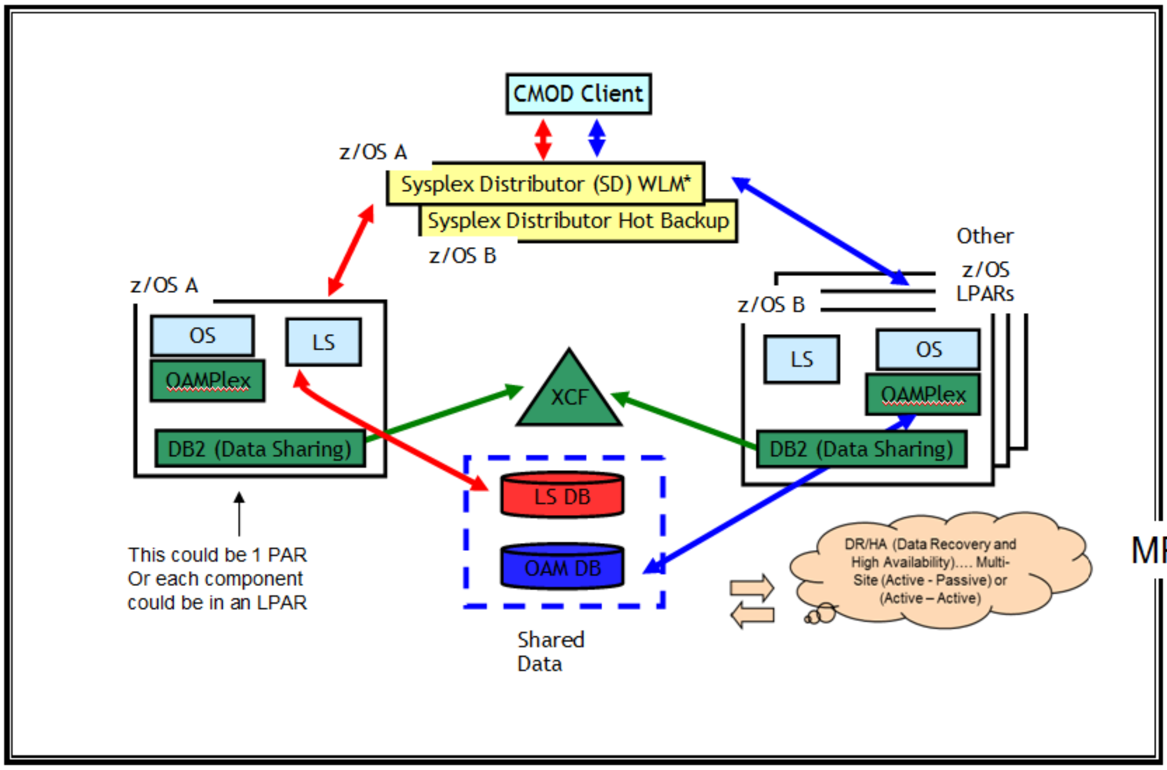


Figure 7. Scalability (parallel sysplex/multiple LPARs (z/OS))

As shown in the figure, z/OS system A contains a library server and an object server. These can be either combined in a single executable (most common z/OS implementation) or separated into two executables in which case they would be installed in separate LPARs. z/OS system B shows a multiple LPAR system with library/object servers installed in each of the LPARs.

Both of these systems (all LPARs and all instances of the CMOD server) access a single set of CMOD database tables via DB2 datasharing. They also access a single OAM archive system via an OAMPLEX. Not shown in the figure is the access to a single JES spool and a single set of HFS/zFS file systems. The term “single” is used to imply that the same set of data is available to all systems concurrently. Each of these single systems is composed of highly redundant components and therefore do not represent a single point of failure.

The z/OS parallel sysplex technology enables the CMOD servers to share the same configuration files, database, JES, HFS, and archive. For performance reasons, all HFS read/write directories that are used for temporary storage of data are configured as being unique to each CMOD server.

From a client perspective, the “cluster of CMOD servers” is a single IP address. Incoming client requests are received by the sysplex distributor/Work Load Manager (WLM). The WLM monitors the various systems in the parallel sysplex and will select the appropriate CMOD server to which to forward the request based on the current system workload and availability, such that the system that is more available (less busy) will receive the request.

2.5 High availability - IBM i

“PowerHA SystemMirror for i” is the integrated IBMi storage-based clustering solution for high availability and disaster recovery. Data and applications are deployed into storage pools (called independent auxiliary storage pools or IASPs). IASPs can be deployed using either internal or external storage. At any time the nodes in the cluster can switch roles and become either a primary or secondary node. PowerHA SystemMirror is designed and intended to be used for on demand role swap operations.

The Power Systems Capacity BackUp (CBU) offerings are designed to support your disaster recovery and high availability needs. The Capacity BackUp offerings recognize that true high availability or disaster recovery solutions require at least two systems. If one system is not available the other one "takes over". The CBU offering is designed to give your flexible and economic options for deploying business continuity operations.

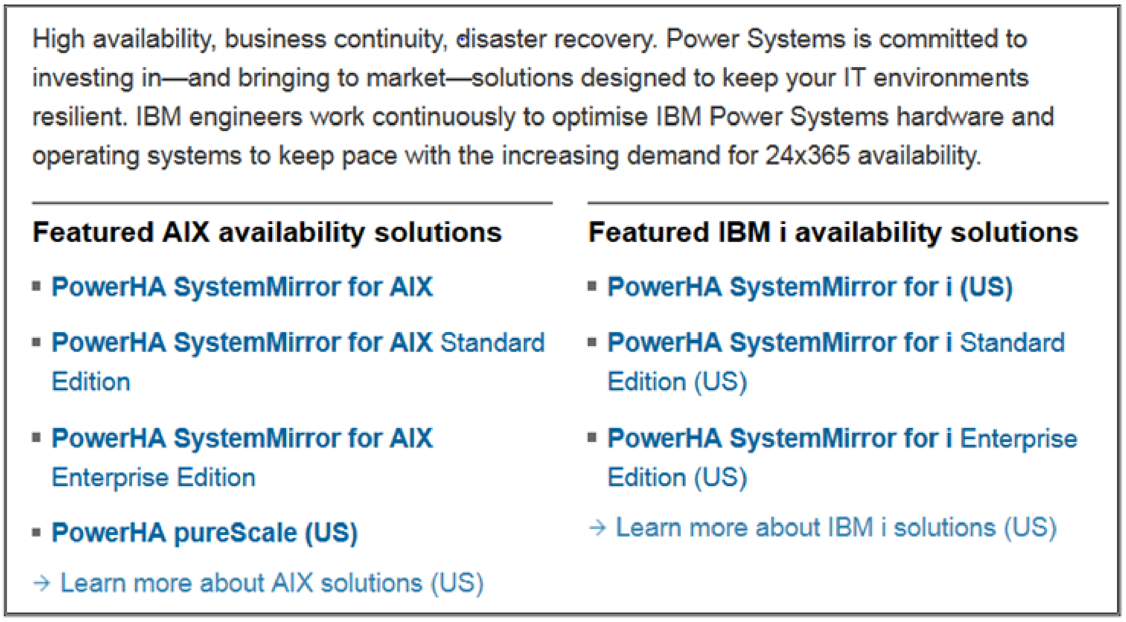
In a high availability environment on IBM i:

- Do not replicate the temporary IFS directories for your instance(s). For example, do not replicate /QIBM/UserData/OnDemand/QUSROND/TMP or /QIBM/UserData/OnDemand/QUSROND/PRTTMP where QUSROND is your instance name.
- Do not replicate the home directory for the user storing data. For example, if JOHNDOE is the name of the user profile that stores data into CMOD, do no replicate /home/JOHNDOE.
- Do not replicate the /tmp directory.

2.6 Process scalability and high availability – AIX and IBM i

The graphic below depicts the high availability solutions for UNIX (AIX) and IBM i. More details are available on the Web at:

http://www.ibm.com/systems/uk/power/software/availability



High availability, business continuity, disaster recovery. Power Systems is committed to investing in—and bringing to market—solutions designed to keep your IT environments resilient. IBM engineers work continuously to optimise IBM Power Systems hardware and operating systems to keep pace with the increasing demand for 24x365 availability.

Featured AIX availability solutions

- **PowerHA SystemMirror for AIX**
- **PowerHA SystemMirror for AIX** Standard Edition
- **PowerHA SystemMirror for AIX** Enterprise Edition
- **PowerHA pureScale (US)**

→ Learn more about AIX solutions (US)

Featured IBM i availability solutions

- **PowerHA SystemMirror for i (US)**
- **PowerHA SystemMirror for i** Standard Edition (US)
- **PowerHA SystemMirror for i** Enterprise Edition (US)

→ Learn more about IBM i solutions (US)

Figure 8. Featured scalability and availability solutions for AIX and IBM i

2.7 Performance, scalability, reliability, and availability summary

The architectural flexibility of CMOD allows you to select the “right sized” system based on your needs. A CMOD implementation can be scaled both vertically (by using larger and larger systems) and horizontally (by increasing the number of systems that are part of the CMOD instance).

Scalability and High Availability

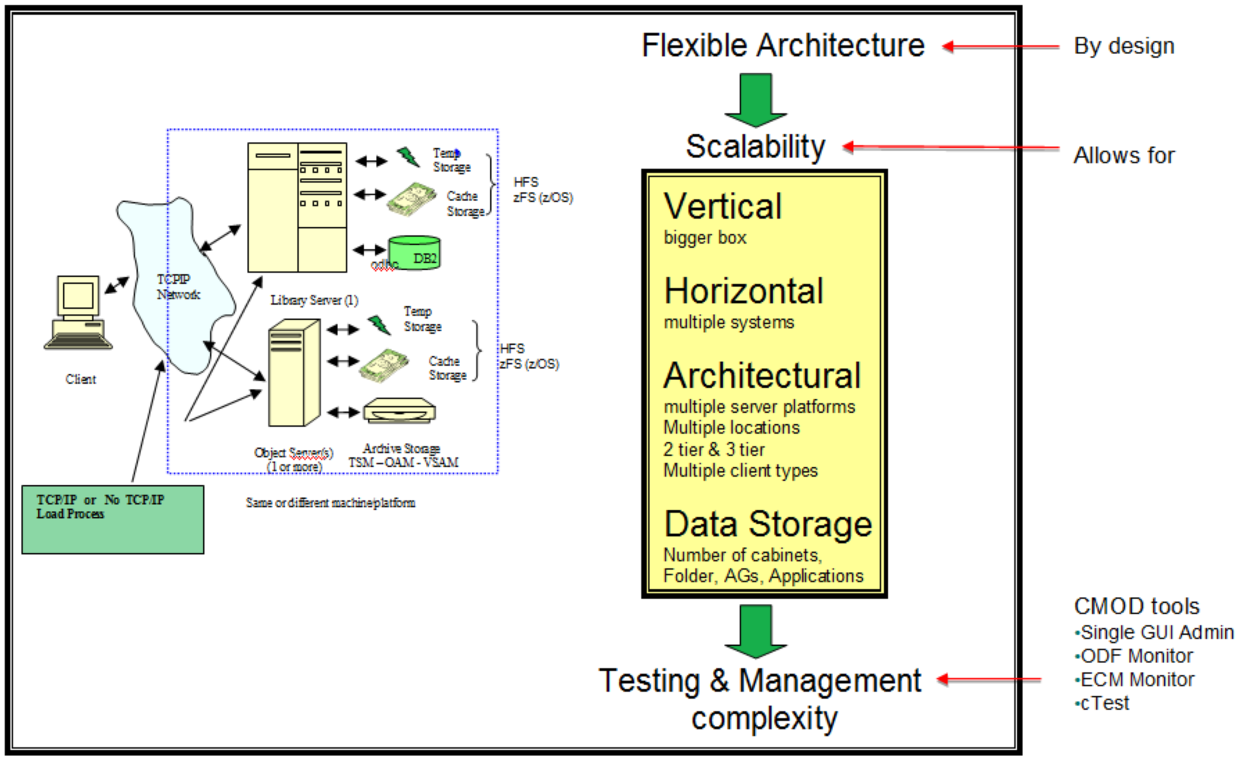


Figure 9. Performance, scalability, reliability and availability

A CMOD server can scale from a Windows server up to a cluster of z/OS systems. It is important to initially select an installation that:

1) is appropriate for your current workload in terms of:
    a. performance
    b. scalability
    c. reliability
    d. availability
2) can support your future growth requirements if you:
    a. increase the number of users accessing the system
    b. increase the quantity of data stored in the system
    c. change the types of data archived and/or the pre-processing requirements

3 Installation environment (serviceability)

3.1 Serviceability overview

Serviceability refers to the ease with which a system (or subsystem) can be maintained and repaired. Early detection via testing and monitoring of potential current or future problems is critical in this respect. Identified corrective action should cause little to no system downtime.

Serviceability in CMOD is achieved in the areas listed below, which are further discussed in the following sections.

- Installing and upgrading
- Testing
- Monitoring and tracing
- Support and documentation

Creating and maintaining a serviceability environment reduces operational costs and supports business continuity.

3.2 Installing and upgrading

Installing or upgrading CMOD (or any other software) is never done directly into a production environment. The new software is introduced and tested incrementally through a series of controlled environments. Typically there are four environments. These environments and their purposes are outlined below. Your particular implementation might vary slightly from what is described, but the concept of incremental software installation should be the same.

1) **Test -** The CMOD software is received and applied/installed into this environment. After the software is installed, basic regression testing is performed to ensure that the installation was successful and that the software functions correctly. Once everything appears to be satisfactory, the software is then copied to the Development environment.

2) **Development -** If you have implemented any exits, extensions or modifications to the CMOD software, this is where you would

- Verify that they are still valid for the current level of software; and
- Add or remove extensions or modifications based on the new features available in the CMOD software.

Items generally examined include exits, interfaces to other applications and interfaces to custom clients. Once everything appears to be satisfactory, the software is then copied to the QA environment.

3) **QA (Quality Assurance) -** This is where the majority of the testing occurs. At a minimum, you should:

- run a full set of regression tests, to ensure that the new release and code extensions function as expected.
- run a set of performance tests to ensure that the new release and the code extensions still provide performance equal to or greater than the previous release.
- tune the test environment. An understanding of this tuning needs to be reflected into your production environment.
- perform other testing, such as endurance testing, which refers to tests typically done to find out whether an application can withstand the processing load it is expected to endure for a long period.
- conduct all testing using production-like data.

The more testing that is done at this stage, the more you will be assured that you will not face any unexpected problems during production. Once everything appears to be satisfactory, the software is then copied to the production environment.

4) **Production -** This is the final destination for your new software. Based on the performance testing and environment tuning in the QA environment, you might also need to tune your production environment. If the

previous three steps were performed correctly, then you will have many happy users and few if any operational problems.

IBM recommends that you:

- Update to the latest CMOD code maintenance level.
- Consider whether system tuning parameters need to be changed as new releases of CMOD are installed. For example, at CMOD version 8.5, the AIX version of the code was modified from being process-based to being thread-based. This further improved the CMOD performance on AIX but in order to take advantage of this change, changes were required for AIX (for example ulimits). In CMOD version 9.5, the ulimit settings are displayed in the output of your ARSLOAD.
- Keep the client and server code at compatible code levels.
- Keep the library and object servers at the same code levels, if you are using multiple library and object servers (on one or more platforms).
- When updating subsystems (ex. DB2) or operating systems (ex. AIX), take into consideration that they also have their own performance tuning requirements and recommendations. This might include both tuning and hardware requirements.

3.3 Testing

With CMOD, you perform three basic functions:

- Index/Load data
- Retrieve data
- Expire data

Your testing should cover all three of these functions and any other functions that you may be performing (for example, Report Distribution or Full Text Search). Operationally, you also backup and (if needed) restore data. So, you should also test your backup and restore procedures.

In most cases, you will not be able to test in your production environment, so you should ensure that your testing environment:

- Matches the specifications of your production environment and if not, then the differences must be fully understood.
- The code tested is identical to what will be run in the production environment. Code refers to ALL code, including CMOD, exits, transforms, operating system, runtime, …
- The data is the same data.
- The metadata definitions (applications, application groups, folders) are the same.
- The number of rows in the application group data tables is the same.
- The indexes defined on the application group data tables are the same.
- If you use an external interface such as LDAP in production, then you need to include that in your tests.
- The number of users and number of load jobs is the maximum that you will encounter in your production environment.
- For performance testing, at the very least, you should compare the new code in your test environment to the existing code in production. This comparison should be on a function-by-function basis and at the maximum workload.

For performance testing, there are basically two types of testing:

- Extrapolation-based testing
- Simulation-based testing

These are illustrated in the figure below.

With extrapolation, each main function of the CMOD system is tested separately. This allows for a function-by-function analysis of the performance and an accurate determination of where any system degradation is occurring. It also simplifies identifying the cause of the degradation.

With simulation, a typical user's actions (logon through logoff) are simulated and then the test is run with different numbers of users to simulate the real world system response. You might need to create several typical users, one for each type of user on your system. Then you should run your simulation with the appropriate number of each type of user accessing the system concurrently.

To get an accurate picture of the behavior of your system, you should run both extrapolation and simulation tests. The simulation tests will indicate if the system is capable of sustaining your maximum workload with your transaction mix. The extrapolation testing will allow you to determine which transactions consume the most system resources based on your system configuration and tuning.

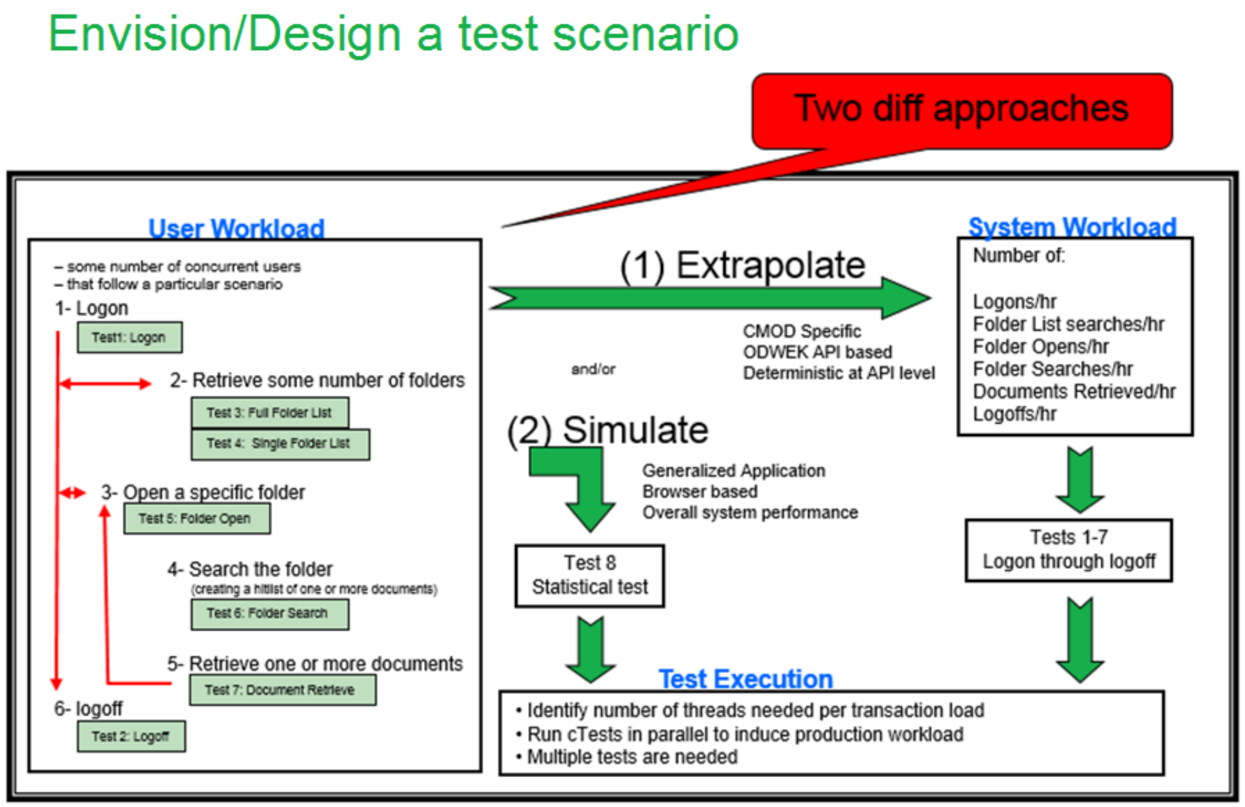
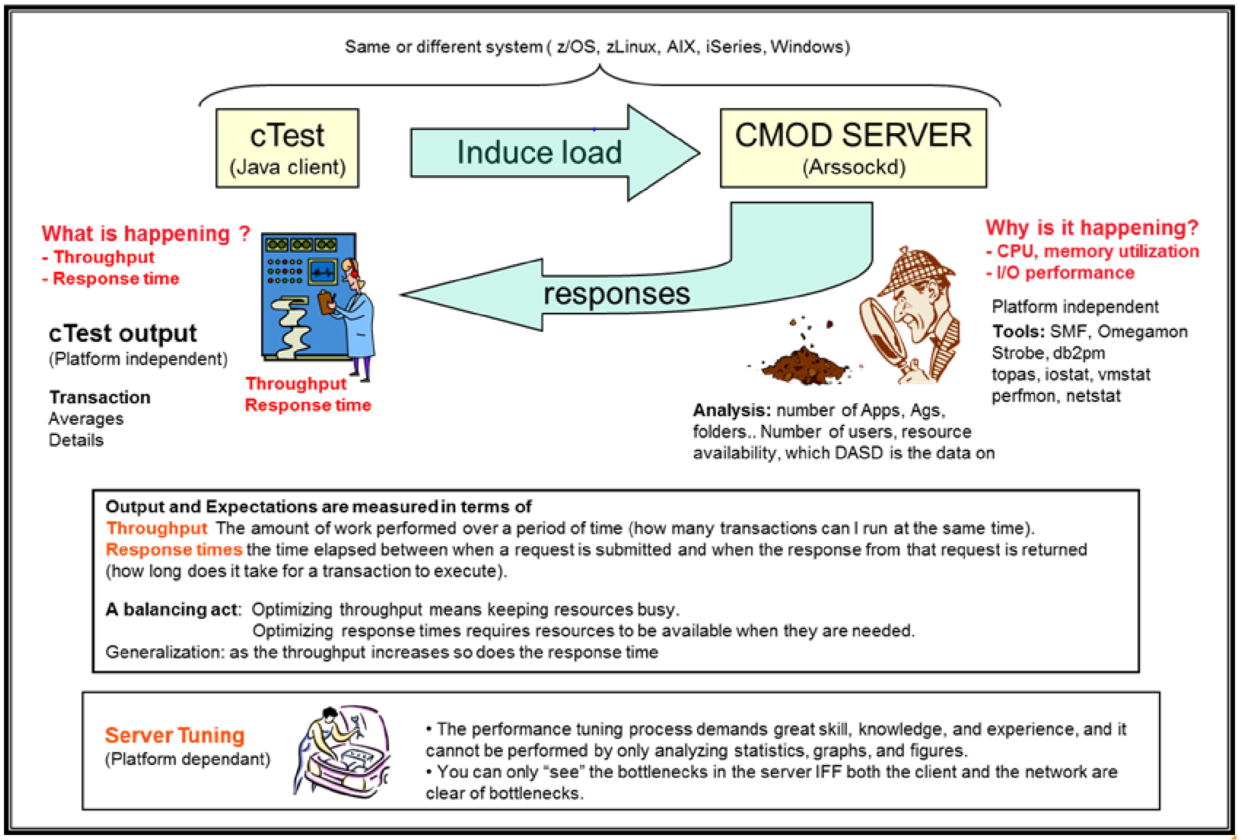

Envision/Design a test scenario
Two diff approaches
User Workload
– some number of concurrent users
– that follow a particular scenario
1- Logon
Test1: Logon
2- Retrieve some number of folders
Test 3: Full Folder List
Test 4: Single Folder List
3- Open a specific folder
Test 5: Folder Open
4- Search the folder
(creating a hitlist of one or more documents)
Test 6: Folder Search
5- Retrieve one or more documents
Test 7: Document Retrieve
6- logoff
Test 2: Logoff
(1) Extrapolate
CMOD Specific
ODWEK API based
Deterministic at API level
and/or
(2) Simulate
Generalized Application
Browser based
Overall system performance
Test 8
Statistical test
System Workload
Number of:
Logons/hr
Folder List searches/hr
Folder Opens/hr
Folder Searches/hr
Documents Retrieved/hr
Logoffs/hr
Tests 1-7
Logon through logoff
Test Execution
• Identify number of threads needed per transaction load
• Run cTests in parallel to induce production workload
• Multiple tests are needed

Figure 10. Envisioning and designing test scenarios

Performance testing will involve a test driver to generate the retrieval workload on the CMOD server. The test driver itself might be a bottleneck to the CMOD server performance. For example, if the test driver is only capable of creating, receiving and tracking 25 transactions per second and a peak transaction rate of 25 transactions per second is achieved during testing, one might mistakenly believe that the CMOD server is the cause. To avoid this problem in the CMOD lab, we use a customized test driver built upon the ODWEK APIs that has been demonstrated to work at extremely high transaction rates.

There are other factors besides the test driver that will limit the performance of the system. The figure below illustrates a methodology that you could follow to perform accurate performance testing and tuning.

Performance Tuning



Analysis: number of Apps, Ags, folders.. Number of users, resource availability, which DASD is the data on

Output and Expectations are measured in terms of
Throughput The amount of work performed over a period of time (how many transactions can I run at the same time).
Response times the time elapsed between when a request is submitted and when the response from that request is returned (how long does it take for a transaction to execute).

A balancing act: Optimizing throughput means keeping resources busy.
Optimizing response times requires resources to be available when they are needed.
Generalization: as the throughput increases so does the response time

- The performance tuning process demands great skill, knowledge, and experience, and it cannot be performed by only analyzing statistics, graphs, and figures.
- You can only "see" the bottlenecks in the server IFF both the client and the network are clear of bottlenecks.

Figure 11. Performance testing

The test is setup such that:

1. the test driver (cTest) generates a retrieval workload on the system.
2. the test driver is multithreaded and thus can send any number of specified requests to the CMOD server concurrently.
3. the test driver can be placed on a TCP/IP network so as to emulate the values observed by system users and/or can be placed on the CMOD server system so as to produce the maximum possible workload on the CMOD server.
4. the CMOD server provides responses back to the test driver.
5. multiple test drivers can be run in parallel, each simulating a different system workload (different user access pattern).

There are two points of measurement:

1. On the test driver side, you can see "what is happening". You are able to capture and record both the response time and throughput which are measured at predefined workloads.
    a. If users use the CMOD Windows client, then these values plus the workstation overhead will be similar to the values that the system users see on their workstations.
    b. If users are connected through a second tier such as the ODWEK APIs, WebSphere, or Web Services, then these values will be the values observed at that second tier. The values observed by the users will be the measured values plus the second tier overhead plus the network connection from the second tier to the users' workstations plus their workstation overhead.
2. On the CMOD server side, you can determine "why this is happening".
    a. You can use platform specific tools to measure system specific values. Items measured include CPU activity, memory consumption, and I/O activity.

b. You can also use subsystem specific tools or commands to monitor specific subsystems such as DB2, TSM, OAM and TCP/IP.

c. You can use CMOD monitoring tools such as the system log or ECM System Monitor to observe what is happening at the CMOD server level.

Analyzing the test result values obtained on the server side enables you to tune and add the appropriate resources that will allow you to reach the desired results on the test driver/client side for your specific workload pattern, environment and data.

3.4 Monitoring and tracing

Monitoring and tracing are two very distinct activities.

3.4.1 Monitoring

Monitoring is typically performed on an ongoing basis. You monitor your system to ensure that it is performing as intended and that there are no problems that are occurring due to a component failure or an unexpected occurrence (workload or other). So, by design, monitoring must be non-intrusive and must have minimal impact on resource consumption. The figure below depicts a group of methods that are used to monitor CMOD in a z/OS production environment. The concepts are the same on all computer platforms and the figure is thus described in generic terms.

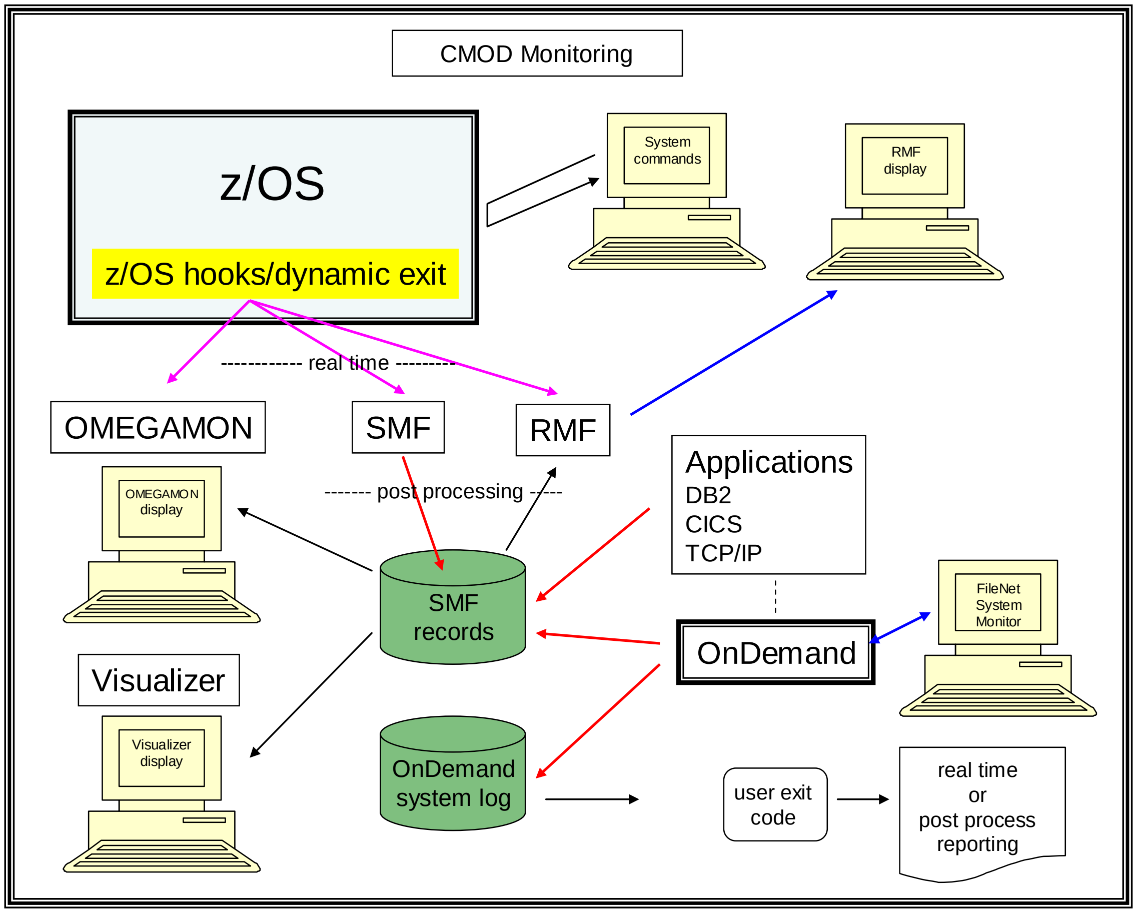


Figure 12. Monitoring in a z/OS environment

The monitoring functions can be described as:

- Monitoring the operating system: System level tools specific to your operating system are used (ex. OMEGAMON, RMF, svmon, vmstat, iostat, …)
- Monitoring the subsystems: Subsystem specific tools specific to your subsystem are used (ex. DB2PM)
- Monitoring CMOD: There are four general methods:
    - looking at values in the CMOD System Log
    - placing tooling in the exits to identify undesirable performance
    - using the ECM System Monitor
    - writing a specific piece of code that perform a known transaction every time slice and measuring the response to that transaction

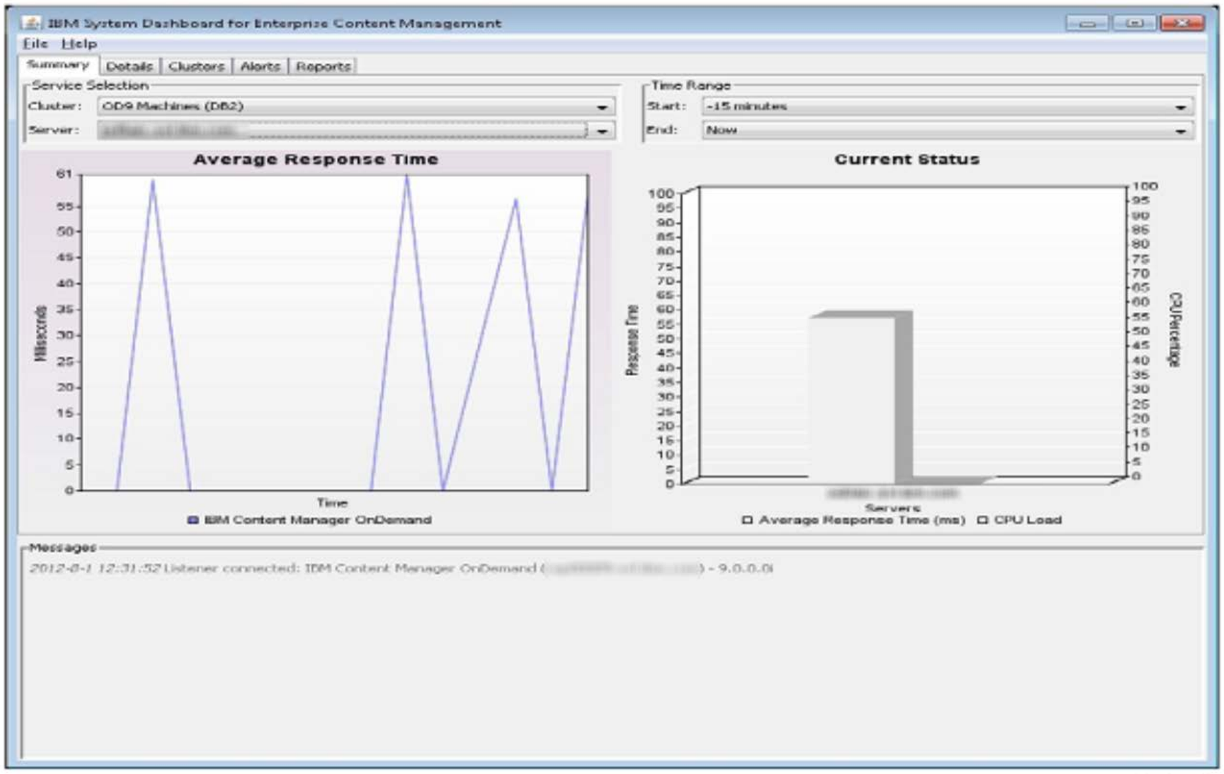
Automated responses to the monitored system can be defined by creating a trigger based on a value or combination of values from the monitoring steps.

3.4.2 ECM System Monitor

ECM System Monitor is a separately priced product which can monitor other programs. ECM System Monitor supports multiple ECM products such as Filenet and Content Manager version 8. ECM System Monitor support for CMOD became available with CMOD version 9. CMOD is integrated with ECM System Monitor in two ways:

- ECM System Monitor can monitor the CMOD System Log and report when errors occur. This is done through an ODWEK module provided to and shipped by ECM System Monitor.
- ECM System Monitor can also monitor the status of the CMOD server (ARSSOCKD program) via the new command line options to provide thread listings and “ping” status.

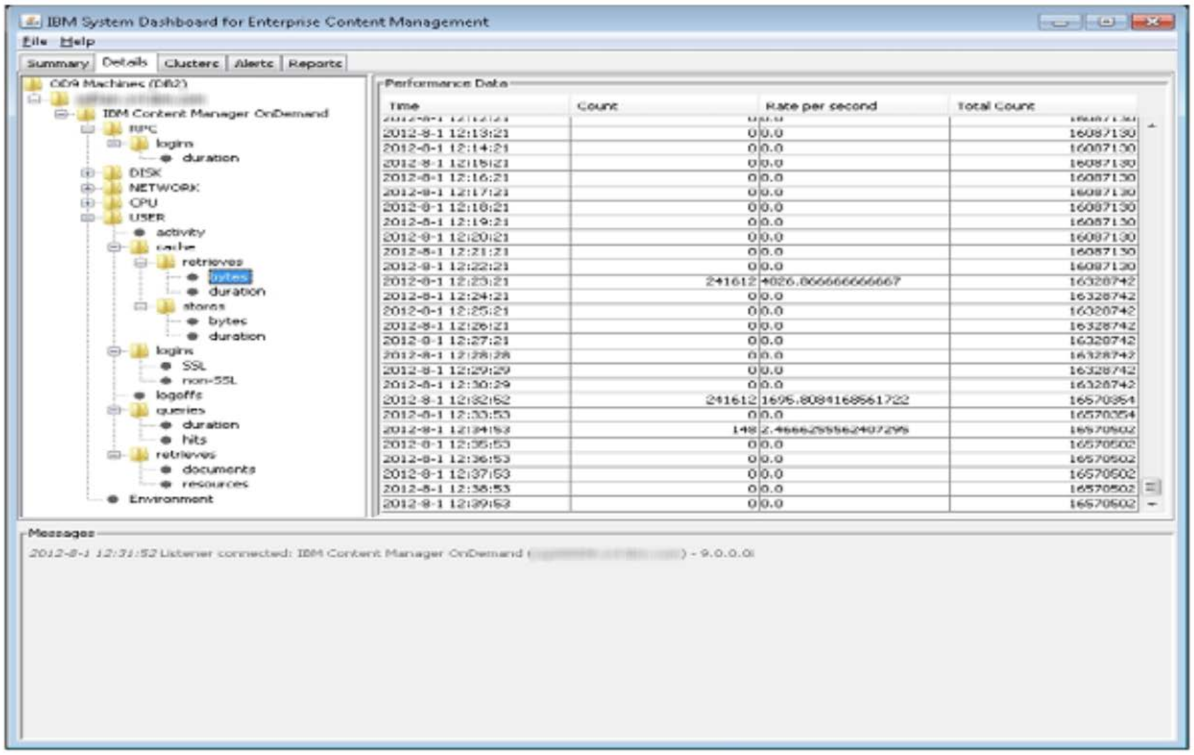
ECM System Monitor listener, which is directly implemented in the CMOD server program, can be attached to via a network port to relay information to a listening program, in this case, ECM System Monitor. The port is defined in the ars.cfg file (or registry section on Windows) for the appropriate server. The environment variable is ARS_LISTENER_PORT. If not defined, then the listener is inactive. If defined as 0, the default port of 32775 is used. The ECM System Monitor listener should only be enabled when needed. The following three figures show various monitoring screens produced by ECM System Monitor:



Cluster: OD9 Machines (DB2)
Server: [redacted]
Start: -15 minutes
End: Now

```
2012-8-1 12:31:52 Listener connected: IBM Content Manager OnDemand ([redacted]) - 9.0.0.0
```

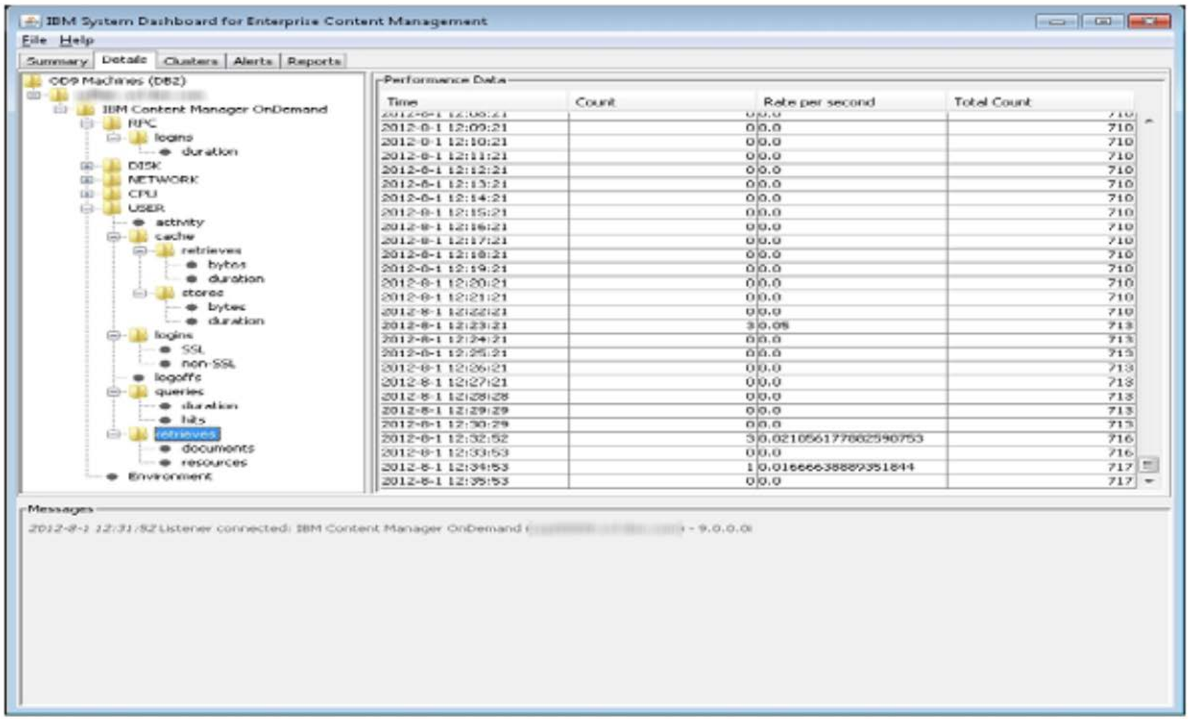
Figure 13. ECM System Monitor - Response Time



Time	Count	Rate per second	Total Count
2012-8-1 12:13:21	0	0.0	16087130
2012-8-1 12:14:21	0	0.0	16087130
2012-8-1 12:15:21	0	0.0	16087130
2012-8-1 12:16:21	0	0.0	16087130
2012-8-1 12:17:21	0	0.0	16087130
2012-8-1 12:18:21	0	0.0	16087130
2012-8-1 12:19:21	0	0.0	16087130
2012-8-1 12:20:21	0	0.0	16087130
2012-8-1 12:21:21	0	0.0	16087130
2012-8-1 12:22:21	0	0.0	16087130
2012-8-1 12:23:21	241612	4026.866666666667	16328742
2012-8-1 12:24:21	0	0.0	16328742
2012-8-1 12:25:21	0	0.0	16328742
2012-8-1 12:26:21	0	0.0	16328742
2012-8-1 12:27:21	0	0.0	16328742
2012-8-1 12:28:28	0	0.0	16328742
2012-8-1 12:29:29	0	0.0	16328742
2012-8-1 12:30:29	0	0.0	16328742
2012-8-1 12:32:52	241612	1695.8084168561722	16570354
2012-8-1 12:33:53	0	0.0	16570354
2012-8-1 12:34:53	148	2.4666255562407295	16570502
2012-8-1 12:35:53	0	0.0	16570502
2012-8-1 12:36:53	0	0.0	16570502
2012-8-1 12:37:53	0	0.0	16570502
2012-8-1 12:38:53	0	0.0	16570502
2012-8-1 12:39:53	0	0.0	16570502

```
2012-8-1 12:31:52 Listener connected: IBM Content Manager OnDemand ([redacted]) - 9.0.0.0
```

Figure 14. ECM System Monitor – Bytes Retrieved



Time	Count	Rate per second	Total Count
2012-8-1 12:09:21	0	0.0	710
2012-8-1 12:10:21	0	0.0	710
2012-8-1 12:11:21	0	0.0	710
2012-8-1 12:12:21	0	0.0	710
2012-8-1 12:13:21	0	0.0	710
2012-8-1 12:14:21	0	0.0	710
2012-8-1 12:15:21	0	0.0	710
2012-8-1 12:16:21	0	0.0	710
2012-8-1 12:17:21	0	0.0	710
2012-8-1 12:18:21	0	0.0	710
2012-8-1 12:19:21	0	0.0	710
2012-8-1 12:20:21	0	0.0	710
2012-8-1 12:21:21	0	0.0	710
2012-8-1 12:22:21	0	0.0	710
2012-8-1 12:23:21	3	0.05	713
2012-8-1 12:24:21	0	0.0	713
2012-8-1 12:25:21	0	0.0	713
2012-8-1 12:26:21	0	0.0	713
2012-8-1 12:27:21	0	0.0	713
2012-8-1 12:28:28	0	0.0	713
2012-8-1 12:29:29	0	0.0	713
2012-8-1 12:30:29	0	0.0	713
2012-8-1 12:32:52	3	0.021056177862590753	716
2012-8-1 12:33:53	0	0.0	716
2012-8-1 12:34:53	1	0.016666388893351844	717
2012-8-1 12:35:53	0	0.0	717

```
2012-8-1 12:31:52 Listener connected: IBM Content Manager OnDemand ([redacted]) - 9.0.0.0
```

Figure 15. ECM System Monitor – Total Retrieves

Troubleshooting and logging

When the listener is running, it will create log files in the directory specified by the ARS_TMP parameter in the ars.cfg file. The log files are named *instance*.pch*nnn* where *instance* is the name of your CMOD instance and *nnn* is a sequential number.

The listener monitors the following:

- login counts (SSL and non-SSL)
- logoff counts
- login durations
- activity counts
- queries (hits and durations)
- retrieve counts (documents and resources)
- cache stores and retrieves (bytes and durations)
- storage manager stores and retrieves (type, bytes, and durations)

3.4.3 Tracing

Tracing is typically performed in the event of a failure of some kind. If the failure is intermittent, it can be very hard to track down. If the failure always occurs, such as document number 20345 in report X always fails, then by tracing the events leading to the failure it is possible to determine the cause of the problem.

3.4.4 Application Group/Message Logging tab

CMOD system logging can be used for usage monitoring, chargeback, or troubleshooting. Since system logging involves writing all of the selected log messages to disk, you will incur an increase in both resource usage and response time. Logging increases both the amount of CPU used and the amount of I/O to disk. For this reason, you should select only the types of logging that you want performed for a particular application group. Depending on your system usage requirements, you might decide to:

- turn off all System Logging
- record a minimal amount of information (only that information that is needed for reporting functions)
- record all transactions
- record the log information to one or more external files using the System Log exit
- turn on system logging only while troubleshooting the system
- turn on system logging once every time period to sample the system usage patterns

3.5 Support and documentation

CMOD provides various levels of support and documentation that can be customized to meet your needs.

3.5.1 Support

Level 2 support - If you are experiencing problems with the CMOD software, then Level 2 support will be able to assist you with defect resolution and minor installation and configuration issues. IBM's software support organization is a network of centers with expertise across our broad product portfolio. The various support teams work together to provide you with the responsive software support that you require. For complex problems, they have specialized, skilled teams with access to the experts in our development labs. Therefore, you have access to the right level of IBM expertise when you need it -- no matter where they are located.

The L2 support team’s goal is to ensure your satisfaction each time you need to contact them for support by:

- Responding to your requests within targeted guidelines
- Providing ongoing communication regarding your problem status through problem resolution
- Taking ownership of your request for support
- Providing a defined escalation process when management assistance is needed
- Maintaining our commitment to continuous improvement of our service processes

Additional premium support levels are available, such as the Accelerated Value Program or IBM Services. Contact your IBM marketing representative for details.

CMOD Lab Services - ECM Software Services is a uniquely skilled services organization within the IBM Software Group. IBM provides deep subject matter skills and expertise to ensure each engagement is delivered following and sharing ECM best practices. Examples of CMOD services included in their engagements:

- Implementation services
- Conversion services
- Performance optimization services

The ultimate goal within ECM Software Services is to execute successful projects that are on time and within budget. They are responsible for blending customer, IBM and, when appropriate, partner resources into a team, and for ensuring that the team works together to effectively deliver services related to ECM products. Additionally, they are responsible for providing price estimates and contracts. They want your project to be a positive interaction with IBM, thus paving the way for more successful interactions with IBM in the future.

OnDemand Users Group (ODUG) - ODUG is a community group composed of current CMOD users and consultants. ODUG meets at least once a year in conjunction with the Information On Demand (IOD) annual conference described below. Members meet to exchange CMOD experiences, ideas, and issues, and help to identify potential future product enhancements.

Conferences - CMOD participates regularly in two sets of conferences.

- Information On Demand (IOD): This is the primary IBM Information Management conference. In the United States, it is held annually, currently in Las Vegas towards the end of October. It is also conducted in other geographies around the world during different times of the year.
- IBM Content 2015 (formerly UserNet): These are smaller regional ECM conferences. They are conducted throughout the year in various locations. The locations for IBM Content 2015 can be found on the web at: www.ibm.com/software/ecm/events/content2015/

More information about these conferences and other events, as well as their schedules can be obtained by searching the IBM website at www.ibm.com.

3.5.2 Documentation

The CMOD product has a wide variety of documentation available.

Knowledge Centers - The CMOD Knowledge Centers (formerly known as Information Centers) contain information about getting started, troubleshooting, downloading fixes, and many other CMOD common tasks. The links to the Knowledge Centers for CMOD and related products are listed in Appendix A of this document.

Documentation -

- Product manuals - With each new CMOD version, a new set of product manuals is created.
- Redbooks - Redbooks are created periodically and are focused on specific topics of interest.
- White papers, techdocs and technotes - These are shorter documents created between releases that include a variety of topics that need to be addressed in a timely manner.
- CMOD Newsletter - This quarterly publication includes information on the latest CMOD developments, various tips for using CMOD, and any issues that might have been identified.

4 CMOD system configuration

The CMOD system is configured through a set of configuration files. Customization of the CMOD objects such as user IDs, applications, application groups, folders, and storage sets is accomplished by using the OnDemand Administrator client. These objects can also be manipulated by using the batch administration facility (by using the ARSXML command). These three sets of configuration options are discussed in the following sections. The parameters discussed here are only those that relate to best practices. For a complete list of the parameters, refer to the appropriate CMOD documentation.

4.1 Configuration files

The primary, initial CMOD settings are located in and controlled by four configuration files and the OnDemand Administrator client. The configuration files are:

1) ars.ini - the server initialization file
2) ars.cfg - the server configuration file
3) ars.cache - the cache file definition file (Multiplatforms and z/OS)
4) ars.dbfs - the database file system definitions (Multiplatforms)

The settings for the configuration files are discussed in the following subsections. You should keep a current backup of your configuration files.

4.1.1 Server initialization file (ars.ini)

The ars.ini file contains information about all of the CMOD server instances that you have installed on your system. For each CMOD server instance, the ars.ini file contains a reference to the ars.cfg and ars.cache files that belong to that server. Each CMOD server instance must have its own unique ars.cfg and ars.cache files. The ars.ini file contains several parameters, a number of which are critical for configuring each instance.

In the ars.ini file, the instance name is identified with surrounding square brackets []. Each instance must have a unique instance name. Multiple instances can be specified in the ars.ini file. Parameters that belong to a particular instance are entered in the lines following the instance name.

HOST - Identifies the host name alias, IP address, or fully qualified host name of the system on which the library server is running. It should be an actual host name or IP address.

PORT - identifies the TCP/IP port number that the instance monitors for client requests. Each CMOD instance must have its own unique port number assigned to it.

SSL_PORT – identified the port number on the Content Manager OnDemand server dedicated to communicating with the SSL protocol. The SSL_PORT is optional. Other SSL parameters are required to setup SSL. If you decide to implement SSL refer to your CMOD documentation for further details.

SRVR_INSTANCE - identifies the name of the CMOD database. Each CMOD instance must have its own unique SRVR_INSTANCE assigned to it.

SRVR_OD_STASH - identifies the location of the stash file used by the instance. This parameter is new for version 9 and is optional, used only if you are using stash files. IBM recommends that you store your user IDs and passwords in the stash files for security purposes.

Note: When running any of the CMOD commands, if the instance name is not specified, then the default instance name of ARCHIVE (on Multiplatforms or z/OS) or QUSROND (on IBM i) is used, and the parameters related to the ARCHIVE or QUSROND instance (from the ars.cfg file) will be used by the command.

4.1.2 Server configuration file (ars.cfg)

The server configuration file (ars.cfg) contains several parameters that affect performance. The general parameters are listed here. Other parameters are listed in their relevant section. For example, parameters that effect load are listed in the **Data loading** chapter. Tablespace allocation parameters are listed in the **Database definitions** chapter.

You should refer to the CMOD product documentation for your version of the software for a complete description of these parameters.

Parameters common to all platforms

ARS_NUM_DBSRVR - This is the maximum number of threads that are concurrently opened between the CMOD library server and DB2. Typically, this is set to a number between 4 and 30. This number needs to be large enough to support all of the concurrent database requests from all users/clients and CMOD commands and daemons such as ARSLOAD, ARSDOC, ARSDB, ARSMAINT, and ARSADMIN. This number must not exceed the number of DB2 batch connections (MAXDBATS for z/OS and MAXAPPLS for Multiplatforms). In other words, the number of DB2 batch connections should be greater than the ARS_NUM_DBSRVR plus all other connections required by all DB2 applications that you have defined in your DB2 configuration.

To ensure you have set this number to a value that will support your processing, you can issue the ARSSOCKD command as shown below, where ARCHIVE is the name of your CMOD instance. The results should show one or more DB - Idle threads which will indicate that you have successfully supported all concurrent database requests with no queues or waits:

arssockd -I ARCHIVE -p

```
2014-10-02 08:08:20
PID       TID   START TIME           CPU           MEM    STYPE     USERID  INFO
27852898  -     2014-10-01 09:50:23  01:30.967822  48416  Program   -       ARCHIVE
27852898  1     2014-10-01 09:50:23  00:06.211602  -      Main      -       Accepting
27852898  515   2014-10-01 09:50:23  00:00.422849  -      Activity  -       2
27852898  1029  2014-10-01 09:50:23  00:01.381708  -      Message   -       Idle
27852898  1286  2014-10-01 09:50:23  00:15.917662  -      DB        -       Idle
27852898  1543  2014-10-01 09:50:23  00:17.344243  -      DB        -       Idle
27852898  1800  2014-10-01 09:50:23  00:17.962715  -      DB        -       Idle
27852898  2057  2014-10-01 09:50:23  00:17.076270  -      DB        -       Idle
```

Additionally, on z/OS you can issue the OMVS command as follows:

D OMVS,PID=xxxx Displays the USS status of an address space down to the thread level

```
D OMVS,PID=67110818
BPXO040I 10.53.44 DISPLAY OMVS 353
OMVS     000E ACTIVE          OMVS=(P3)
USER     JOBNAME  ASID        PID    PPID STATE   START      CT_SECS
ARSADMIN ARSSOCKD 003B   67110818      1 HFI   17.36.07  14953.031
  LATCHWAITPID=         0 CMD=ARSSOCKD
 THREAD_ID        TCB@     PRI_JOB USERNAME ACC_TIME SC  STATE
 1868340000000000 007FDC60 OMVS             455.547 SEL  FU
 TAG=ARSSOCKD: (ACCEPTING)
 18684C0000000001 007CC200 OMVS                .004 KIN JKOV
 1868580000000002 007C3E88 OMVS              92.130 RED JFOV  TAG=ARSSOCKD: LICENSE (CUR(5),
 AX(40))
 1868640000000003 007BF190                  140.226 PIP JYOV  TAG=ARSSOCKD: DB (IDLE)
 1868700000000004 007C3510                 3469.170 PIP JYOV  TAG=ARSSOCKD: DB (IDLE)
 18687C0000000005 007C36A8                 3478.902 PIP JYOV  TAG=ARSSOCKD: DB (IDLE)
 1868880000000006 007C3938 OMVS            3339.741 RED JFOV  TAG=ARSSOCKD: DB (IDLE)
```

ARS_TMP - This is the temporary work directory used by CMOD. It is used by multiple CMOD functions including loading data and importing migrated index data. For Multiplatforms and z/OS, it is a separate non-shared mount point. For z/OS, it is configured through bpxprmxx. For Windows servers, it is defined using the configurator program. You should verify that there is sufficient free space available. If you are running multiple load jobs in parallel, then each load job should be defined to have its own temp directory.

ARS_PRINT_PATH - This is the temporary storage directory for the server print function. On Multiplatforms and z/OS, for performance reasons, this should be a different mount point from the ARS_TMP directory. Also, you must make sure that there is enough space in the specified location to hold the print files for the maximum number of concurrent print requests that the server will handle.

Requirement: If the installation has multiple CMOD servers (also known as ARSSOCKD processes on Multiplatforms and z/OS), then each server must have its own ars.cfg file, its own ARS_TMP directory, and its own ARS_PRINT_PATH.

4.1.3 Cache file definitions (ars.cache) - Multiplatforms and z/OS

The cache file systems are defined in the ars.cache file. For a distributed library/object server system, configure one copy of the ars.cache file on each server that is part of the CMOD system.

If there are multiple file systems in the ars.cache file, CMOD uses the file system with the greatest amount of free space to store the objects.

The cache file system can consist of one or more caches. Increasing the number of cache file systems allows CMOD to distribute data storage among them.

4.1.4 Database file system definitions (ars.dbfs) - Multiplatforms

This file is used only by Multiplatforms and is discussed in the **Database definitions** chapter.

4.2 OnDemand Administrator client

The OnDemand Administrator client allows for the customization of the CMOD instance objects. This customization affects the efficiency of the storage, retrieval and expiration of reports and documents. There exists a wide variety of storage and retrieval patterns which are based on organizational needs and requirements. These patterns vary from “store forever, retrieve rarely” to “store for a short period and retrieve frequently.” The OnDemand Administrator client parameters are discussed throughout the rest of this document based on the functionality that they provide.

4.3 Batch administration – ARSXML

You can use an XML interface to import and export administrative objects from and into a CMOD system. This XML interface expands the functionality and enables you to export all administrative objects into a single XML file, and later import them into the same CMOD system or another system. You can also create an XML file manually through a user application or Web interface according to the specifications defined in the CMOD schema file, and import it into the system.

Object order

In earlier versions of ARSXML, objects could be listed in any order within a single XML input file. While this allowed flexibility in how input files were built, it meant that in some scenarios ARSXML would have to process some objects out of order. Beginning with version 8.5, a strict object order is now enforced. The required object order is:

- systemParameters
- user
- group
- printer
- storageSet
- applicationGroup
- application
- folder
- cabinet

- hold
- odfRecipient
- odfRecipientList
- odfEmailTemplate
- odfReportId
- odfDistribution with odfReportBundle(s)

This new order allows ARSXML to process objects as they are encountered in the input file. To bring your existing input files into compliance, you must rearrange the objects in the input file to match the order listed above. Placing objects out of order can cause an error during processing.

Native application

CMOD version 8.5 includes an updated version of ARSXML. The ARSXML command is now a native application and no longer requires the use of the Xerces Java parser, resulting in better performance. The Java Runtime Environment requirement has also been removed.

Schema changes

The schema definition for ARSXML was modified in version 8.5 to address issues related to performance, error detection, and error reporting. These enhancements might require changes to your XML input files. If you receive errors when processing your XML input files, check the CMOD schema file to make sure your XML still follows the rules defined in the schema file.

New validate function

A new ARSXML action was added to help determine if your input files are valid according to the new schema definitions. To check your input file, run this command:

```
arsxml validate -i input_file
```

This command will check the conformance of the file as it relates to the schema definitions.

5 Database definitions

CMOD provides an administrative client that performs system administrative tasks, such as defining users, application groups, application, folders, and cabinets to the CMOD system.

There are two ways to define a report to CMOD:

- Adding a separate application group, application, folder and cabinet (if desired) by using the OnDemand Administrator client
- Using the Report Wizard

Although the Report Wizard function is not new to CMOD, you may not know about it if you are new to CMOD or if you have not yet discovered the Report Wizard toolbar button on the main window of the Administrator client.

The Report Wizard defines a report to CMOD by combining the tasks of adding an application group, an application, and a folder into one task. Information for the application, application group, and folder is gathered by answering a series of questions and by using the graphical indexer to define the indexer parameters, the database fields, and the folder fields.

Start the Report Wizard by clicking the Report Wizard toolbar button located on the main window of the OnDemand Administrator client, as shown below.

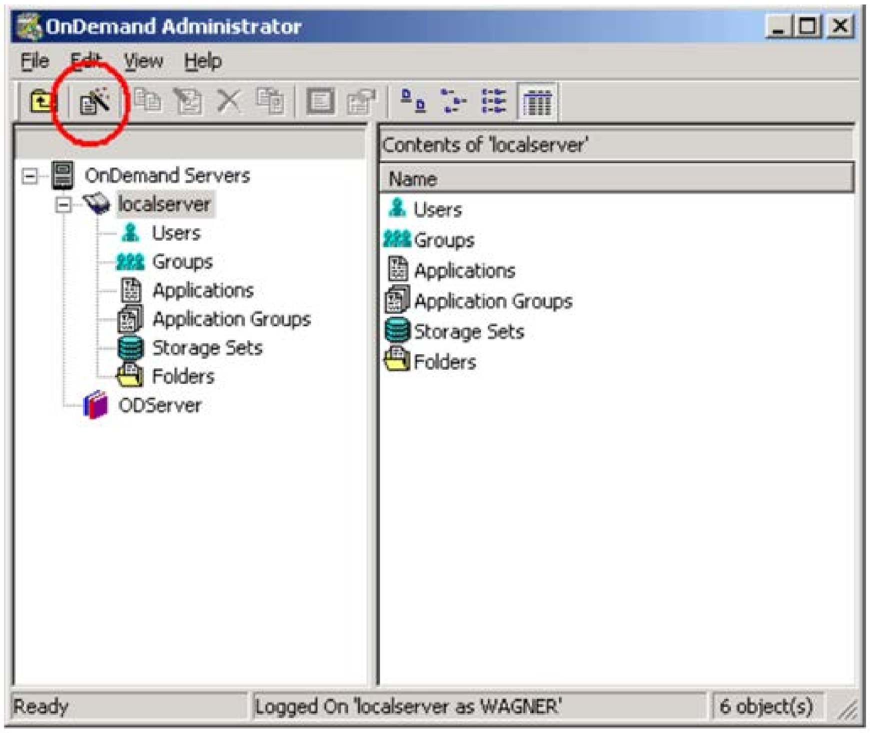


Figure 16. Location of Report Wizard toolbar button

5.1 OnDemand Administrator client - application group parameters

Application groups define how the index data located in the application group data tables is created and managed by the CMOD system.

5.1.1 Application Group/Field Information tab

Segment field - A segment field is a date or date/time field that CMOD uses to limit the number of tables that it searches to complete a query. If the **Expiration Type** is **Segment**, then CMOD also uses the segment field to determine when to delete data from the application group.

Segmenting your data provides the best query performance. IBM recommends that you define a segment field for each of your application groups. Limiting a query to a specific table or set of tables significantly improves the performance of queries for application groups that contain large amounts of data.

5.1.2 Application Group/General tab/Advanced button

CMOD can partition an application group data table into multiple segments. This partitioning is done by size and is controlled by the **Maximum Rows** field which specifies the maximum number of rows (of CMOD indexes) that will be stored in an application group data table segment before it is closed and a new application group data table segment is created for the same application group. The ideal number depends on your particular application. The maximum rows value should be chosen such that your table sizes are based on your search and expiration criteria. You want to avoid having too many tables for a single application group since this causes a query of the application group to span multiple tables (based on the search restriction, for example date range), but you do not want to have tables that are so large that maintenance (runstats) and backups of these tables take too long. In most cases, the value chosen is between 10 million to 200 million rows. But the requirements are customer and Application Group specific and you might find that in your case setting lower than 10 million or higher than 200 million works best.

Multiple loads per database table - Each time that you load a report into the application group, CMOD adds the index records to the application group database table that is currently open. When the open database table reaches the **Maximum Rows** value, CMOD closes it and opens a new table. Choose this database organization when the reports in the application group are part of a logical sequence of information that contains a unique identifier, such as an invoice number. Based on the report type, a report might generate from one to tens of thousands of indexes. A table can hold millions to hundreds of millions of indexes. **Multiple loads per database table** is the recommended option.

z/OS and IBM i only: You can select the **Single table for all loads** check box if you want to create one database table for each application group. This option is most frequently used when you load a ‘known’ amount of data. In this case the table size must be managed by the DB2 administrator. If you select this option, the **Maximum Rows** field is not applicable and is removed from the tab. For z/OS, the table will be created by CMOD with a default size of ten million rows unless you use the ARSMVS_NOMAXROWS_* parameters in the ars.cfg file.

Multiple field index - You can improve the performance of complex folder searches when an end-user normally enters multiple search criteria. You should use the standard Structured Query Language (SQL) performance tools available for your system to assist you in making the determination that a multiple field index will improve your CMOD search performance. One available resource for IBM i is a redbook titled “SQL Performance Diagnosis on IBM DB2 Universal Database for iSeries, SG24-6654”. This redbook helps you understand the basics of identifying and tuning the performance of SQL statements using DB2 for i.

Annotation flags in document database - If you use annotations and retrieve documents often, IBM recommends that you set this value to Yes. Setting the value to Yes causes CMOD to set the annotation flag in the database when a user adds an annotation to a document. When an annotation exists for a document, the client displays a note icon next to the document in the document list. If you set this value to No, then see the Folders/General tab for further details.

Query using parameter markers - A parameter marker is a placeholder for host variables in a dynamic SQL statement. Select this check box to use parameter markers. Clear this check box to turn off the parameter markers when you perform database queries.

In most cases, selecting **Query using parameter markers** improves the performance of the SQL query processing. For Multiplatforms and z/OS, in DB2 version 9, the REOPT(AUTO) can specify whether dynamic SQL queries with predicates that contain parameter markers are to be automatically re-optimized when DB2 detects that one or more changes in the parameter marker values renders dramatic selectivity changes. The newly generated access path replaces the current one and can be cached in the statement cache. Consider using this functionality especially when queries are such that a comparison can alternate between a wide range and a narrow range of unknown host variable values. IBM recommends that you query using parameter markers.

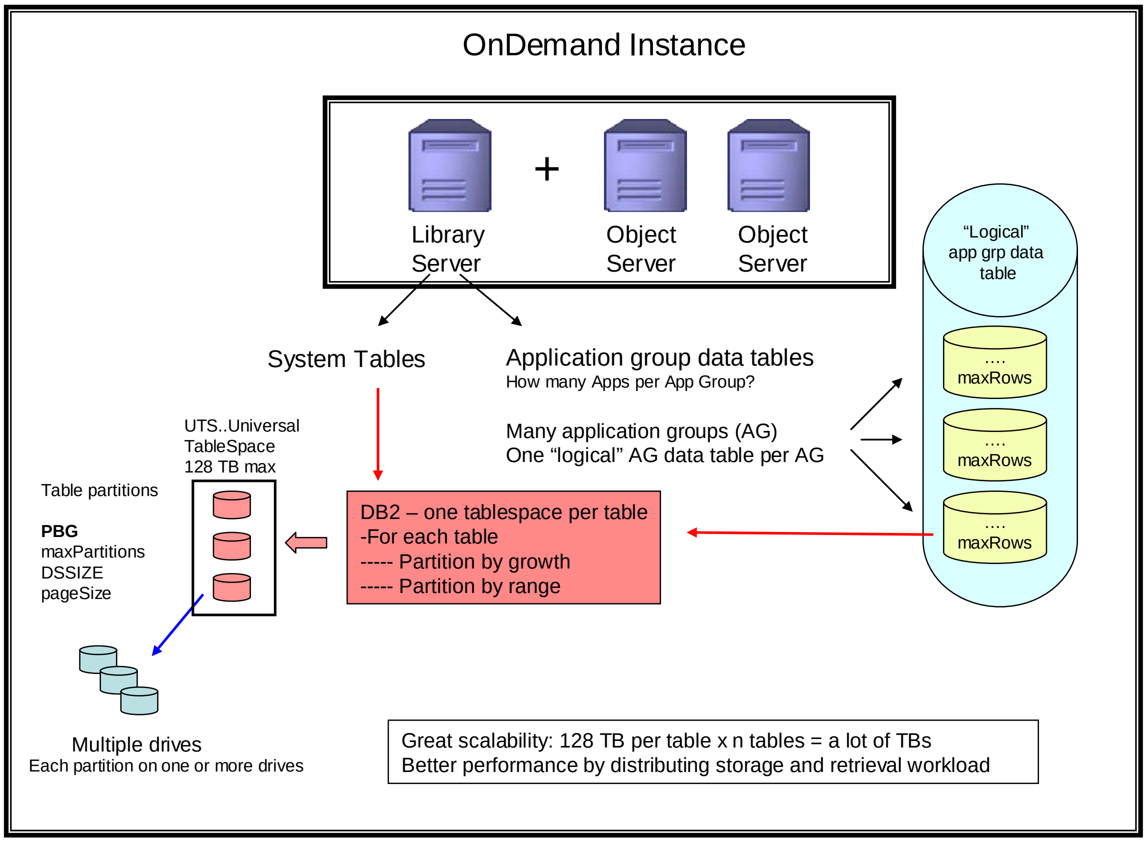
5.2 Database definitions – Multiplatforms and z/OS

The CMOD database contains both the CMOD system tables and the application group data tables (index tables). Database creation on Multiplatforms is controlled by the CMOD server. Database creation on z/OS is controlled externally to the CMOD server.

When CMOD creates a tablespace, table, or index tables that will be used to store application index data, you can invoke an external exit routine by defining the ARS_DB_TABLESPACE_USEREXIT parameter in the ars.cfg file. For the sample ARSUTBL, you would specify the following statement in the ars.cfg file at version 9.0:

```
ARS_DB_TABLESPACE_USEREXIT=/opt/IBM/ondemand/V9.0/bin/exits/arsutbl
```

Details on this exit and its usage can be found in the CMOD installation guides for Multiplatforms or z/OS.



OnDemand Instance
Library Server
+
Object Server
Object Server
System Tables
Application group data tables
How many Apps per App Group?
Many application groups (AG)
One "logical" AG data table per AG
"Logical" app grp data table
.... maxRows
.... maxRows
.... maxRows
UTS..Universal TableSpace 128 TB max
Table partitions
PBG
maxPartitions
DSSIZE
pageSize
DB2 – one tablespace per table
-For each table
----- Partition by growth
----- Partition by range
Multiple drives
Each partition on one or more drives
Great scalability: 128 TB per table x n tables = a lot of TBs
Better performance by distributing storage and retrieval workload

Figure 17. Database, tablespace and table creation

5.3 Database definitions – Multiplatforms

A CMOD server instance running on a Multiplatforms system stores all its data in a single database. When a new table is created, CMOD will search the defined tablespaces for the tablespace with the most free space and will create the table in that tablespace.

The ars.dbfs configuration file lists the file systems on the library server that can be used by the database manager to maintain index data in tablespaces. The rules for using tablespace file systems are:

- You should store only CMOD application group data in the tablespace file systems.
- You should define a minimum of two tablespace file systems.

- Each tablespace file system should be a separate mount point, on a separate drive connected through a separate I/O card.
- In general, the more tablespace file systems that you define, the better for performance and recovery.
- You should allocate equal amounts of disk space to each tablespace file system. If you increase the amount of space in one tablespace file system, you should increase the amount of space in the other tablespace file systems by an equal amount.

The following parameters are specified in the ars.cfg file:

ARS_DB_TABLESPACE - The name of the tablespace for the CMOD system tables. The value of this parameter must match an existing tablespace name in the database. You must have created the tablespace in DB2 or Oracle.

ARS_DB_PARTITION - Determines whether you can partition the database across nodes or systems. By default, you cannot partition the database. If the database manager product that you are using with CMOD supports partitioning, then you can specify that you want to partition the database by changing the value of this parameter to 1 (one). CMOD supports partitioning with DB2 Extended Enterprise Edition only. To store application group index data in partitions, your application groups must specify a partition field. The ARS_DB_PARTITION parameter is ignored on object servers.

5.4 Database definitions – z/OS

A CMOD server instance running on a z/OS system can store all its data in multiple tablespaces within multiple databases.

Each application group data table segment is created in its own tablespace. The tablespace is created within the database (identified by the database name) that is defined for that application group.

The **Database Name** (located on the Application Group/General tab/Advanced button) specifies the name of the database in which the application group data table/tablespace is created. For installations where a large number of application group data tables/tablespaces will be created, it is advantageous to distribute the created tablespaces amongst two or more databases. You should take into account future potential growth requirements as well as current requirements. If you define a **Database Name**, you must create the database using DB2 facilities prior to storing any CMOD data in the application group.

ARS_DB_RUNSTATS_DISABLE - Specifies that all runstats requests are to be disabled. Valid values are 0 or 1. If a value of 1 is specified, Runstats will not be run during table creation, ARSDB specified with -s or ARSMAINT specified with -r. The default is ARS_DB_RUNSTATS_DISABLE=0.

5.4.1 CMOD database tablespace allocations - z/OS

By default, the primary and secondary allocations for tablespaces created for the application group are based on the **Maximum Rows** setting for the application group. This setting is located in the Database Information window, which opens when you click Advanced on the General page for the application group.

If none of the parameters described below are specified, then the total amount of space required to contain 120 percent of the specified number of rows is determined. The PRIQTY of the tablespace is 50 percent of that total. The SECQTY is set to 12.5 percent of the total.

The following parameters (defined in the ars.cfg file) allow you to explicitly override and control the tablespace allocations.

For application groups with maximum rows specified

ARSMVS_MAXROWS_PRIQTY - This parameter specifies the PRIQTY that will be used for the CREATE TABLESPACE. If not specified, the value calculated from the maximum rows will be used.

ARSMVS_MAXROWS_SECQTY - This specifies the SECQTY that will be used for the CREATE TABLESPACE. If not specified, the value calculated from the maximum rows will be used. This parameter is not honored unless the ARSMVS_MAXROWS_PRIQTY parameter is also specified.

ARSMVS_MAXROWS_INDEX_PRIQTY - This specifies the PRIQTY that will be used for the CREATE INDEX. If not specified, the value calculated from the maximum rows will be used.

ARSMVS_MAXROWS_INDEX_SECQTY - This specifies the SECQTY that will be used for the CREATE INDEX. If not specified, the value calculated from the maximum rows will be used. This parameter is not honored unless the ARSMVS_MAXROWS_INDEX_PRIQTY parameter is also specified.

For application groups with Single table for all loads checked

ARSMVS_NOMAXROWS_INDEX_PRIQTY - This specifies the PRIQTY that is used for the CREATE INDEX action. If not specified, a value is calculated as if a maximum number of rows of 10 million had been specified.

ARSMVS_NOMAXROWS_PRIQTY - This specifies the PRIQTY that is used for the CREATE TABLESPACE action. If not |specified, a value is calculated as if a maximum number of rows of 10 million had been specified.

ARSMVS_NOMAXROWS_INDEX_SECQTY -This specifies the SECQTY that is used for the CREATE INDEX action. If not specified, a value is calculated as if a maximum number of rows of 10 million had been specified. This parameter is not valid unless ARSMVS_NOMAXROWS_INDEX_PRIQTY is also specified.

ARSMVS_NOMAXROWS_SECQTY - This specifies the SECQTY that is used for the CREATE TABLESPACE action. If not specified, a value is calculated as if a maximum number of rows of 10 million had been specified. This parameter is not honored unless ARSMVS_NOMAXROWS_PRIQTY is also specified

ARSMVS_BPOOL_INDEX - Specifies the buffer pool to use for application group data table indexes. If not specified, the buffer pool associated with the CMOD database is used. For example: ARSMVS_BPOOL_INDEX=BP3

ARSMVS_BPOOL_TSPACE - Specifies the buffer pool to use for application group data tablespaces with a row length less than 1024. If not specified or the row length is larger than or equals to 1024, the buffer pool associated with the CMOD database is used. For example: ARSMVS_BPOOL_TSPACE=BP2

ARSMVS_TABLESPACE_COMPRESS - This parameter indicates that CMOD is to include a COMPRESS clause of the CREATE TABLESPACE action with the value YES or NO for the application group data tables. Valid values are YES or NO. The COMPRESS clause is omitted if an invalid value is specified or if the parameter is absent.

ARSMVS_TABLESPACE_TRACKMOD - This parameter indicates that CMOD is to include a TRACKMOD clause of the CREATE TABLESPACE action with the value YES or NO for the application group data tables. Valid values are YES or NO.

For more information about specifying the above parameters, see the DB2 for z/OS: SQL Reference.

5.4.2 CMOD tablespace creation exit – Multiplatforms and z/OS

The CMOD tablespace creation exit allows an installation to take action when CMOD creates a tablespace, table, or index tables that will be used to store application index data. The exit is not called for the CMOD system tables. The CMOD System Log is an application group data table and treated as such. For table and index creation, the installation can alter the SQL that will be used to create the table or index.

CMOD specifies the application group indexes as single columns. For example, a name and date field would be two indexes. One of those indexes could be a clustered index.

Prior to server version 8.4.1, the CMOD system did not directly support the creation of composite indexes. Composite indexes were created by means of the tablespace creation exit. This exit allowed for the creation of “other” indexes during the creation of application group tables spaces.

Beginning with server version 8.4.1, the CMOD system does support the creation of composite indexes. It is still possible to use the index creation exit for customized index creation.

The following figure shows an overview of the tablespace creation exit.

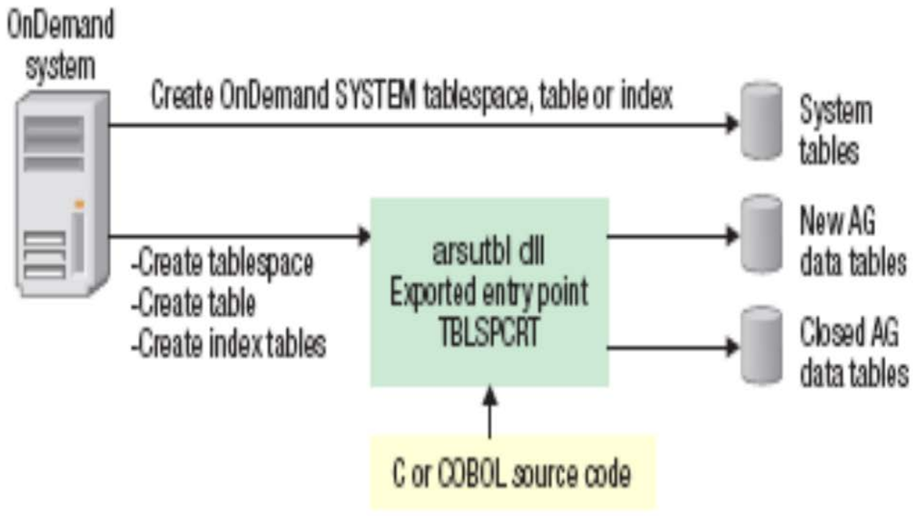


Figure 18. Tablespace creation exit

The tablespace exit is called multiple times each time a table is created.

Action 1 – for tablespace creation if tablespaces are used
Action 2 – for the actual table creation
Action 3 – for each index creation
Action 4 – for any additional work
Action 5 – for tablespace deletion if tablespaces are used
Action 6 – for table deletion
Action 7 – for index deletion
Action 8 – for table alteration

Refer to the Content Manager OnDemand Configuration Guide for further information.

5.4.3 Placing the CMOD system tables in separate tablespaces - Multiplatforms and z/OS

In earlier versions of CMOD, all of the CMOD System tables were in a single tablespace. A change was made so that each CMOD System table is now created in its own tablespace. This allows many tables to be placed in a 4K buffer pool, and allows row-level locking to be used for tables to allow for better concurrency when using those tables.

If you currently have all the CMOD system tables in a single tablespace, you do not need to migrate your tables to multiple tablespaces. If you wish to migrate, follow these steps:

1. Shutdown the CMOD server.
2. Backup the ARSDBASE database (or the database associated with the server you are trying to migrate).
3. Use the /usr/lpp/ars/bin/arsdb -x to export the CMOD system tables.
4. DROP TABLESPACE ARSTSPAC (or the tablespace associated with the server you are trying to migrate).
5. Run the ARSTSPAC job.
6. Run /usr/lpp/ars/bin/arsdb -c to create the tables in the new tablespaces.
7. Run /usr/lpp/ars/bin/arsdb -i to import the CMOD system tables.

If you choose to migrate, you should examine any RUNSTATS jobs you use to ensure that the new tablespace names are used. The tablespaces associated with each table are as follows:

Table	Tablespace
ARSAG	ARSAGT
ARSAGFLD	ARSAGFLT
ARSAGFLDALIAS	ARSAGFAT
ARSAGINDEX	ARSAGIDT
ARSAG2FOL	ARSAG2FT
ARSAGPERMS	ARSAGPET
ARSANN	ARSANNT
ARSAPP	ARSAPPT
ARSAPPUSR	ARSAPPUT
ARSCAB	ARSCABT
ARSCAB2FOL	ARSCABFT
ARSCABPERMS	ARSCABPT
ARSCFSODWORK	ARSCFSWT
ARSFOL	ARSFOLT
ARSFOLFLD	ARSFOLFT
ARSFOLFLDUSR	ARSFOLUT
ARSFOLPERMS	ARSFOLPT
ARSGROUP	ARSGROUT
ARSHOLD	ARSHLDT
ARSHOLDMAP	ARSHLDMT
ARSHOLDPERMS	ARSHLDPT
ARSHOLDWORK	ARSHLDWT
ARSLOAD	ARSLOADT
ARSNAMEQ	ARSNAMET
ARSNODE	ARSNODET
ARSPRT	ARSPRTT
ARSPRTOPTS	ARSPRTOT
ARSPRTUSR	ARSPRUST
ARSRES	ARSREST
ARSSEG	ARSSEGT
ARSSET	ARSSETT
ARSSYS	ARSSYST
ARSUSER	ARSUSERT
ARSUSRGRP	ARSUSRGT
ARSUSRGRPID	ARSUSGIT

Refer to the Content Manager OnDemand Administration Guide for your platform/release to obtain the list of system tables that apply to your installation.

6 Storage Management

6.1 OAM

OAM is a hierarchical data management system used for archive storage. CMOD can retrieve the stored object directly from disk or archive media such as a virtual tape system. CMOD uses the OSREQ macro interface. OAM uses DB2 both for its internal (indexing) tables and for online storage objects.

ARS_NUM_OAMSRVR - This is the maximum number of concurrently attached threads between the CMOD object server and OAM. Typically this is set to a number between 4 and 30, depending on client access patterns and object storage locations (disk vs. tape). This parameter has a maximum value of 30. Any value larger than 30 will result in a U0039 abend.

ARS_NUM_OAMSRVR_SLOW_RETRIEVE - Determines the number of task control blocks (TCBs) that the CMOD server starts to handle connections to OAM for retrievals from objects with a slow retrieval time as defined by the ARS_OAM_SLOW_RETRIEVE_THRESHOLD parameter. The ARS_NUM_OAMSRVR_SLOW_RETRIEVE parameter applies to all object servers. If the value specified for this parameter is zero, no TCBs are dedicated for slow retrievals and all retrievals are processed by the TCBs that are associated with the ARS_NUM_OAMSRVR parameter. The default is zero. The ARS_NUM_OAMSRVR_SLOW_RETRIEVE TCBs are in addition to the ARS_NUM_OAMSRVR TCBs and use additional DB2 connections.

ARS_OAM_SLOW_RETRIEVE_THRESHOLD - Specifies the threshold at which OAM retrievals are processed by the TCBs that are associated with the ARS_NUM_OAMSRVR_SLOW_RETRIEVE parameter. If the estimated retrieval time for an object (as indicated by QELQERRT) is greater than or equal to the value of the ARS_OAM_SLOW_RETRIEVE_THRESHOLD parameter, the OSREQ RETRIEVE is processed by a ARS_NUM_OAMSRVR_SLOW_RETRIEVE TCB. The default value is 12000. Refer to the Object Access Method Application Programmer's Reference manual for other valid QELQERRT values. An ARS_OAM_SLOW_RETRIEVE_THRESHOLD value of zero along with a non-zero ARS_NUM_OAMSRVR_SLOW_RETRIEVE value causes all OAM retrieve requests to be processed by the ARS_NUM_OAMSRVR_SLOW_RETRIEVE TCBs, while the ARS_NUM_OAMSRVR TCBs will process store, query, and delete requests.

6.1.1 OAM components

- OSR - Object Storage and Retrieval
- OTIS - OAM Thread Isolation Support
- OSMC - OAM Storage Management Component
- LCS - Library Control System

For detailed information on the OAM components and the parameters described below refer to the Object Access Method Planning, Installation, and Storage Administration Guide for Object Support.

6.1.2 OAM recommendations

- Define a user catalog exclusively for collection names.
- Cache the user catalog in VLF.
- Migrate objects to optical or tape (OSMC) during non-peak hours.
- Spread OAM collections over multiple DB2/disk/channels.
- Spread out the application groups to different collection names:

  OAM collections ==> storage groups ==> DB2 database

- Group your applications based on retrieval expectations (collecting small high use applications together, isolate your important applications so that the other applications do not get in the way).
- DB2
  - Ensure that there are enough DB2 connections available to support the OAM requests.
  - Run REORG, RUNSTATS, and REBIND as appropriate.
  - Partition OAM tablespaces larger than 2 GB.
- Devices
  - Determine and set the Initial Access Response Seconds (IARS).
  - Assign objects to storage classes that have an adequate initial access response seconds defined and to management classes that do not cause a transition to a slower storage class until the frequency of retrieving the objects is reasonably low.
  - Determine whether to place the object on disk or removable (optical or tape) media.
  - If REMOVABLE media is opted for by the IARS, determine whether to place the object on optical or tape.
  - Verify that the required Sustained Data Rate is achieved based on the selected object placement.
- Tape
  - Modify (CBROAMxx) parameters **MAXTAPERETRIEVETASKS** and **MAXTAPESTORETASKS** (if using tape retrieves, because the default=1). Both these parameters are configured at the global level and at the storage group level.
  - The global level MAXTAPERETRIEVETASKS (tasks) is defined by SETOAM. This specifies the total number of concurrent tape retrievals possible at a time (i.e. controls the maximum number of tape drives that can be concurrently allocated to the OAM address space for reading object data from tape). This number must be less than or equal to the number of tape drives on the system. Do not specify a number greater than the number of tape drives available to OAM for the MAXTAPESTORETASKS or the MAXTAPERETRIEVTASKS subparameters. This can cause a system to go into allocation recovery and attempt to allocate tape drives after all tape drives are in use causing system problems.
  - The storage group level MAXTAPERETRIEVETASKS(tasks) specifies the maximum concurrent tape retrievals possible for a specific storage group. Unless this is set, the default for each storage group is 1. This value needs to be set for each storage group that requires a value larger than the default of 1. For a single storage group, you must set this parameter if you need to retrieve documents from two or more tapes concurrently.
- Set the OAM cataloged procedure parameter **MAXS** (the number of storage groups that the storage management cycle processes concurrently) to an appropriate value. As MAXS is increased above 10, the effectiveness of concurrency is diminished and can severely constrain processing in OAM or cause OAM processing to be unsuccessful. (**Note:** If concurrent processing includes OBJECT storage groups writing to tape volumes, the correct corresponding (global level) MAXTAPERETRIEVETASKS and MAXTAPESTORETASKS values on the SETOAM statement must be specified.
- If you are using optical platters or tapes, set the tape **DEMOUNTWAITTIME** time parameter appropriately. This parameter determines how long OAM will leave a tape volume mounted in anticipation of another retrieval request from that device.
- Maximum Object Size (MOS)
  - OAM now supports storing object sizes up to 256 MB. APAR OA03623 lists the PTF available for each release of z/OS. To enable support for object sizes up to 256 MB, you need to specify the maximum object size by adding MOS =256 to the OAM subsystem definition parameter INITPARM in the SYS1.PARMLIB(IEFSSNxx) member.
  - The MOS can be viewed using the command D SMS,OAM, which would result in:
    - OAM1 Parms: TIME=GMT MSG=EM UPD=N QB=Y
    - MOS= 256 OTIS=N LOB=N DP=N
  - If the MOS is too small, CMOD returns an error similar to "OAM Store Failed with an OSREQ RC=8 and Reason=24020202". You should increase the MOS size. For more information, review the following Flash on the web at http://www.ibm.com/support/docview.wss?uid=isg3S1002890

- Optical platters, check and adjust appropriately the values for the following parameters in SYS1.PARMLIB(CBROAMxx):

    - MOUNTWAITTIME - Specifies the amount of time (in minutes) that can pass while a volume is waiting to be mounted on an operator-accessible drive within an optical library. After this time has expired, message CBR4426D is issued to allow the operator to retry or to cancel the volume mount request. This value can be any numeric value from 1 to 9999. If the operator retries the mount request, the value specified in the MOUNTWAITTIME parameter is used for the retry. The default value of this parameter is five minutes.

    - OPTICALDISPATCHERDELAY - Specifies the number of seconds that the OAM optical dispatcher is to delay processing of certain requests in order to minimize flipping of optical disk cartridges in an automated optical storage library expecting that another read request for the currently mounted optical disk volume will arrive within this delay interval. The OAM optical dispatcher will delay processing of a unit of work for a specific period of time, when ALL of the following conditions are true:
        - A read request for an object on a currently mounted optical disk volume has just been completed.
        - There is no request for the currently mounted optical disk volume waiting to be processed on the OAM optical dispatcher queue.
        - The OAM optical dispatcher has found a read request for another optical disk volume (either the opposite side of the currently mounted volume or an unmounted optical disk volume) and is about to dispatch this unit of work.
        - A nonzero optical dispatcher delay value has been specified with the OPTICALDISPATCHERDELAY keyword on the SETOPT statement in the CBROAMxx PARMLIB member.

    If another read request for the currently mounted optical disk volume arrives within the delay interval, that unit of work will be dispatched immediately upon arrival. If no read request for the currently mounted optical disk volume arrives within the delay interval another request for a different optical disk volume (either the opposite side of the currently mounted optical disk volume or an unmounted optical disk volume) is dispatched.

    Valid values for seconds are decimal number between 1 and 60. If use of this parameter is necessary, use of a low value between 1 and 5 is suggested.

- OAMplex … If this is an OAMplex installation, check and adjust appropriately the values for the following parameters in SYS1.PARMLIB(CBROAMxx):

    - **XCFTIMEOUT(XCFOPTREADA(20)** - The number of seconds (1 to 999999) that this instance of OAM waits for a response that indicates the completion of a shipped transaction from another instance of OAM in an OAMplex.
    - **XCFOPTREADM(50)** - The number of seconds that an OAM originating an optical read request for a shelf-resident volume, which is shipped to another OAM within the OAMplex that owns the library where the object resides for processing, should wait for completion of the read request.
    - **XCFOPTWRITEA(150)** - The number of seconds that an OAM originating an optical write request targeted for an object storage group that contains real (automated) optical libraries, which is shipped to another OAM within the OAMplex that owns the optical library defined to the object storage group for processing, should wait for completion of the write request.
    - **XCFOPTWRITEM(150)** - The number of seconds that an OAM originating an optical write request targeted for an object storage group that contains pseudo libraries, which is shipped to another OAM within the OAMplex that owns the pseudo library defined to the object storage group for processing, should wait for completion of the write request.
    - **XCFTAPEREADA(40)** - The number of seconds that an OAM originating a tape read request targeted for an automated tape library dataserver, which is shipped to another OAM within the OAMplex that owns the library in which the object resides for processing, should wait for completion of the read request.

- **XCFTAPEREADM(50)** - The number of seconds that an OAM originating a tape read request targeted for a manual tape library dataserver, which is shipped to another OAM within the OAMplex that owns the library in which the object resides for processing, should wait for completion of the read request.

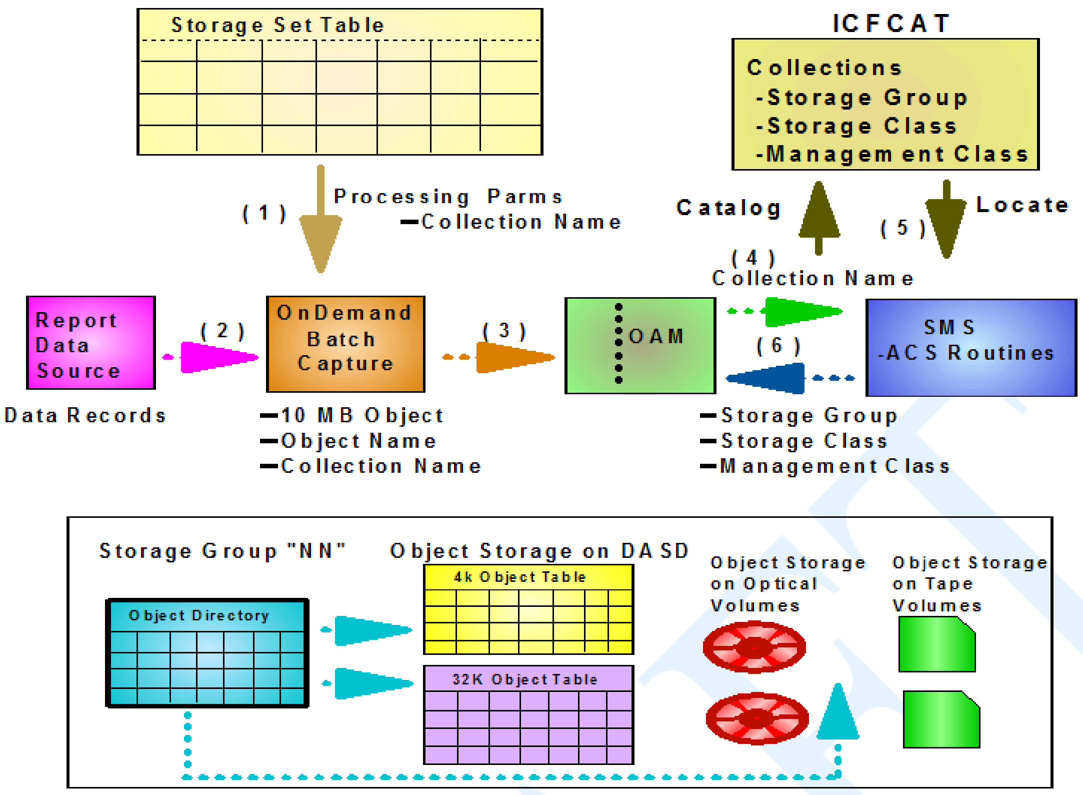


Figure 19. CMOD object storage in OAM

6.1.3 Sample OAM commands

The following table lists of some useful OAM commands. For more details about the listed commands see the z/OS: DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Object Support.

OSMC commands	
F OAM, START, OSMC	Storage management cycle for all
F OAM, START, STORGRP, SGROUP00	Storage management cycle for an individual storage group
F OAM, START, LIBMGT,OPTLIB01	Library management cycle
F OAM, START, DASDSM,SGROUP00	disk space management cycle
F OAM, START, RECOVERY,VOLSER	volume recovery
F OAM, START, OBJRECV,collection-name, obj-name	Single object recovery
F OAM, START, MOVEVOL,volser	Move volume recovery
Display commands	
DISPLAY SMS,OAM	Display the OAM status
DISPLAY SMS, LIBRARY (library_name) , DETAIL	Display the OAM library status
DISPLAY SMS,LIBRARY(library_name),STATUS	Display library connectivity
DISPLAY SMS, STORGRP (storgrp_name)	Display storage group status
DISPLAY SMS, VOLUME (volser)	Display tape volume status
LIBRARY DISPDRV, library_name	Display tape drive status
F OAM,QUERY,options	Querying Summary and Detail Information for Pending and Active (optical and Tape) Requests
F OAM,DUMP,(operands)	Capture OAM diagnostic data

6.1.4 CMOD OAM commands

CMOD z/OS V8.5 Display OAM threads

Command
The server has been enhanced to accept a console command of the form:

```
MODIFY ARSSOCKD,D,OAM
```

The MODIFY is actually a z/OS command that routes command strings to listening programs. The z/OS abbreviation for MODIFY is F, so an acceptable variation would be:

```
F ARSSOCKD,D,OAM
```

Response
In response, the server will display the status of the OAM threads to the console that issued the command in the form:

```
ARS0375I hh.mm.ss DISPLAY OAM
DB2      ssid     PLAN oamplan   GENERAL gcgcg  SLOW scscs
TCB@     STATUS COLLECTION/OBJECT
               IGST
                START=hh.mm.ss.ttt
                ISFE
                START=hh.mm.ss.ttt  STOP=hh.mm.ss.ttt
```

where

- hh.mm.ss is the time of the D OAM command
- ssid is the DB2 subsystem id for OAM
- oamplan is the OAM plan
- gcgcg is the count of OAM non-SLOW TCBs
- scscs is the count of SLOW TCBs
- xxxxxxxx is the TCB address
- STATUS is a four character status field:
  - B/I/* = Busy (B), Idle (I), or inconsistent (*)
  - G/S = General (G) or Slow (S)
  - ST = Store
  - FE = Fetch
  - QU= Query
  - DE= Delete
  - RL= Reload
- START=hh.mm.ss.ttt is the start time of the request in thousandths
- STOP=hh.mm.ss.ttt is the stop time of the most recent request in thousandths (for Idle TCBs)

Notes:
Threads are used to process each command. To prevent flooding of the server, an ars.cfg file setting is introduced:

```
ARS_NUM_CMD_THREADS=nnn
```

This parameter specifies the maximum number of threads to be used for command processing. The range is 1 to 255. The default value is 1. At startup of the ARSSOCKD server, that thread is created. As additional commands are entered, the threads will be created up to the limit specified in the parameter setting. If a command is entered and there is no thread available to process the command, the following message is issued:

```
ARS0376I Command rejected.  Server busy.
```

If the command verb was not D, the following message is issued:

```
ARS0377I Command rejected.  Verb xxx invalid.
```

If the operand following the D was not OAM, the following message is issued:

```
ARS0378I Command rejected.  Parameter xxx invalid.
```

Comments
Comments are allowed in the command and each comment must be contained between a slash-asterisk and asterisk-slash pair. Operands might be separated by commas or blanks. If a comment is started with a '/*' but no '*/' is found, the following message will be issued:

```
ARS0381I Command rejected.  Invalid comment.
```

If no verb was specified, the following message will be issued:

```
ARS0379I  MODIFY ignored, no verb specified.
```

Responses
Command responses, including error responses, will be directed to the console issuing the command, and will specify the CART (command and response token) associated with the command.

In the event that a task cannot be created to process a command, the following message will be issued:

```
ARS0380I Command failed creating task. RC = nnn
```

Number of parameters
If more than 10 parameters are specified, the following message will be issued:

```
ARS0383I Command rejected.  Excessive number of parameters
```

Security
For the D OAM command, a CMDAUTH READ check will be made using the CIB (Command Input Buffer) of the command issuer and the entity 'ARS.DISPLAY.OAM'. This will allow the installation to restrict who can issue the command, if desired. If permission is denied, the following message will be issued:

```
ARS0382I DISPLAY authority invalid, failed by security product.
```

There might also be messages issued by the security product. In the case of RACF, an ICH408I message is issued.

6.2 TSM

6.2.1 TSM overview

Tivoli Storage Manager (TSM) is implemented as a client/server system. The TSM client APIs are called by the CMOD server to perform store and/or retrieve requests from the TSM server program.

The TSM server program is typically installed on a separate system (or LPAR) so as to avoid competition for resources between it and the CMOD server program. You need to ensure that both the CMOD server and the TSM server have sufficient resources to perform in the manner that you require.

Connectivity between the TSM client and the TSM server is by means of TCP/IP. The allocated TCP/IP bandwidth could be a bottleneck. For optimal and consistent performance, IBM recommends that a dedicated network connection/path be established between the TSM client and the TSM server.

The TSM server implements a hierarchical storage mechanism in which data/objects are migrated from high speed devices to low speed devices. You should keep your data on the high speed devices for as long as it will be frequently accessed. In some implementations it is preferable to initially keep the data in cache storage in addition to TSM storage. Users then transparently access the cache storage for data retrieval. After some amount of time (determined by the administrator) the cache storage copy of the data is deleted and users now transparently retrieve the data from TSM.

Note: Tivoli Storage Manager with CMOD on z/OS and IBM i was deprecated in the announcement letters for CMOD V9.5 for z/OS and CMOD V7.2 for IBM i. On z/OS, this means that TSM will no longer be an option when defining a CMOD storage set. On IBM i, you will no longer be able to use TSM as a policy level in your CMOD migration policies. You will be able to continue to use TSM as a storage manager for both z/OS and IBM i if you configure a CMOD for Multiplatforms object server.

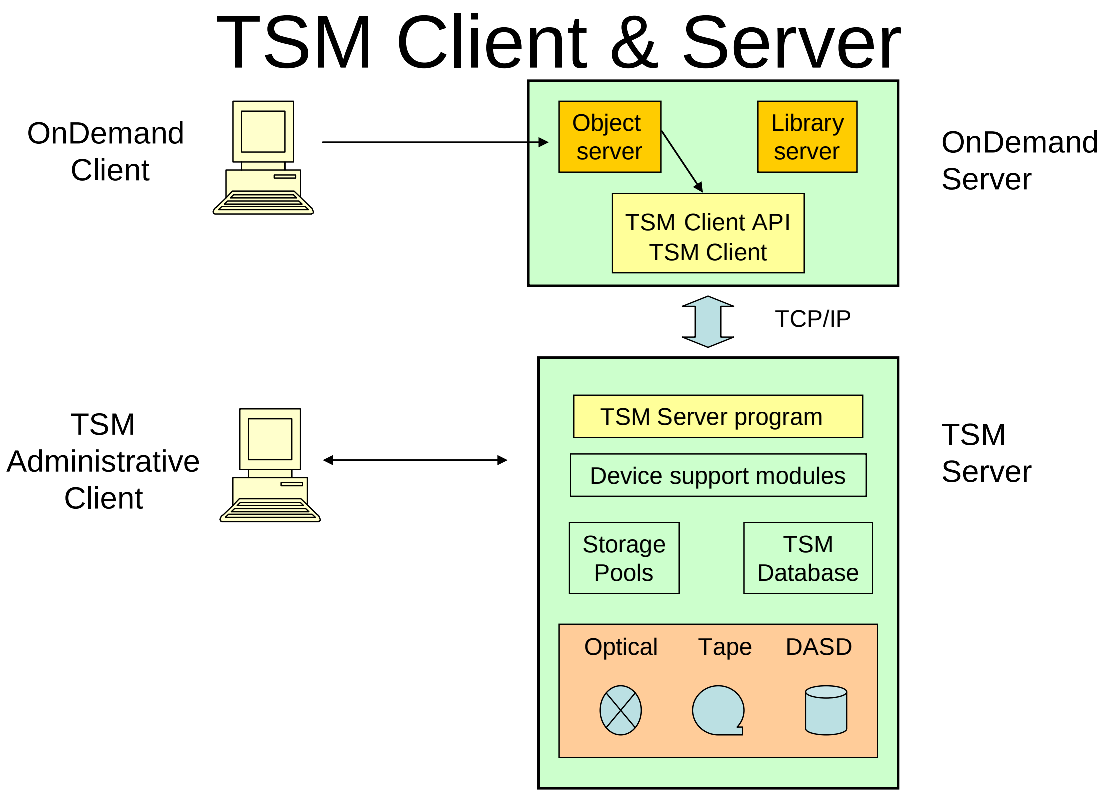


Figure 20. TSM client and server configuration

6.2.2 TSM for Windows

When implementing TSM on a Windows server:

Disk Space

- At least 3 GB of free disk storage (for a typical installation)
- 200 MB temporary directory space
- 2 GB partition size in the C:\ drive
- 300 MB in the instance directory

Significant additional disk space is required for database and log files. The size of the database depends on the number of CMOD objects to be stored and the method by which the server manages them. The default active log space is 16 GB, the minimum that is needed for most workloads and configurations. Allocate at least three times the active log space for the archive log (48 GB). Ensure that you have sufficient resources if you are using deduplication or expect a heavy client workload.

For optimal performance and to facilitate I/O, specify at least four separate directories or Logical Unit Numbers (LUNs) to be used by the database and logs. By making these specifications, I/O is balanced across multiple directories or mounts.

Memory

64-bit Windows systems:

- 12 GB
- 16 GB if you are using deduplication
- At least 32 GB for heavily used servers. Using 32 GB or more of memory enhances performance of the Tivoli Storage Manager server database inventory.
- If you plan to run multiple instances, each instance requires the memory listed for one server. Multiply the memory for one server by the number of instances planned for the system.

Configuring performance

The Performance Configuration helps you optimize the performance of the Tivoli Storage Manager server. Most CMOD customers should select the Mostly Large Files option and the 2 – 49 Client Nodes option.

6.2.3 TSM server console monitoring

Use the following to monitor activity on your TSM console:

```
dsmadmc -cons
```

Enter userid and password when prompted.

The information below is a sampling of the information that is displayed in the TSM server console:

```
ANR0469W Session 4 for node ODNODE (ADMIN) refused - client is not configured for
archive retention protection.
ANR0406I Session 5 started for node ODNODE (ADMIN) (Tcp/Ip WIN-8CLTEDBTH6F(50816)).
ANR0403I Session 5 ended for node ODNODE (ADMIN).
ANR0406I Session 6 started for node ODNODE (ADMIN) (Tcp/Ip WIN-8CLTEDBTH6F(50823)).
ANR0403I Session 6 ended for node ODNODE (ADMIN).
ANR0406I Session 7 started for node ODNODE (ADMIN) (Tcp/Ip WIN-8CLTEDBTH6F(50828)).
ANR0403I Session 7 ended for node ODNODE (
```

6.2.4 TSM server tuning

TSM performance can be influenced by many tuning parameters. Tuning these functions for optimal performance requires testing and expertise on the part of the implementer. Tuning TSM can be quite complex because of the many operating systems, network configurations, and storage devices that TSM supports.

TSM functions in the client/server world, supports many operating systems, works across networks, and accepts different communication protocols. Consequently, the use of TSM introduces more factors that affect performance. These factors can affect TSM performance significantly:

- Average transmitted file size
- CMOD Server configuration
    - Client hardware (CPUs, RAM, disk drives, network adapters)
    - Client operating system
    - Client activity (non-Tivoli Storage Manager workload)
- TSM Server
    - Server hardware (CPUs, RAM, disk drives, network adapters)
    - Server storage pool devices (disk, tape, optical)
    - Server operating system

    - Server activity (non-Tivoli Storage Manager workload)
    - Final output repository type (disk, tape, optical)
- Network
    - Network hardware and configuration
    - Network utilization
    - Network reliability
    - Communication protocol
    - Communication protocol tuning

6.2.5 TSM server tuning options

The following parameters can be tuned on most TSM servers. Not all options are supported on all server platforms. Reference the TSM Administrators Reference for applicability to a particular platform. These parameters can all be changed in server options file (dsmserv.opt). Some might also be changed with the server SETOPT command. Any changes made in the server options file will require the server to be halted and then restarted for the updates to take effect.

- **BUFPoolsize** - The database buffer pool provides cache storage, which allows database pages to remain in memory for longer periods of time. When database pages remain in cache, the server can make continuous updates to the pages without requiring I/O operations to external storage. While a larger database buffer pool can improve server performance, it will also require more memory.

  An optimal setting for the database buffer pool is one in which the cache hit percentage is greater than or equal to 99%. Typical implementation sizes are 1/8 to 1/2 of real memory.
- **EXPInterval** - Specifies the interval, in hours, between an automatic inventory expiration run by the TSM server. Inventory expiration removes client backup and archive file copies from the server. EXPInterval 0 (which means no expiration processing) and use an administrative schedule to execute expiration at an appropriate time each day.
- **LOGPoolsize** - Specifies the size of the recovery log buffer pool size in kilobytes. A large recovery log buffer pool might increase the rate by which recovery log transactions are committed to the database, but it also requires more memory. The recommended LOGPoolsize should be within the range of 2048 – 8192.
- **MAXSessions -** Specifies the maximum number of simultaneous client sessions that can connect with the TSM server. The default value is 25 client sessions. This is specified in the client NODE definition.
- **MAXNUMP** - The MAXNUMMP setting for the node, in the server, specifies the maximum number of mount points a node is allowed to use on the server and can be set to an integer from 0 – 999.
- **MOVEBatchsize / MOVESizethresh** - This option specifies the number of files that are to be moved and grouped together in a batch, within the same server transaction. The default value for MOVEBatchsize is 40 and the maximum value is 1000. The default value for MOVESizethresh is 500 and the maximum value is 2048. MOVEBatchsize and MOVESizethresh options help tune the performance of server processes that involve the movement of data between storage media.
- **RESTOREINTERVAL** - Restartable restores allow restores to continue after an interruption without starting at the beginning. This reduces duplicate effort or manual determination of where a restore process was terminated. RESTOREINTERVAL defines the amount of time an interrupted restore can remain in the restartable state.
- **SELFTUNEBUFPOOLsize** - Controls the automatic adjustment of the buffer pool size and is set to Yes or NO. The default is NO on the server. Specifying YES causes the data base cache hit ratio statistics to be reset prior to starting expiration processing and examined AFTER expiration processing completes. The buffer pool size is adjusted if the cache hit ratio is less than 98%. The percentage increase in buffer pool size is half the difference between the 98% target and the actual buffer pool cache hit ratio. This increase is not done if a platform specific check fails:

  UNIX - Buffer pool size cannot exceed 10% of physical memory.

  Windows - Same as UNIX with an additional check for memory load not exceeding 80%.

  MVS - Cannot exceed 50% of region size. The recommended setting for this parameter is Yes.
- **SELFTUNETXNsize** - Specifies whether TSM can automatically change the values of the TXNGROUPMAX, MOVEBATCHSIZE, and MOVESIZETHRESH server options. TSM sets the

TXNGROUPMAX option to optimize client-server throughput and sets the MOVEBATCHSIZE and MOVESIZETHRESH options to their maximum to optimize server throughput. The default is NO. The recommended setting for this parameter is Yes.

- **TAPEIOBUFS** - To use the BSAM overlap I/O buffering methods, a new server option is available: **TAPEIOBUFS** number of buffers
- **TCPNodelay** - The number of buffers specifies a number from 1 to 9. If 1 is specified, then Tivoli Storage Manager does not use the overlapping I/O for 3590 tape. The recommended setting for this parameter is Yes. This parameter applies to z/OS only.
- **TCPWindowsize** - The TCPWindowsize option specifies the amount of receive data in kilobytes that can be in-transit at one time on a TCP/IP connection.
- **TXNGroupmax** - Specifies the number of files transferred as a group between commit points. This parameter is used in conjunction with the TXNBytelimit client option. The default is 256.
- **USELARGEbuffer** - The USELARGEbuffer server option is now the default setting. This option increases communication and device I/O buffers. Both the client/server communication buffer and disk device I/O buffers are increased from 32 KB to 256 KB when this option is set to Yes.

6.2.6 TSM disk buffer size

For IBM i, IBM recommends that you increase the DISKBuffsize option in the dsm.sys.

The DISKBuffsize option specifies the maximum disk I/O buffer size (in kilobytes) that the client might use when reading files. The range of values is 16 through 1023; the default is 32. Optimal backup, archive, or HSM migration client performance might be achieved if the value for this option is equal to or smaller than the amount of file read ahead provided by the client file system. A larger buffer will require more memory and might not improve performance.

```
SErvername                                TSM1
   COMMMethod                             TCPip
   TCPPort                                1500
   DISKBuffsize                           1023
   TCPServeraddress                       tsm1.company.com
   COMPRESSION                            OFF
   ENABLEARCHIVERETENTIONPROTECTION       YES
```

Running a TSM trace with the flag 'verbdetail' will verify that you are using a larger buffer size to the TSM server. For more information regarding these server tuning parameters and the recommended settings, see the IBM Tivoli Storage Manager Performance Tuning Guide on the web at http://publibfp.dhe.ibm.com/epubs/pdf/c3291013.pdf

6.2.7 TSM client tuning options

The following parameters are tunable on all TSM clients for all available releases:

- `COMPRESSION`
- `COMPRESSALWAYS`
- `COMMRESTARTDURATION / COMMRESTARTINTERVAL`
- `QUIET`
- `DISKBUFFSIZE`
- `LARGECOMmbuffers`
- `MEMORYEFFICIENTBACKUP (previously SLOWINCREMENTAL)`
- `Multi-Session Restore`
- `RESOURCEUTILIZATION`
- `TAPEPrompt`
- `TCPBuffsize`
- `TCPNodelay`

- `TCPWindowsize`
- `TXNBytelimit`

Note: On UNIX clients, the TSM client options are located in either the dsm.sys file or the dsm.opt file. For more information regarding these client tuning parameters and the recommended settings, see the IBM Tivoli Storage Manager Performance Tuning Guide on the web at http://publibfp.dhe.ibm.com/epubs/pdf/c3291013.pdf

6.2.8 TSM connection pooling

Starting with CMOD 8.5.0.6, in order to enable TSM to work in a CMOD highly scalable environment, a connection pooling design was implemented. Connection pooling allows the CMOD server to choose a connection from among a pool of available connections rather than create a new connection to the TSM server for each TSM activity that CMOD invokes. Connection pooling allows for the creation of new connections to TSM only if necessary, and also allows pooled connections to close when activity levels drop.

CMOD allows up to 20 pooled connections for each CMOD storage node. Each pooled connection has an inactivity timeout of 15 seconds so under light use there will be no discernible difference in TSM performance with CMOD between versions 8.5.0.5 and 8.5.0.6. However, when under heavy load, the pooled connections will not be idle and so will not time out and will thus appear as permanent connections to TSM until the load on the system drops.

Configuration changes will need to be made to the TSM server and storage pools to allow a minimum of 100 simultaneous connections to the TSM server and to allow at least 20 mount points when possible for each device class utilized.

For physical media like tapes, the number of mount points possible for an actual volume may only be one. Each session to TSM ties up a piece of physical media as long as the session is active. This is the reason for the 15 second idle timeout for the connection pool sessions. This allows for the termination of an idle session in a reasonable amount of time and thus free a locked volume in TSM for another session to access when necessary.

6.3 ASM

The Archive Storage Manager (ASM) is the IBM i interface to archive media. ASM manages the long-term copy of documents. Before loading documents, you must define storage management objects such as migration policies, disk pools, and tape volumes as needed. Supported media includes:

- Disk pools (System ASP, User ASPs, Independent ASPs, or Network File System)
- Tivoli Storage Manager (**Note:** Tivoli Storage Manager with CMOD on IBM i was deprecated in the CMOD 7.2 IBM i announcement letter. You will no longer be able to use TSM as a policy level in your CMOD migration policies used by ASM. Data already in a TSM policy level should be migrated to another media type. You will be able to continue to use TSM as a storage manager if you configure a CMOD for Multiplatforms object server.)
- Optical (physical or virtual)
- Tape (physical or virtual)

6.3.1 Media Migration Facility (MMF)

The CMOD Spool File Archive Media Migration Facility (MMF) on IBM i provides a tool to move Spool File Archive data from one media type to another in an easy, recoverable way that can be stopped and restarted as needed. The source media for MMF can be an optical volume, a tape volume, or an individual report name. The target media can be disk for all types of source media, optical if the source media is optical, and ASM (Archive Storage Manager) if the source media is disk and contains reports that have been migrated from the Spool File Archive environment to the Common Server environment.

It might be possible to improve the performance of MMF by creating the following index using SQL:

```
CREATE INDEX QUSRRDARS/QARLRSRTX ON QUSRRDARS/QARLRSRT (CDTYPE, ONAME, RESTIND)
```

To determine if additional indexes might benefit your specific environment, you can run MMF in debug mode for some period of time to allow it to process several documents before ending MMF in a controlled manner.

If you plan to run MMF in an interactive environment, first issue the Start Debug command: STRDBG UPDPROD(*YES), to start debug. Once you start debug, start MMF and let it process long enough to complete the move of several documents. Then end MMF in a controlled manner, giving it at least 300 seconds to end. Issue the End Debug Mode (ENDDBG) command to end the debug environment. Review the resulting job log for message CPI432F which indicates that a new index is suggested. Use the CREATE INDEX command in SQL to create the additional index. You can repeat this process until there are no longer any CPI432F messages issued in the job log.

If you plan to submit MMF to run in batch, start the MMF job and then hold it. Issue the Start Service Job (STRSRVJOB) command for the MMF batch job, using the MMF batch job's qualified job name. Issue STRDBG UPDPROD(*YES) to start debug mode and then release the MMF job, allowing it to process several documents before ending it in a controlled manner as described previously.

If you need assistance with this process, contact IBM Support.

6.4 Cache Storage

Cache storage is a CMOD file structure on disk storage that allows for faster access to archived data in some CMOD configurations.

6.4.1 Disk

For Multiplatforms and z/OS, the cache file system can consist of one or more caches. Increasing the number of cache file systems allows CMOD to distribute data storage among them.

- Each cache file system should be a separate mount point.
- Cache performance can be further increased by placing each cache file system on a separate volume on a separate channel/disk I/O adaptor.
- For optimum performance, you should plan for multiples cache file systems on multiple mount points on multiple volumes, so that the data is divided between them and there is always space available on all of the file systems.
- Placing a cache file system offline prevents CMOD from accessing the documents stored in that file system.
- The first cache file system is the primary one. It contains CMOD control data and should never be placed offline. The control data serves as indexes to the objects stored in cache (resources and stored documents).
- If the first cache file is filled up, you will have to stop CMOD and add more disk space to that cache file before restarting CMOD.
- The cache file system should be cleaned up periodically with ARSMAINT. You should not plan to fill a cache file system then add another one. You should have multiple cache file systems throughout which the document data is distributed.
- For optimum performance, you should ensure that you do not fill any of your cache file systems beyond 80%.
- Cache file systems must be owned by the CMOD instance owner and the system group. Make sure that only the user file permissions are set, not the group or other file permissions.

z/OS only: In z/OS high availability environments, if the CMOD instance is configured such that there are redundant CMOD servers, then each of the caches must be a shared file system (for example, zFS or HFS) where each cache has its own mount point.

6.4.2 NFS

Network File System (NFS) is a distributed file system protocol originally developed by Sun Microsystems. NFS allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This allows for a great deal of flexibility in placing and assigning storage but also incurs a certain amount of overhead in terms of performance. You should weigh the benefits and costs of implementing NFS.

IBM has updated the recommendations for the use of Network File System (NFS) mounts with CMOD. The use of synchronous writes is now recommended. The use of asynchronous writes can result in data loss in the event of a system failure on either the CMOD server or the NFS server.

If you are exporting a Network File System on IBM i, you should edit the EXPORTS file and remove the parameter NoWaitForWrites. The EXPORTS file is located in the /etc directory. More information on NFS can be found in the IBM Knowledge Center on the web at www.ibm.com/support/knowledgecenter/ssw_ibm_i/welcome. More information on NFS support in CMOD for i can be found in the Knowledge Center or in the Content Manager OnDemand for i: Common Server Administration Guide for version 7.1 or higher, or on the web at http://www.ibm.com using ‘7010455’ for your search criteria for CMOD versions prior to 7.1.

6.4.3 NAS, SAN or any network-attached disk

If you are using any type of network-attached disk, then the network throughput will have an effect on the data retrieval rates. You need to test the network under heavy retrieval loads to verify the throughput. You might also need to place the device on a dedicated network so that other workloads do not affect the network throughput.

7 Choosing an indexer

Indexing is the process by which CMOD extracts report/document indexes from the input data and stores the indexes in the CMOD application group data tables.

CMOD supports multiple indexers. You should choose the indexer that you want to use based on

- The data type of the input data (APF, Line, PDF, SCS….)
- The data type that you want to store in the CMOD archive. The input data type can be transformed to another data type during the load/index process.
- The data type that you will be transmitting to your end users. The stored data can be transformed at retrieve time before it is sent to the end user.

Resource consumption throughout the system will be affected by the indexer that you choose and if, when, and where you decide to transform the data.

The figure below provides an overview of some of the various options available to you.

7.1 Indexers

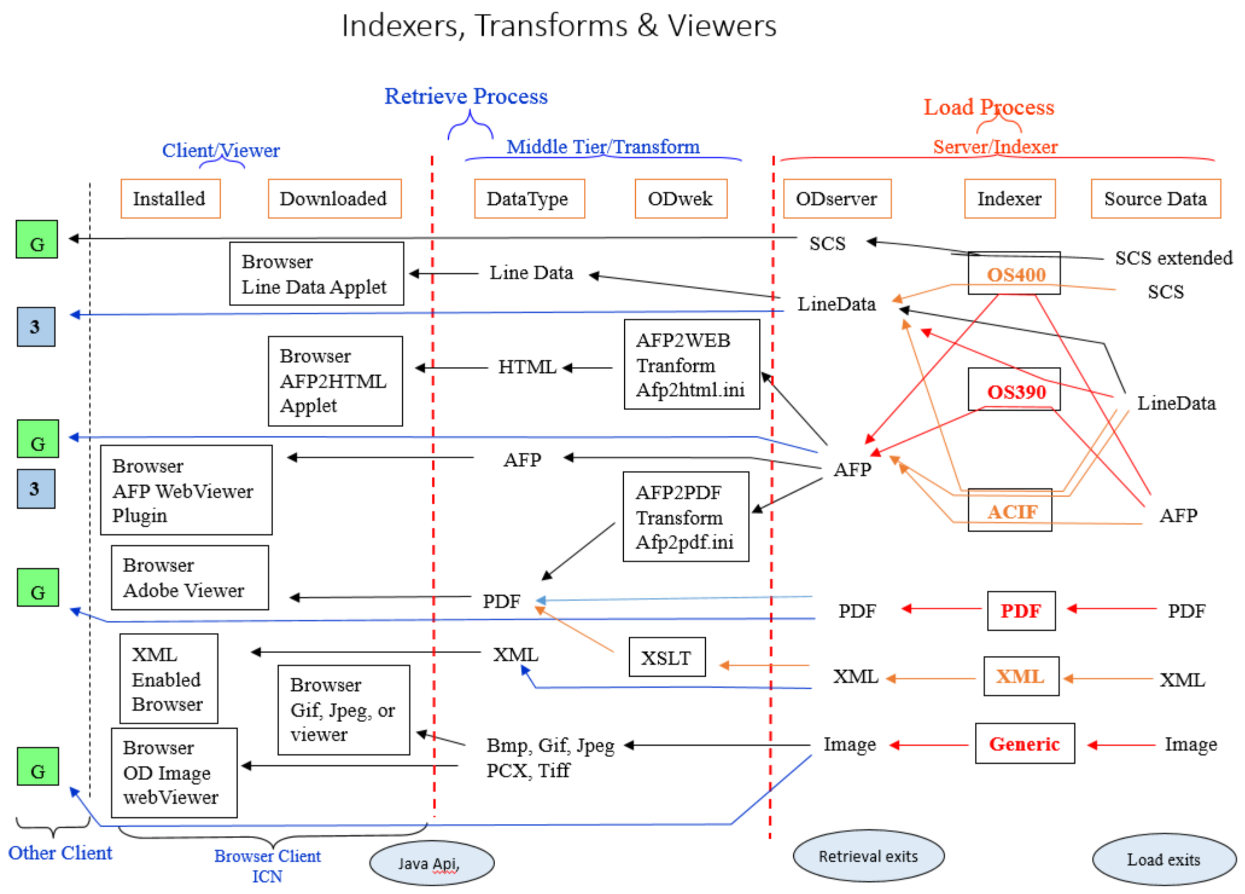


Figure 21. Indexers, transforms, and viewers

The diagram helps you determine the answers to questions such as:

1. When do you transform the data? At load time or at retrieval time?
2. In what format should you store the data?
3. What is the impact of the transforms on the retrieval processing requirements?

In looking at the right side of the diagram, the Load Process details:

- Source Data – the input report data type
- Indexer – the indexer that is capable of processing the input source data
- ODServer – the data type that is stored in the CMOD Server

The left side of the diagram shows the Retrieve Process details:

- Middle tier/Transform: If the data is retrieved by the middle tier it can optionally be transformed
  - ODWEK - shows the transforms that are available on the middle tier to transform the data retrieved from the server
  - DataType - shows the resulting data types
- Client Viewer: Shows some of the available client options:
  - Downloaded: shows components that are dynamically downloaded to the browser
  - Installed: shows components that must be installed on the client
  - Client: OD = OnDemand Windows client, 3 = 3270 CICS client

Note: Java APIs and the SAPI APIs can retrieve all data types. Displaying the data is determined by the customer application.

7.2 Input file size

The maximum input file size limit is 64GB. PDF is the exception, where the input file size limit is 4 GB. IBM recommends that the size of a PDF input file not exceed 500 MB.

The input file is logically divided into multiple indexed documents. The documents are compressed in a storage object (default size = 10 MB) and then stored in the archive.

In an OAM archive, a storage object (including large objects) cannot be larger than 256 MB. This is true for all data types.

As a best practice, IBM recommends that a storage object should not be larger than 50 MB. During the data loading process, the complete storage object will be in memory so most customers choose to implement the default object storage size of 10 MB.

7.3 ACIF Indexer

The ACIF (AFP Conversion and Indexing Facility) utility consists of three separate but related functions. ACIF can do the following:

- convert Line Data to AFP
- index data
- collect resources

ACIF accepts either Line Data or AFP as input, and can produce three files as output:

- the output file, called the .out file, which is either Line Data or AFP
- the index file, called the .ind file, which is an AFP file
- the resource file, called the .res file, which is an AFP file

Tools for working on ACIF issues

- a hex editor, available at http://www.hexedit.com/

A hex editor allows for the display of the raw contents of the file. No encoding or translation into text – just the raw input data. Note that line numbers instead of being based on "lines" are an offset address from the beginning of the file.

This will assist you in debugging your code.

- ARSAFPD utility, available with CMOD V9.0, will dump AFP into a readable text file

7.4 Generic Indexer

CMOD provides the Generic indexer to allow you to specify indexing information for input data that you cannot or do not want to index with ACIF, the OS/390 indexer, the OS/400 indexer, or the PDF indexer.

The input data is indexed using CMOD generic index data. If the **Data Type** on the View Information page of the CMOD application definition is **BMP**, **GIF**, **JFIF**, **PCX**, **PNG**, **TIFF**, or **User Defined**, you must use generic index data. You must specify the generic index data in an index file outside of the OnDemand Administrator client. Refer to the Content Manager OnDemand Indexing Reference for detailed information about the generic index data.

When you use the Generic indexer, you must specify all of the index data for each input file or document that you want to store in and retrieve from the CMOD system. This index data is placed in a parameter file. The parameter file contains the index fields, index values, and information about the input files or documents that you want to process. The Generic indexer retrieves the index data from the parameter file and generates the index information that is loaded into the CMOD database.

CMOD creates one index record for each input file (or document) that you specify in the parameter file. The index record contains the index values that uniquely identify a file or document in CMOD.

The generic indexer supports group-level indexes. Group indexes are stored in the database and used to search for documents. You must specify one set of group indexes for each file or document that you want to process with the Generic indexer.

The following shows an example of file names in daemon processing mode:

```
MVS.JOBNAME.DATASET.FORM.YYYYDDD.HHMMSST.ARD
MVS.JOBNAME.DATASET.FORM.YYYYDDD.HHMMSST.ARD.IND
MVS.JOBNAME.DATASET.FORM.YYYYDDD.HHMMSST.ARD.OUT
```

The .ARD file is the dummy file that triggers a load process in daemon mode. The .ARD.IND file is the Generic indexer parameter file, and the .ARD.OUT file is the input file to process. Note that on IBM I, the Start Monitor (STRMONOND) command with the *DIR option is triggered by the .IND file. The STRMONOND command with the *DIR2 option is triggered by the .ARD file.

After successfully loading the data, the system deletes all three files.

7.5 OS/390 Indexer

The OS/390 indexer provides higher throughput by extracting indexes and storing documents during a single pass of the input data. The OS/390 indexer indexes reports based on the organization of the data in the report.

The OS/390 indexer supports indexing of a wide variety of input:

- **AFP reports -** For AFP reports, the index values are already specified within the AFP data stream.
- **Document organization -** For reports made up of logical items, such as statements, policies, and invoices, the OS/390 indexer can generate index data for each logical item in the report.
- **Report organization -** For reports that contain Line Data with sorted values on each page, such as a transaction log or general ledger, the OS/390 indexer can divide the report into groups of pages and generate index data for each group of pages.

The OS/390 indexer also provides additional optional functions to assist in indexing data with unique requirements:

- **Anystore exit -** This exit can determine the content and index values of each document.

- **Large Object support -** Large object support is designed to provide enhanced usability and better retrieval performance for reports that contain very large documents by segmenting the documents into groups of pages and downloading only the page groups that the users request to view.

You run the OS/390 indexer as part of the CMOD load process with the ARSLOAD program. The CMOD application retrieves the indexer parameters from the CMOD database and uses the parameters to process the input data. The OS/390 indexer can logically divide reports into individual items, such as statements, policies, and bills. You can define up to 128 index fields for each item in a report.

The OS/390 indexer has been enhanced to allow for the storage of documents (or large object segments) that exceed 2 GB. A report might contain multiple documents (or large object segments) each of which exceeds 2 GB in size. This enhancement does not affect the limitations imposed by other indexers.

The limitations on the document size are based on the available hardware and any other limitations placed on the operating environment:

- If the document (or large object segment) size exceeds 20 MB, then the document data is temporarily stored in the CMOD temporary HFS directory (described below). For example, if the largest document is 6 GB, then the temporary HFS directory must have at least 6 GB of available space. If the available HFS disk space is not sufficient to store the largest document in the report, the load fails.

  The temporary HFS directory is identified by the first defined of the following options:
  - The -c option in the ARSLOAD parameters. If this is not specified, then
  - The environment variable named ARS_TMP. If this is not specified, then
  - The environment variable named TEMP. If this is not specified, then
  - The current working directory
- In the final load stage, the complete document (or large object segment) needs to be loaded into memory. Therefore, if the document (or large object segment) is 6 GB in size, then the load program needs to be able to acquire 6 GB of memory to load the data. If the available memory is not sufficient to store the largest document in the report, the load fails.

Any data type can be captured using the OS/390 indexer. Native support exists for Line Data and AFP data. Other data types, such as PDF and TIFF images, can be captured by using the Anystore exit. This provides a method to capture documents of any type and size (including those greater than 2 GB) into CMOD.

OS/390 indexer support on AIX

Starting with version 9, the OS/390 indexer became available on AIX. This allows for report data to be indexed on AIX then stored to AIX or z/OS or any running CMOD server.

7.6 OS/400 Indexer (IBM i)

The OS/400 indexer is the primary indexer used when running CMOD on an IBM i system. You can use the OS/400 indexer to specify indexer parameters for SCS, SCS-Extended, Advanced Function Presentation (AFP), and Line spooled files that you want to store in CMOD.

7.6.1 Find Once indexer function

Years ago, when customers were migrating from CMOD Spool File Archive to Common Server, an issue arose regarding the difference between the method in which Spool File Archive captured the date for a particular report versus how Common Server captures the date for that same report. In Spool File Archive, the date had to be found on the first page of the report. Once the date was found, it was never looked for again within an input file, and that date was used for all segments (documents) of the report. In Common Server, the date is looked for in every document (page group). If the location of the date changes from one document to the next, or the date format changes between documents, or the date is not found in every document, the spooled file fails to load.

On IBM i, CMOD now has a function to capture the date from the first document and then propagate that date for the remaining documents in the spooled file so that the processing of the date behaves similarly to how the date capture was handled in Spool File Archive. To enable this function, a default value must be specified for the field in the CMOD indexer parameters. The default value is '_*FINDONCE*_'. For any field that contains this special default value, the value for the field is found once, in the first document of the input file, and then that value is propagated to the remaining documents in the file. The following are two examples of how the indexer parameter for the field would look using this special default value:

```
FIELD6=0,39,8,(TRIGGER=7,BASE=0,DEFAULT='_*FINDONCE*_')
```

or

```
FIELD3=0,78,12,(TRIGGER=2,BASE=0,DEFAULT=X'6D5CC6C9D5C4D6D5C3C55C6D')
     /* _*FINDONCE*_    */
```

Notes:

1. If you are using the OnDemand Administrator client at a version prior to version 9, this special default value must be entered manually into the indexer parameters for each field for which you wish to use this function using the Keyboard modify option of the Indexer Information tab of the application definition.
2. This special default value is only valid for TRIGGER-based fields.

7.6.2 Merge Spooled Files command

The Merge Spooled Files (MRGSPLFOND) command is designed to improve the archive performance of small SCS spooled files. Archiving many small spooled files takes longer and uses more system resources than archiving one large spooled file. The MRGSPLFOND command is included in CMOD for i version 7.1 and higher.

The MRGSPLFOND command is distributed at V6.1 in a PTF containing the compiled program, sample command source, and help text. The required PTF is SI31629 for 6.1 (or its superseding PTF). The PTF cover letter contains detailed installation instructions for creating the help text and the command. Some parameter keywords changed between the sample command distributed at 6.1 and the official command included in 7.1. If you have written CL programs using the MRGSPLFOND command at 6.1, you must change these programs at version 7.1.

Limitations

- Only SCS spooled files are supported. There is no support for AFPDS, Line, or SCS-Extended spooled files.
- If the first page of each spooled file is a banner or alignment page, MRGSPLFOND might not work correctly.
- The Host Servers feature (option 12) of the operating system (IBM i, i5/OS) must be installed to use the MRGSPLFOND command.
- The help text is available only in English at version 6.1.

The MRGSPLFOND command combines multiple SNA character stream (SCS) spooled files and writes the result to a single spooled file or database file member. Spooled files to be merged must be contained in a single output queue and must be in ready (RDY) status. Spooled files that are not in a ready (RDY) status and that are not SCS will be left in the source output queue.

The first spooled file that is processed is used to define the attributes of the merged spooled file. All subsequent spooled files to be merged must have identical attributes. After each spooled file is merged successfully, it can be deleted or moved to another output queue where it can be optionally placed on hold. The resulting merged spooled file or database file member can also be automatically archived into a specified application group and application within a CMOD instance, after merge processing has completed.

How are spooled files selected?

You specify how the spooled files are selected, first by specifying the source output queue, and then one or more of the following:

- Spooled file name
- Form type
- User data
- Job name
- User defined options
- User defined data

All other spooled file attributes must be identical, such as record length and other attributes retrieved by the spooled file APIs. See the online help text for the command for more information about different options available when running MRGSPLFOND.

Performance

Note that no guarantees as to the performance you can obtain in your environment are made or implied in presenting this material.

Performance data included here was measured on an iSeries Model 820 with 4 processors running version 5.4 with CMOD 7.1.2.8 server and no other workload. Relative performance was calculated by taking the STRMONOND time in seconds divided by MRGSPLFOND time in seconds.

Performance Scenario 1

240 spooled files of 130 pages each, 31200 pages total

MRGSPLFOND: One merge job, elapsed time from start to end was 4 minutes 49 seconds

STRMONOND: One monitor job, elapsed time from start to end was 15 minutes 20 seconds

Relative performance: 920 seconds / 289 seconds = 3.2

In this scenario, the Merge Spooled Files command was 3 times as fast as the CMOD output queue monitor.

Performance Scenario 2

10,000 spooled files totaling 20,687 pages. Selection criteria created 5 sets of spooled files for merging

MRGSPLFOND: Ran merge jobs sequentially, elapsed time from first job starting to last job ending was 10 minutes 1 seconds

STRMONOND: Ran 5 monitor jobs simultaneously, elapsed time from first job starting to last job ending was 5 hours 0 minutes 58 seconds

Relative performance: 18058 seconds / 601 seconds = 30

In this scenario the Merge Spooled Files command was 30 times as fast as the CMOD output queue monitor.

7.6.3 When to use the SCS-Extended data type

When creating a CMOD application definition, data type must be specified. Most CMOD applications should be created with data type **SCS** rather than **SCS-Extended**. SCS-Extended is a unique data type that is processed differently to accommodate extended print attributes that might be ignored in an SCS data stream, such as variable lines per inch and variable characters per inch.

Both SCS and SCS-Extended spooled files appear as Printer device type *SCS when you display the spooled file attributes. However, other less common attributes under the heading of Device requirements can help you determine if your data requires the SCS-Extended data type. For example, the spooled file attributes of the data you are defining might look like the attributes shown below. Note the Variable CPI setting, which indicates that this spooled file has extended print attributes and should be defined as SCS-Extended.

```
                    Work with Spooled File Attributes
Job  . . . . . . . :   QPRTJOB      File . . . . . . . . . :   TESTSCS
  User . . . . . . :     DBRYANT      Number . . . . . . . :     000050
  Number . . . . . :     006123     Creation date  . . . :   08/01/07
Job system name  . :     IBMI7      Creation time  . . . :   11:16:35
Device requirements:
  Final form text . . . . . . . . . . . : N
  Variable font . . . . . . . . . . . . : N
  Variable LPI  . . . . . . . . . . . . : N
  Variable drawer . . . . . . . . . . . : N
  Superscript/subscript . . . . . . . . : N
  Variable character ID . . . . . . . : N
  Highlight . . . . . . . . . . . . . : N
  Extended 3812 font  . . . . . . . . : N
  Graphics 522X . . . . . . . . . . . : N
  Graphics 4214 . . . . . . . . . . . : N
  Graphics 4234 . . . . . . . . . . . : N
  Graphics  . . . . . . . . . . . . . : N
  Bar codes . . . . . . . . . . . . . : N
  Rotation  . . . . . . . . . . . . . : N
  PC printer emulation  . . . . . . . : N
  Defined characters  . . . . . . . . : N
  Variable CPI  . . . . . . . . . . . : Y
  Transparency  . . . . . . . . . . . : N
  IPDS transparent data . . . . . . . : N
  Field outlining . . . . . . . . . . : N
```

Figure 22. SCS Extended spooled file attributes

Due to the difference in the data stream and how it is handled within CMOD, the following facts should be noted when considering whether to use SCS or SCS-Extended data type:

- SCS-Extended is displayed in the AFP Viewer, not in the Line Data viewer. Most viewer options supported for SCS are not supported for SCS-Extended. This includes overstrike mode and all options related to logical view fields and headers.
- The Start Indexing on Page (STARTINDEXINGONPAGE) indexer parameter is not supported for SCS-Extended.
- The Merge Spooled File (MRGSPLFOND) command does not support SCS-Extended.
- Orientation (rotation) and Paper Size are supported for SCS-Extended.

7.6.4 Using carriage control character X'F1' as trigger 1

You should use carriage control character X'F1' as trigger 1 only if all the documents in the report are one page in length or if a better choice is not available. The use of X'F1' forces the indexer to search every page for triggers and fields.

A better choice for trigger 1 might be to choose characters that appear only on the first page of each new document, such as 'Page 1'.

- If you print statements, invoices, or phone bills, for example, the first page of each new document will begin with Page 1, and there is no need to look for group triggers in subsequent pages of the document.
- If you want to archive an entire report as a single document (such a report might be a good candidate for large object support), it is more efficient just to find trigger 1 on page 1 of the report.

7.6.5 Using Group triggers or Float triggers

Use Group triggers when:

- The trigger is always found on the same page as trigger 1
- The trigger is always found in the same row or range of rows relative to trigger 1
- The trigger is found more than once, but you only want to use the first occurrence

Use Float triggers when:

- The trigger might be found on the same page or on a different page as trigger 1
- The trigger is found more than once and you want to use all occurrences
- The trigger and associated fields and indexes are not found in every document
- The index will be defined as multi-key

7.7 PDF Indexer

IBM recommends the following for best performance of the PDF indexer:

- Running the PDF indexer on Windows can reduce the workload on your CMOD server.
- If you are running the PDF indexer on z/OS, ensure that the PDF load file resides under UNIX System Services (USS).
- Create PDF documents using the base 14 (standard) fonts, as described in the Adobe PDF Reference.
- Consider setting **Collect PDF Resources** to **Yes** in Indexer Properties in the CMOD graphical indexer (or RESTYPE=ALL in indexer parameters) for your CMOD application definition to optimize the storage of PDF resources. There is a tradeoff between the CPU usage required to collect the resources versus the amount of space required to store them. You might want to determine which of these is more important. Note that setting **Collect PDF Resources** to **Yes** is the default in CMOD version 8.5 and higher.
- Avoid using Type 3 fonts (which are user-defined fonts). These fonts cannot be subset and are not scalable.
- Create PDF documents with no bookmarks.
- Change Acrobat Distiller to minimize the amount of font data embedded.
    - Click Settings > Edit Adobe PDF Settings > Fonts
    - Check **Subset embedded fonts when percent of characters used is less than** and specify **100** for the percentage

When using the PDF indexer, the maximum size of any individual document within a PDF input file is 2 GB.

Note: The PDF indexer is not available on z/OS beginning with version 9.5. To load PDF reports to a CMOD 9.5 z/OS instance, IBM recommends that you run the load/indexing job on a Windows system and allow the output indexes and data to be automatically stored to the z/OS server.

7.7.1 Viewing and printing

When viewing PDF documents from the OnDemand Windows client:

- Beginning in CMOD version 9.5, regardless of whether the CMOD data type is native PDF or user-defined PDF, the OnDemand Client will attempt to view it with Adobe Acrobat if it is installed.
- When printing CMOD data that is defined with a data type of user-defined PDF, CMOD will now check if Adobe Acrobat is installed and attempt to print locally.

7.8 XML Indexer

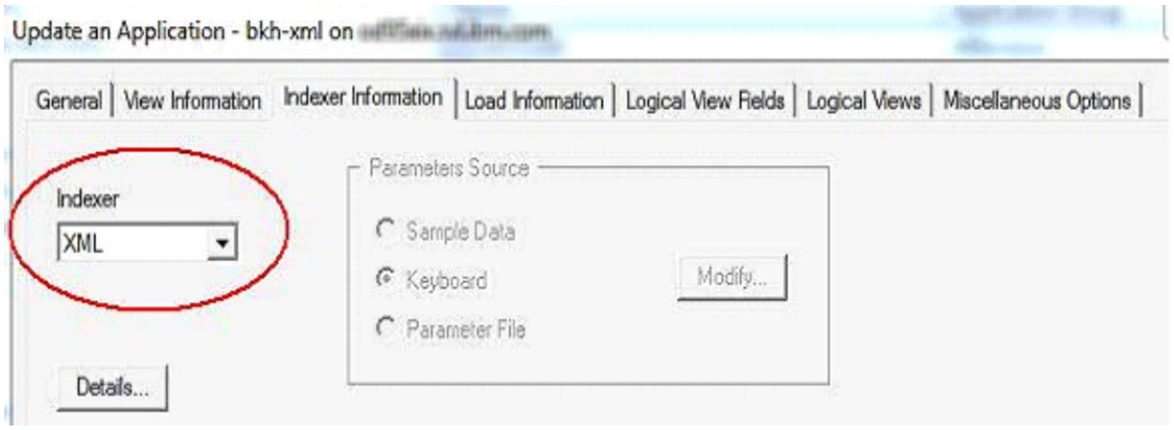
XML (Extensible Markup Language) indexing is new with CMOD version 9.5. With XML indexing, you can index and archive your XML documents into CMOD. You must indicate to the XML indexer which documents you wish to store, along with the index values for each of the documents. This is accomplished by transforming your XML input into an interim format.

7.8.1 Configure

When using the XML indexer for indexing your XML data files, you will need to configure your CMOD application with the following:

- Data Type = XML
- Indexer = XML

The Indexer field is shown here:


Update an Application - bkh-xml on [redacted]

Indexer: XML

Parameters Source: Keyboard

Figure 23. CMOD application configuration with XML

7.8.2 Transform

XML is a markup language that defines a set of rules for encoding documents. There are standards that exist to:

- Define structure
- Define node names
- Define attributes

Data must be normalized/transformed to enable CMOD to:

- Identify what is a "document" in your XML
- Identify what index values should be used for each document
- Identify global resources (stylesheets)

The XML data must be in a format that CMOD can ingest. To get the file in this format and ready for indexing, there are two tools to assist you:

XSLT – (Extensible Stylesheet Language Transformations)

- A language for transforming XML documents into other XML documents
- A new document is created based on the content from the existing one
- Defines what patterns to process and how to process

XQuery– (Extensible Stylesheet Language Transformations)

- A query and functional language
- Used to query and transform your XML data into a CMOD format for XML indexing

7.8.3 Index

The XML indexing process is a:

- Single pass with no intermediate files
- Integrated with ARSLOAD
  - `arsload –nfv –I <instance> -u user –p <stash> -g <ag_name> <input file>`
- All data between <odxmldata> tags in the input file becomes a CMOD document.

7.8.4 Resource

While XML files are readable, there are two technologies that make viewing of XML data easier:

- Cascading Style Sheets (CSS)

- Extensible Stylesheet Languages (XSL)

Multiple CSS or XSL files can be defined for each XML document. These files are considered resources within CMOD and can be archived as resources. These resources are specified in your XML data using the <?xml-stylesheet> processing instruction.

The XML resource process uses the following:

- Style sheets are specified with the <xsl-stylesheet> processing instruction
  <?xml-stylesheet type="text/css" href="file:cd_catalog.css"?>
  - href (refers to the XML declaration) attribute required
    - Must be an Internalized Resource Identifier (IRI)
    - Contains scheme; for example *http* or *file*
- Multiple style sheets can be specified
  - Display on multiple devices (mobile, tablet, desktop)
  - Larger font for visually impaired
- True separation of data and view
- All style sheet instructions are copied to each document
- Only local style sheet instructions are gathered and archived as a resource by ARSLOAD
  - Must be in directory or subdirectory of the execution directory
    - Refer to `arsload –c` option
  - Resource file is in a zip file format
- Embedded resources, for example corporate logo
  - Create your own resource (zip file)
  - Name it <`input file`>.res
  - ARSLOAD will use the .res
    - Note that this disables ARSLOAD collection of any other resources

7.8.5 View

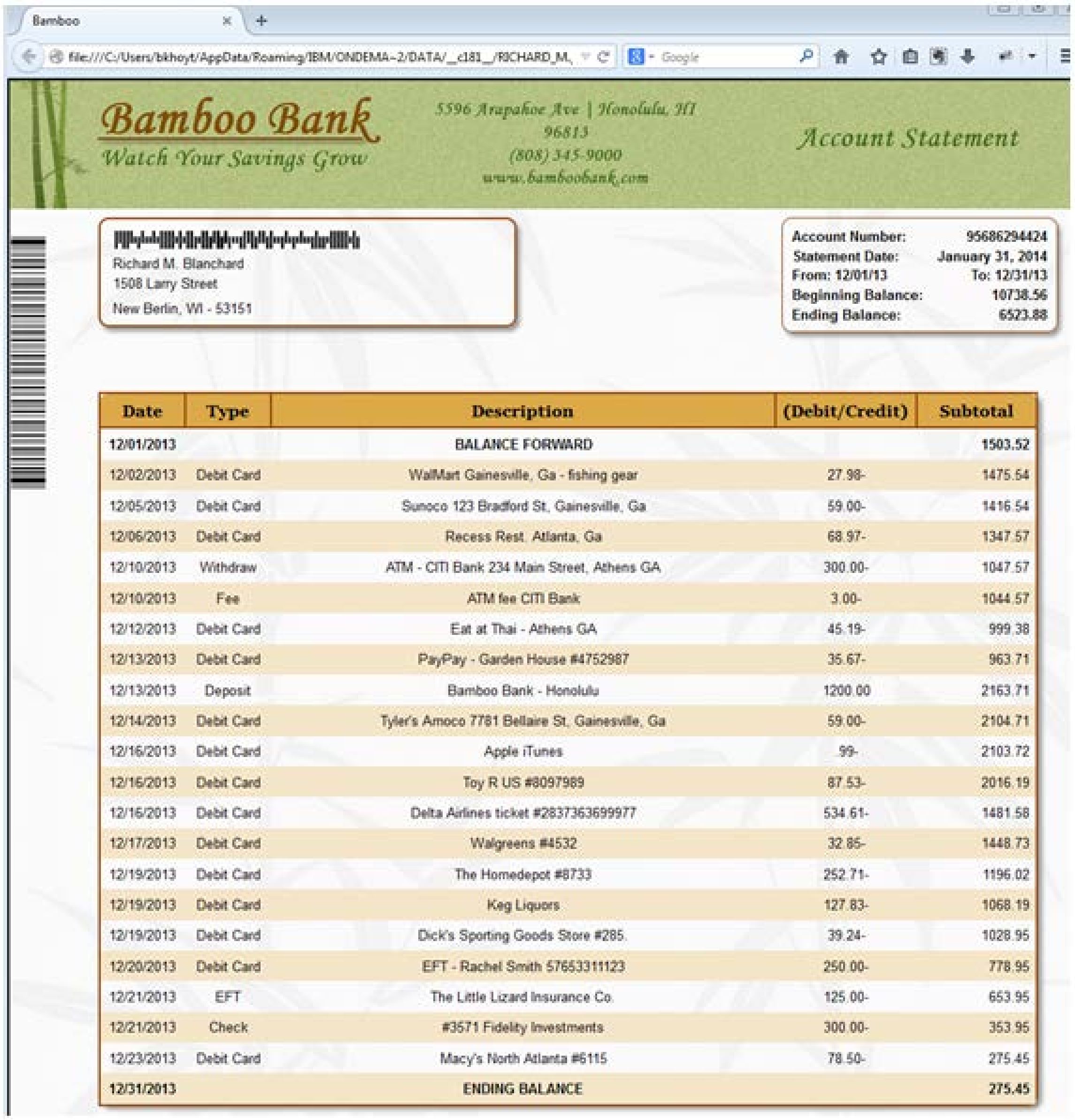
The viewers that are available for viewing the archived XML data:

- OnDemand Client
  - Resources unzipped
  - External viewer based on mime-type application/xml
- IBM Content Navigator (ICN)
  - Resources unzipped
  - View in browser window
- ODWEK
  - CGI/Servlet will return just the XML document (no resources)
  - APIs return XML document and resources
    - `ODHit.getDocument()`and `ODHit.getResources()`

7.8.6 Sample XML statement (Bamboo Bank)

A sample XML statement is illustrated below. There are two components that are shown in this example:

1. The bank statement with multiple resources
   a. Style sheets
   b. Images
2. The resources referenced by documents
   a. Bamboo.xsl
   b. Bamboo.css
   c. BBtemplate1600-3.png
   d. Psbarcode.png


Bamboo

file:///C:/Users/bkhoyt/AppData/Roaming/IBM/ONDEMA~2/DATA/__c181__/RICHARD_M,

Bamboo Bank
Watch Your Savings Grow

5596 Arapahoe Ave | Honolulu, HI
96813
(808) 345-9000
www.bamboobank.com

Account Statement

Richard M. Blanchard
1508 Larry Street
New Berlin, WI - 53151

Account Number:	95686294424
Statement Date:	January 31, 2014
From: 12/01/13	To: 12/31/13
Beginning Balance:	10738.56
Ending Balance:	6523.88

Date	Type	Description	(Debit/Credit)	Subtotal
12/01/2013		**BALANCE FORWARD**		**1503.52**
12/02/2013	Debit Card	WalMart Gainesville, Ga - fishing gear	27.98-	1475.54
12/05/2013	Debit Card	Sunoco 123 Bradford St, Gainesville, Ga	59.00-	1416.54
12/06/2013	Debit Card	Recess Rest, Atlanta, Ga	68.97-	1347.57
12/10/2013	Withdraw	ATM - CITI Bank 234 Main Street, Athens GA	300.00-	1047.57
12/10/2013	Fee	ATM fee CITI Bank	3.00-	1044.57
12/12/2013	Debit Card	Eat at Thai - Athens GA	45.19-	999.38
12/13/2013	Debit Card	PayPay - Garden House #4752987	35.67-	963.71
12/13/2013	Deposit	Bamboo Bank - Honolulu	1200.00	2163.71
12/14/2013	Debit Card	Tyler's Amoco 7781 Bellaire St, Gainesville, Ga	59.00-	2104.71
12/16/2013	Debit Card	Apple iTunes	.99-	2103.72
12/16/2013	Debit Card	Toy R US #8097989	87.53-	2016.19
12/16/2013	Debit Card	Delta Airlines ticket #2837363699977	534.61-	1481.58
12/17/2013	Debit Card	Walgreens #4532	32.85-	1448.73
12/19/2013	Debit Card	The Homedepot #8733	252.71-	1196.02
12/19/2013	Debit Card	Keg Liquors	127.83-	1068.19
12/19/2013	Debit Card	Dick's Sporting Goods Store #285.	39.24-	1028.95
12/20/2013	Debit Card	EFT - Rachel Smith 57653311123	250.00-	778.95
12/21/2013	EFT	The Little Lizard Insurance Co.	125.00-	653.95
12/21/2013	Check	#3571 Fidelity Investments	300.00-	353.95
12/23/2013	Debit Card	Macy's North Atlanta #6115	78.50-	275.45
12/31/2013		**ENDING BALANCE**		**275.45**

Figure 24. Sample XML statement – Bamboo Bank

7.9 Full Text Search

Full Text Search is new with CMOD version 9. With Full Text Search (FTS), you now have the ability to search for content based not only by metadata (extracted indexes) but also on the content itself (the document data). The FTS feature of CMOD is based on the Apache Lucene text search engine library. This is the same text search engine used today by IBM in DB2, Content Manager, and FileNet.

7.9.1 Architecture

The CMOD Full Text Search architecture is illustrated below. In addition to the CMOD server, there are two more components that need to be installed and configured. These are:

1. the FTS exporter
2. the FTS server

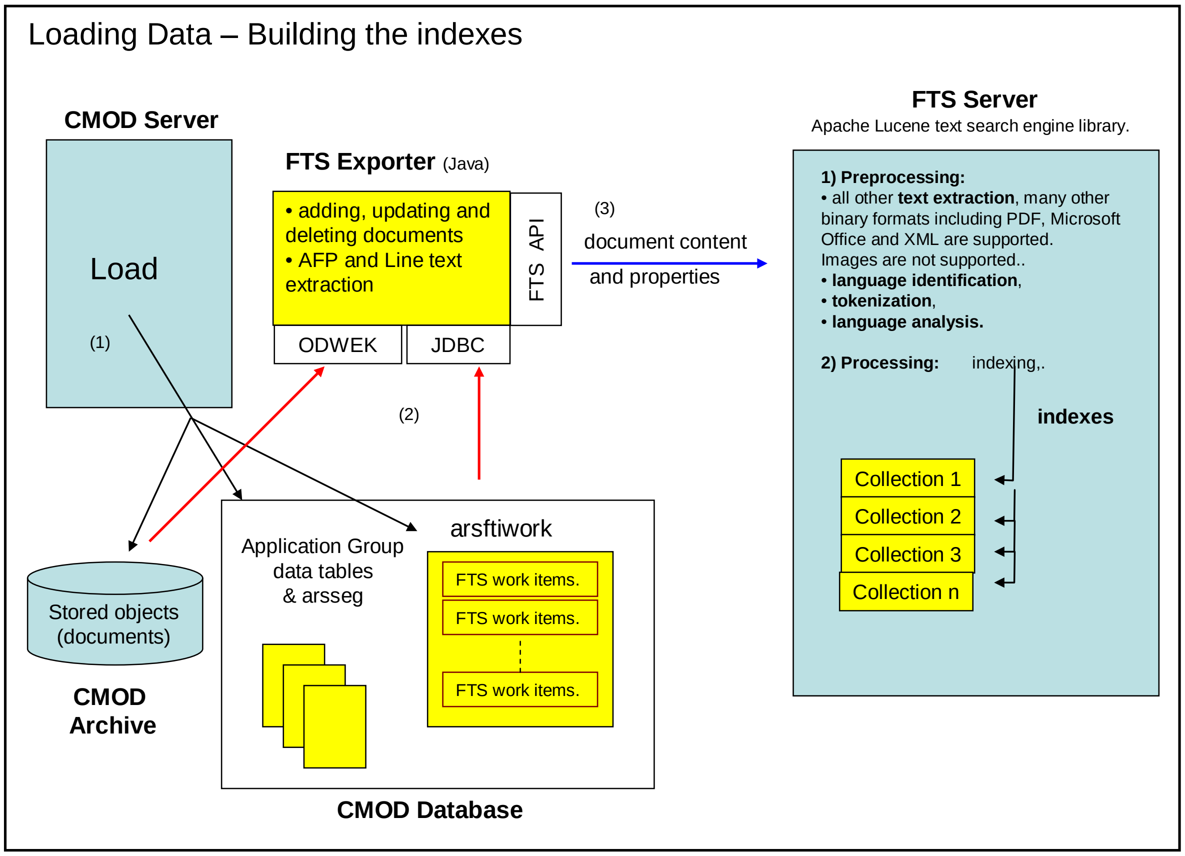

Loading Data – Building the indexes
CMOD Server
Load
(1)
FTS Exporter (Java)
• adding, updating and deleting documents
• AFP and Line text extraction
FTS API
ODWEK
JDBC
(2)
(3)
document content
and properties
FTS Server
Apache Lucene text search engine library.
1) Preprocessing:
• all other text extraction, many other binary formats including PDF, Microsoft Office and XML are supported. Images are not supported..
• language identification,
• tokenization,
• language analysis.
2) Processing: indexing,.
indexes
Collection 1
Collection 2
Collection 3
Collection n
Stored objects (documents)
CMOD Archive
Application Group data tables & arsseg
arsftiwork
FTS work items.
FTS work items.
FTS work items.
CMOD Database

Figure 25. Data loading and index building with FTS

7.9.2 Data loading and FTS index building

When data is loaded to CMOD and the application group is defined to support FTS, the following steps take place:

1. The document data is stored in the CMOD archive. The document indexes are stored in the application group data tables (same process as the usual CMOD load). In addition to this, if the application group is identified as being eligible for FTS, a work item (one row per report) is added to the ARSFTIWORK table.
2. Based on an Administrator-defined time interval, the FTS exporter asynchronously reads a row out of the ARSFTIWORK table and, for each document in the report, retrieves the document data from the CMOD archive. If the document data is AFP or Line Data, then the FTS exporter will perform the text extraction and send the result to the FTS server. Otherwise, the document will be sent directly to the FTS server.
3. The FTS server receives the document content and any associated properties. The FTS server then performs the document indexing and sends a completion code back to the FTS exporter.
4. The FTS exporter deletes the work item from the ARSFTIWORK table.

Documents can also be loaded and indexed through the ARSDOC command and/or the ODWEK Java APIs. In all cases, the interactions with the CMOD server, the FTS exporter, and the FTS server are similar.

7.9.3 FTS index storage

The CMOD system is highly scalable. One of the features that allow for this scalability is the way that the application group data tables are segmented within the CMOD database. The size and number of tables created can be optimized for a particular application group such that data loading, storage, and expiration are performed in the most efficient manner.

This concept has been carried through to the CMOD-FTS implementation. With FTS, the data for an application group segment table is stored in a single FTS server collection. Each time a new application group data table is created within CMOD, a new collection is created for the data's full text index. This means FTS collections maintain a one-to-one relationship with CMOD data tables. Collections are created with the following naming convention: InstanceName_TableName. An example of this relationship is shown below.

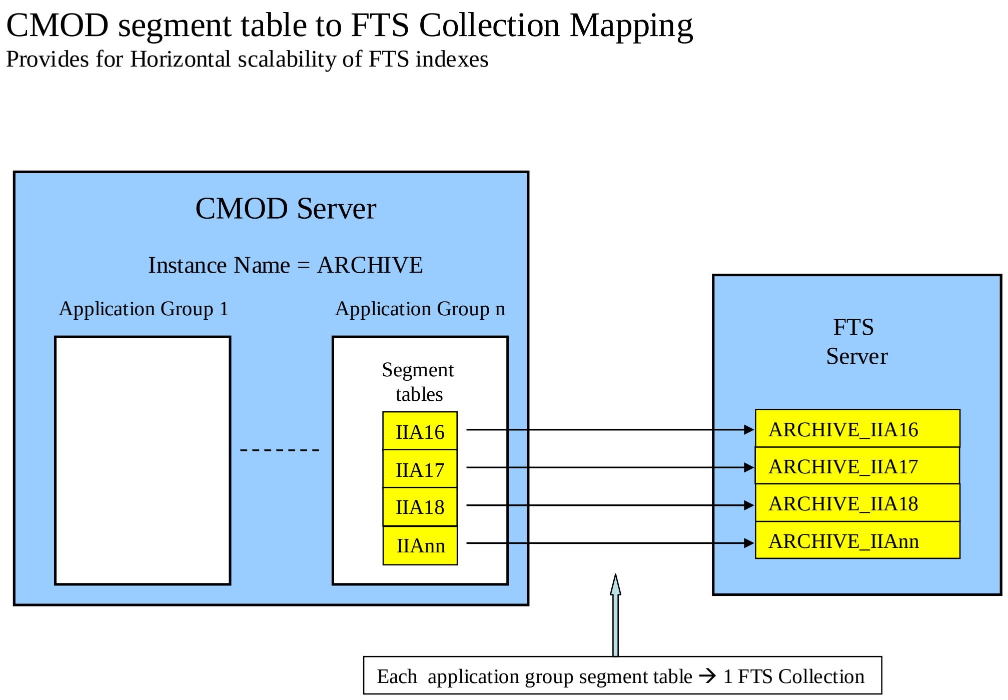


Figure 26. CMOD data table relationship with FTS collections

7.9.4 Querying and retrieving documents

The query and retrieval process is illustrated below.

1. The documents indexed using FTS can be searched from any of the CMOD clients.
2. During the definition of the folder(s) to be used to query the application group, up to four new folder field types are added to support FTS. The field types are **Score**, **Summary**, **Highlight**, and **Full Text Search**. At a minimum, the **Full Text Search** field type must be specified, since this field is used by the end user to specify a query string to the FTS server.
3. During the query process, the string is passed to the CMOD library server where it is used to issue the query against the FTS server.
4. The results from this query will then be filtered by the CMOD server so that CMOD permissions are honored. This means that the CMOD server will also issue a database query against the CMOD data table(s) and only those hits matching that query will be returned from the query.
5. If the CMOD administrator has created any of the other three FTS folder field types, this information will also be returned to the client.
6. The user then selects the document to be retrieved and the retrieval process within CMOD proceeds as normal. The document is retrieved from the CMOD archive and sent back to the client for display.

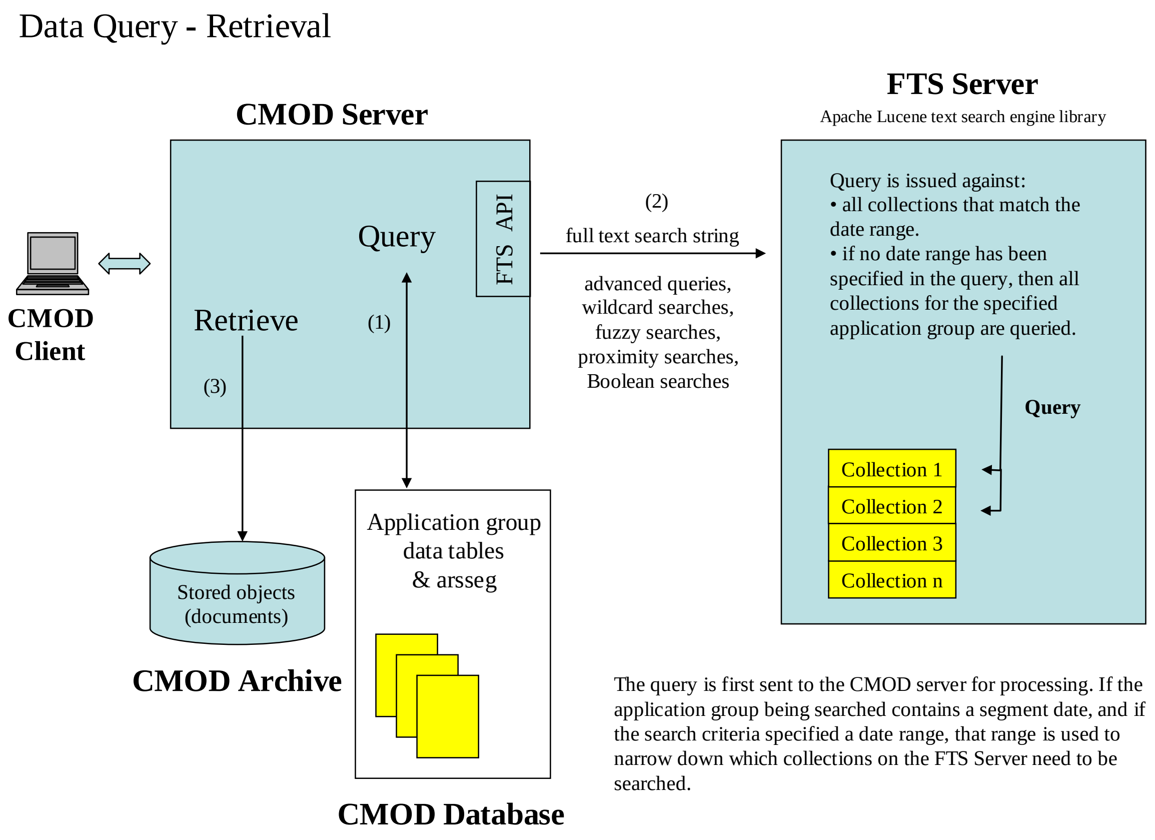

Data Query - Retrieval
CMOD Server
FTS Server
Apache Lucene text search engine library
CMOD Client
Query
Retrieve
FTS API
(1)
(2)
(3)
full text search string
advanced queries, wildcard searches, fuzzy searches, proximity searches, Boolean searches
Query is issued against:
• all collections that match the date range.
• if no date range has been specified in the query, then all collections for the specified application group are queried.
Query
Collection 1
Collection 2
Collection 3
Collection n
Stored objects (documents)
CMOD Archive
Application group data tables & arsseg
CMOD Database
The query is first sent to the CMOD server for processing. If the application group being searched contains a segment date, and if the search criteria specified a date range, that range is used to narrow down which collections on the FTS Server need to be searched.

Figure 27. Data query and retrieval

7.9.5 Deleting the indexes and documents

Deleting documents from FTS is performed by either:

1. the CMOD expiration process. **Note:** Application groups must specify **Expiration Type** of **Load**.
2. the CMOD unload process
3. the ARSDOC command
4. the ODWEK Java APIs

7.9.6 Best practice notes

The FTS Exporter handles all tasks related to adding, updating and deleting documents to and from the FTS index asynchronously. For example, when the ARSLOAD process completes, that does not mean that the report has been completely indexed on the FTS server. There will be a time delay that is proportional to the processing resources allocated to the FTS server versus the processing resources allocated to CMOD.

1. FTS consumes additional CPU and time beyond what is needed by CMOD to extract index data from reports and documents.
   - The time required to complete an FTS task is influenced by the number of documents to be indexed, the document size, and the index type.
   - Apart from merge operations, index updates are linear for average-sized indexes. For example, the time to index 40 GB is about a factor of 4 compared to 10 GB, 80 GB a factor of 8, and so on.
   - The processing time for each document is the sum of a fixed cost and a variable cost. The fixed cost is influenced by the type of document (plain text, XML, or binary). The variable cost is determined by the document size, linguistic processing variations, and the handling of different MIME types for binary indexes.

    - The number of documents that can be processed in a given timeframe increases for smaller sizes, but the total throughput (in GB/hour) is significantly less than for larger documents due to the fixed cost per document.
    - Optimal processing occurs when there are approximately 10-100 KB of text per document. Throughput degrades above 1 MB and below 1 KB of text.
2. The ARSFTIWORK table is a new table created in the CMOD database. Additional space will need to be allocated for that table. Typically the table is not very large (one row per report) and is never deleted (only individual rows are added and deleted).
3. The FTS exporter can potentially consume a large number of CPU cycles especially for AFP and Line Data documents. Sufficient CPU and I/O need to be allocated to the FTS exporter such that it can keep pace with the CMOD document loads.
4. The FTS server can potentially consume a large amount of CPU, memory, I/O and disk. The server hardware needs to be sized appropriately and should be a dedicated system.
5. All the FTS indexes and documents are stored on the FTS server. Procedures need to be put in place (to the extent of a dual system or hot standby, depending on customer needs) to minimize downtime of the FTS server.

7.9.7 Further reading

For more information, see the CMOD Full Text Search technical document available on the web at www.ibm.com using ‘1606305’ as your search criteria.

8 Data loading

8.1 Overview

The following figure illustrates the components of CMOD that are involved in loading data.

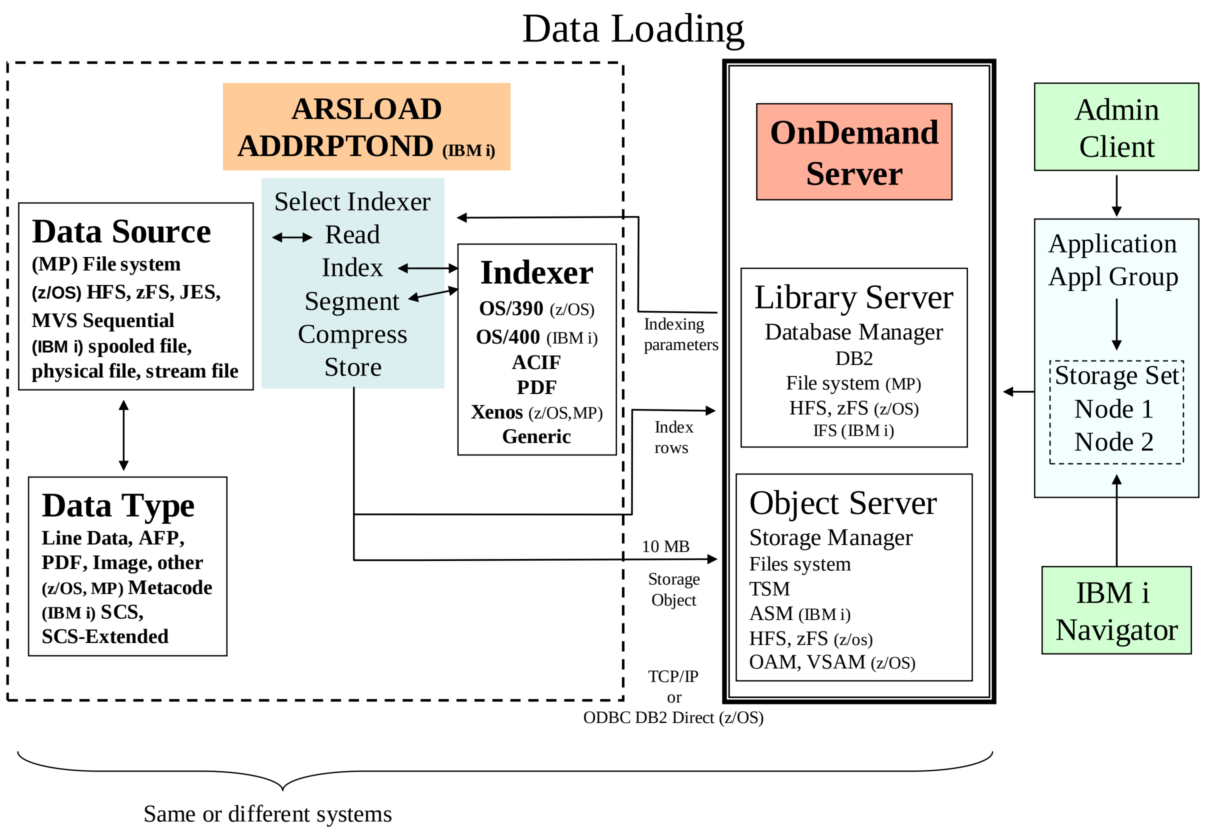


Figure 28. Data loading

Note that IBM i is packaged as a single library server/object server. It does not support separate object servers.

Before you begin loading data, you set up the CMOD definitions for your data using the OnDemand Administrator client. The setup includes creating:

- The application and application group definitions that will be used during the load process
- The storage set and nodes that will be used for storing the loaded data
- The folder definitions that will be used for retrieving the stored data

These definitions are then stored in the CMOD database on the library server.

Note: For IBM i, System i Navigator or Navigator for i (depending on the version you are running) is used for defining the storage sets and storage nodes.

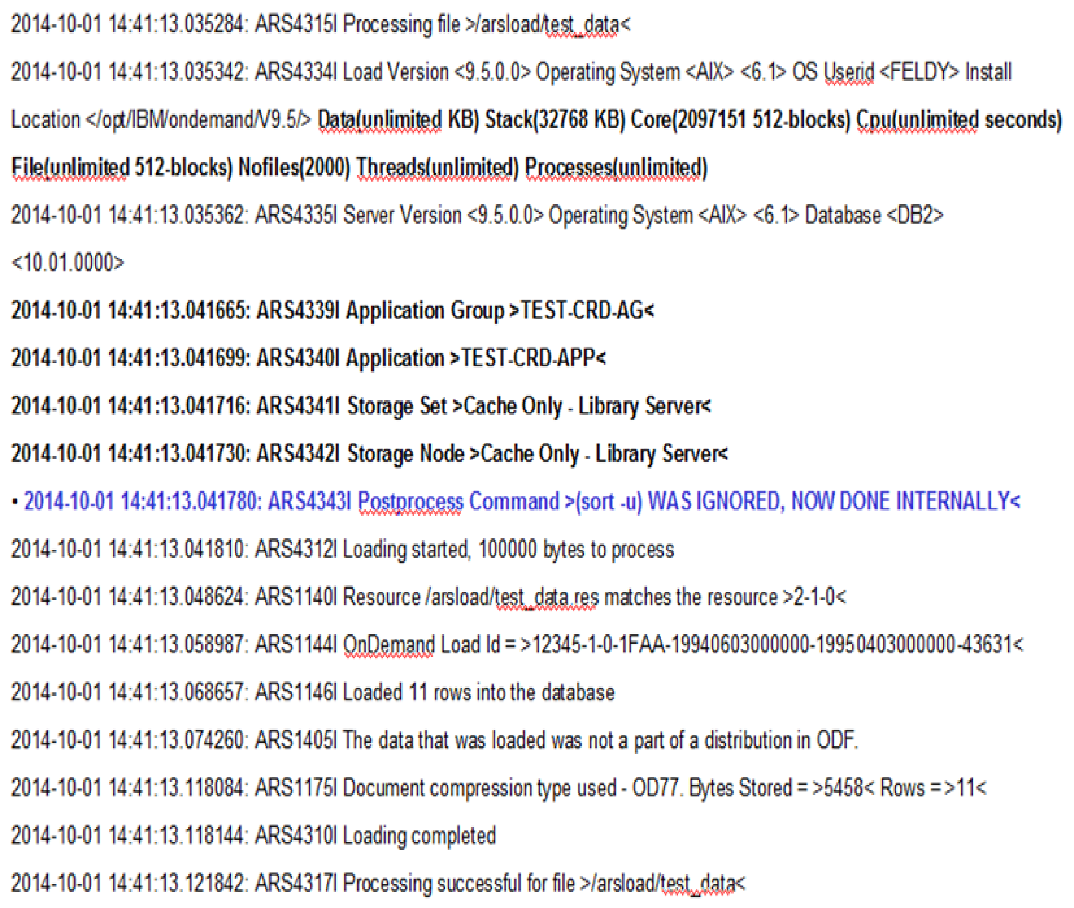
When the data loading process is initiated by using ARSLOAD (Multiplatforms or z/OS) or the Add Report (ADDRPTOND) command (IBM i) which runs ARSLOAD, it is passed the name of the data source from which to read the report data and the indexer name to be used for indexing the report.

The load program then reads the report, extracts the indexes, segments the report into multiple documents based on the index values, accumulates the documents in storage segments, compresses the storage segments, and builds a storage object. Once the storage object is built, it is transmitted to the object server for archiving and its indexes are transmitted to the CMOD server for placement in the appropriate application group data tables.

By default, the load program uses TCP/IP to communicate with the CMOD server. This provides a great deal of flexibility with regard to where the load program is run. This method of communication allows for the reports to be loaded from a separate system to the CMOD server.

The ARSLOAD program saves processing messages in the CMOD System Log. You can open the System Log folder and list the messages that were generated when an input file was processed.

In CMOD V9.5, the ARSLOAD process improves memory utilization and performance. It automatically removes duplicate rows to the same document (i.e. sort –u or sort | uniq). A sample of ARSLOAD output displays this below:



```
2014-10-01 14:41:13.035284: ARS4315I Processing file >/arsload/test_data<
2014-10-01 14:41:13.035342: ARS4334I Load Version <9.5.0.0> Operating System <AIX> <6.1> OS Userid <FELDY> Install
Location </opt/IBM/ondemand/V9.5/> Data(unlimited KB) Stack(32768 KB) Core(2097151 512-blocks) Cpu(unlimited seconds)
File(unlimited 512-blocks) Nofiles(2000) Threads(unlimited) Processes(unlimited)
2014-10-01 14:41:13.035362: ARS4335I Server Version <9.5.0.0> Operating System <AIX> <6.1> Database <DB2>
<10.01.0000>
2014-10-01 14:41:13.041665: ARS4339I Application Group >TEST-CRD-AG<
2014-10-01 14:41:13.041699: ARS4340I Application >TEST-CRD-APP<
2014-10-01 14:41:13.041716: ARS4341I Storage Set >Cache Only - Library Server<
2014-10-01 14:41:13.041730: ARS4342I Storage Node >Cache Only - Library Server<
• 2014-10-01 14:41:13.041780: ARS4343I Postprocess Command >(sort -u) WAS IGNORED, NOW DONE INTERNALLY<
2014-10-01 14:41:13.041810: ARS4312I Loading started, 100000 bytes to process
2014-10-01 14:41:13.048624: ARS1140I Resource /arsload/test_data.res matches the resource >2-1-0<
2014-10-01 14:41:13.058987: ARS1144I OnDemand Load Id = >12345-1-0-1FAA-19940603000000-19950403000000-43631<
2014-10-01 14:41:13.068657: ARS1146I Loaded 11 rows into the database
2014-10-01 14:41:13.074260: ARS1405I The data that was loaded was not a part of a distribution in ODF.
2014-10-01 14:41:13.118084: ARS1175I Document compression type used - OD77. Bytes Stored = >5458< Rows = >11<
2014-10-01 14:41:13.118144: ARS4310I Loading completed
2014-10-01 14:41:13.121842: ARS4317I Processing successful for file >/arsload/test_data<
```

Figure 29. Additional ARSLOAD output

CMOD V9.5 also supports the ability to prevent data from being loaded multiple times if one of the loads encounters a severe or catastrophic error. This is done by using the ARSLOADWORK table, which is new in version 9.5.

8.2 Initiating the load process

8.2.1 Multiplatforms

The load process can be initiated in one of two ways:

1. As a batch command (job)
    - Using the ARSLOAD command
    - Set the task priority appropriately
2. As a daemon (started task)
    - You can configure the ARSLOAD program to run as a daemon (UNIX servers) or service (Windows server) to check specified file systems for input files to process.

The ARSLOAD program can use the following sources for input files to process:

- one or more file systems specified with one or more -d parameters
- one or more load file names

If you do not use the load file name, the ARSLOAD program will run in daemon mode and attempt to load input data from the directories that are specified by the –d parameter. If you do not use the load file name and do not specify the daemon mode parameter (-d), the ARSLOAD program will issue a usage note and exit.

Important: When running the ARSLOAD program in daemon mode, the .ARD and .PDF file name extensions are required to initiate a load process.

8.2.2 z/OS

The load process can be initiated in one of three ways:

1. As a batch job
    - Verify the availability of initiators
    - Use 0 MB on the job card
    - Make the task non-swappable
    - Set the task priority appropriately

2. As a started task**,** loading data from the JES SPOOL or a zFS (hfs) directory

3. Using ARSYSPIN to accumulate multiple JES reports into a single load job

8.2.3 IBM i

The load process can be initiated in one of three ways:

1. To store a single report, run the Add Report (ADDRPTOND) command

The Add Report (ADDRPTOND) command is the primary CMOD for i data indexing and loading command. The ADDRPTOND command determines if the input data needs to be indexed, and if it does, calls the indexing program. The ADDRPTOND command then processes the index data, adds it to the database, optionally compresses the input data into storage objects, and moves the storage objects to the object server.

You run the ADDRPTOND command each time that you want to load data into CMOD. You can either run the command from the command line or use the CMOD output queue or directory monitor to periodically check for input data to process.

2. To store multiple files as they arrive in an output queue or directory, use an output queue or directory monitor

You can use the Start Monitor (STRMONOND) command to start a monitor program for any output queue or directory to receive spooled files or IFS files for processing. The monitor continuously checks for input files, and allows CMOD to capture them as they arrive.

You typically define and then manually store reports with the ADDRPTOND command during testing. Then, when the definitions are ready for production, you can automate storage with the CMOD output queue or directory monitor. You can use the *SPLFNAME, *FORMTYPE, *USERDATA, *JOBNAME, or *USRDFNxxxx spooled file attributes or the parts of the IFS filename to match the application group and application names you defined to CMOD.

To begin capturing data automatically when an input file arrives in a particular output queue or directory, you can issue the STRMONOND command. The monitor runs continuously until a specified end time or number of hours occurs, or until no input files are present in the output queue or directory. You can also use the End Monitor (ENDMONOND) command to stop processing. For every input file that the monitor has processed, there will be either a successful load (message number 87) or a failure (message number 88) added to the system log. Successful loads will also add an entry to the system load facility.

You might choose to add the STRMONOND command to your system startup program so the monitor(s) start each time you IPL the system. When starting a monitor (using the STRMONOND command) from a job scheduler, you might be unsure of what job description to use. In most cases, STRMONOND will work best using the QOND400 job description. You might have special system needs that require the use of your own job description, but this IBM-supplied job description will work successfully for most customers. See the online help for more information about these commands and their parameters.

3. To store a single file or to run as a daemon (similar to a monitor) on a directory, you can also use ARSLOAD.

8.3 OnDemand Administrator client - application parameters

CMOD application definitions specify the structure and format of the data to be indexed as well as the fields to be extracted from which the CMOD index data is created.

8.3.1 Application/General tab

Application ID Field - If you wish to define more than one application to an application group, you must define an application ID field in the application group. IBM recommends that if you are not sure whether you will be adding more applications in the future, you should define the application ID field when the application group is created. For example, a new version of the application might be created when the indexer parameters change. Having an application ID field already defined in the application group simplifies this process.

8.3.2 Application/ Indexer Information tab

Index definitions - It is important that you define the optimum number of application group indexes to meet your business requirements. Too few indexes will impact the users' ability to search reports, while too many indexes might slow down the system's load performance and increase disk space utilization. CMOD index fields cause DB2 to create a corresponding DB2 index. CMOD filter fields do not have corresponding DB2 indexes. Application groups with only filter fields defined will always result in DB2 table scans when performing user searches.

Notes:

- IBM recommends that you define at least one index for each application group. That index should be the most commonly used search criteria.
- When creating an application group by using the Report Wizard, the default for database field type is **Index**. When creating an application group directly, the default for database field type is **Filter**. You should ensure that your most frequently used search criteria have a database field type of **Index**.
- After the application group is created, you can change the database field type from Filter to Index or from Index to Filter. You must close the application group table for the change to take effect.

Page-level index - This parameter is used only for large objects. It is used to display page-level index information within the document. This allows for easier navigation within the document. The page-level index information is not stored in the database and, therefore, cannot be used to search for and retrieve documents. Only group-level indexes can be stored in the database. Page-level indexes are stored with the document. After retrieving a document, the user can use the page-level indexes to move to a specific page in the document. If the parameter is not specified during loading, the page-level information is not displayed.

Use of page-level indexes for large object reports might result in users downloading fewer large object segments, which will reduce the load on the system and thus indirectly improve performance.

8.3.3 Application/Load Information tab

You might select to compress data and/or resources that are loaded into CMOD. Compression is performed in software, which allows for the compressed documents to be decompressed on the client thus offloading CPU cycles during the retrieval process.

Compression - CMOD supports many different compression algorithms. **OD77** is the recommended compression algorithm. **OD77Lite** is a variant on OD77 that utilizes less CPU but achieves less compression. However, the compression achieved is still typically greater than that achieved by other compression methods.

- Setting **Compression** to **Disable** causes CMOD to not compress data during the load process and to not compress data for transmission over the network. This option is suitable for files that are already compressed, for example TIFF images or PDF files.
- Setting **Compression** to **None** is generally discouraged. This setting will cause CMOD to not compress data for storage during the load process. During the retrieval process, CMOD will compress the data during transmission over the network.

IBM recommends that you select the compression type most suitable for a specific application based on testing of your sample data and determining an appropriate balance between load and retrieval CPU consumption, network usage, and disk storage requirements. To determine the best compression method for a particular type of input data, you might compress a sample of the data using each of the four compression methods provided by CMOD (LZW12, LZW16, OD77, and OD77Lite). Then, compare the compression ratio (original file size/compressed file size), the time required to compress and decompress the data, and the amount of CPU consumed to determine the best compression method for your data.

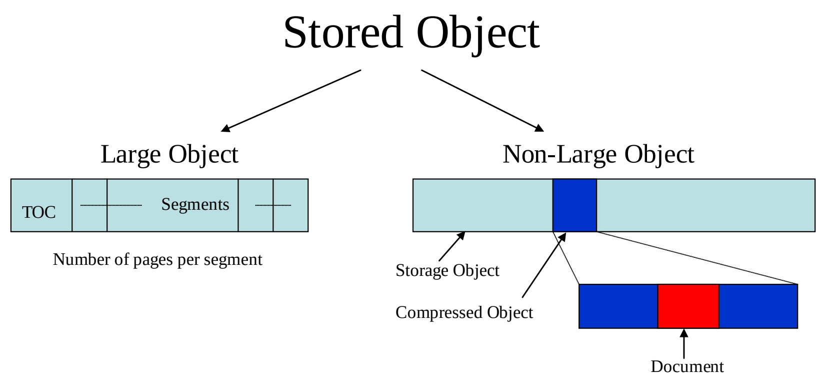
To calculate your compression ratio, look in the CMOD System Load folder for the entry that corresponds to the data you loaded for testing. The system load entry contains both the input file size and the output file size, as well as information about compression type and resource file sizes. Divide the size of the input file by the size of the output file. The result is the compression ratio achieved, such as 10:1.

With Multiplatforms or z/OS, the ARSADMIN COMPRESS function can also be used to estimate the amount by which CMOD can compress your data. For more information on ARSADMIN COMPRESS, see the Content Manager OnDemand Administration Guide for your platform. With IBM i, since the most common input file type is a spooled file, the recommended way to estimate your compression results is to store a sample spooled file and calculate the compression ratio by examining the system load as described above.

Large objects

CMOD allows you to store documents using a “Large Object” storage architecture. This is illustrated in the figure below. In the non-large object case, the storage object (10 MB by default) is composed of multiple compressed objects (100 KB by default). Each of the compressed objects contains data for multiple documents. If, for example, each document is 4 KB compressed, then the compressed object would contain 25 documents. The indexes for the documents are kept in the application group data tables.

In contrast, a large object document is a document that is very large in size (in the megabyte range), such as a telephone book. In this case, the indexes to the different sections of the phone book are stored in a table of contents that is prepended to the data. Further, the telephone directory is divided into (for example) 100 page segments. When the client application requests to view the large object/document/telephone directory, the table of contents and the first segment are downloaded. If the client then requests the next segment or requests to display page 2324, for example, then the segment containing that page is downloaded. Downloading the large object on a segment by segment basis allows for only the segments that will be viewed to be downloaded and reduces the workload on the TCP/IP network.



Setting	Where set
• Storage object size (10M)	Application group
• Large object	Application
• Compressed object size (100k)	Application
• Compression	Application
• Resources	Application

Figure 30. Storage object structures

Large object - The indexing program generates a large object by logically dividing very large documents into smaller parts and defining the indexing information that is used to retrieve the documents. When your users work with large objects, they should be able to retrieve documents faster with less impact on the network.

If **Large Object** is selected, then specify the **Number of pages.** This specifies the number of pages that will be retrieved to the client each time a large object segment is requested. For example, suppose that a document contains 10,000 pages. Using large object support, you divide the document into parts that contain 100 pages each. When users retrieve one of the documents, CMOD sends only the first part of the document to the client. Other parts of the document are automatically sent to the client when the user moves to different pages in the documents. Documents that require a large amount of storage (even when compressed) can also benefit from large object support.

If **Large Object** is not selected, then specify the **Compressed object size**. This determines the number of bytes in a fixed-size block of data stored on the system. By default, CMOD compresses input data into 100 KB blocks. You can specify a number from 1 to 99999. However, IBM recommends that you accept the default value. Specifying too small of a value can result in less than optimal document compression. Choosing too large of a value can result in less efficient document storage and retrieval. The value of the compressed object size must be less than or equal to the size of the storage object for the application group to which the application belongs.

8.4 AFP and PDF resource management

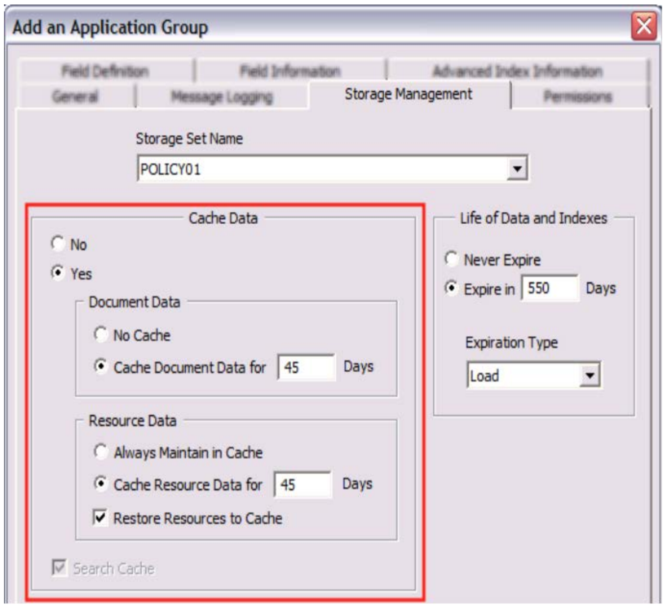
8.4.1 Resource caching

AFP resources

Prior to CMOD server version 8.4.1, if document data was cached, then AFP resources were stored in cache forever. At version 8.4.1 and higher, the cache life of AFP resources can be set separately from document data. The duration of the AFP resources in cache must be equal to or less than the **Cache Document Data for Days** value, as shown in the figure below.

PDF resources

Also at server version 8.4.1 and higher, PDF resources can be stored separately from the data. As with AFP, the cache life of PDF resource data can be set separately from the cache life of the document data. Separating the data and the resources provides the ability to store the resources only once for the application group (unless the resources for that application group change). The duration of the PDF resource data in cache must be equal to or less than the **Cache Document Data for Days** value, as shown in the figure below.



Storage Set Name: POLICY01

Cache Data: No [✓ Yes]

Document Data: No Cache [✓ Cache Document Data for 45 Days]

Resource Data: Always Maintain in Cache [✓ Cache Resource Data for 45 Days]

[✓ Restore Resources to Cache]

[✓ Search Cache]

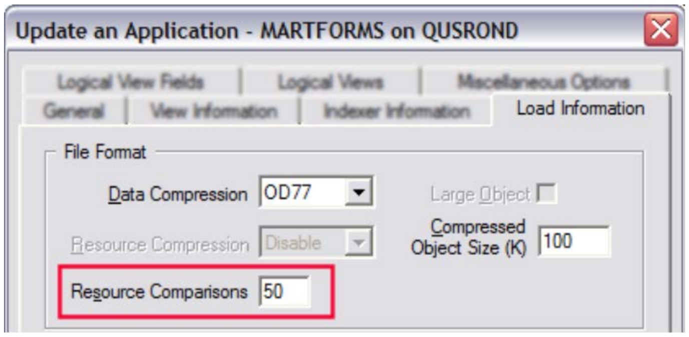
Life of Data and Indexes: Never Expire [✓ Expire in 550 Days]

Expiration Type: Load

Figure 31. Resource caching options

8.4.2 Resource comparison

Server version 8.5 added the ability to specify how many resource group comparisons to perform when a new file is archived. After comparing the size of the previously archived resource groups, if the current resource group has a different size, CMOD stores the new resource group. The allowable range is from 0 to 9999, and the default value is 50, as shown below.



File Format

Data Compression: OD77

Resource Compression: Disable

Resource Comparisons: 50

Large Object: ☐

Compressed Object Size (K): 100

Figure 32. Resource Comparison

8.4.3 Performance impact

At retrieval time, the reinsertion of the resources into the AFP or PDF documents behaves differently.

- For PDF documents, the resources are merged into the document on the server before it is downloaded to the client.
- For AFP documents, the resources and the AFP data are downloaded separately and the client handles the merging of the AFP resources into the AFP document. Additionally, some clients have the ability to store the AFP resources locally. In this case, the client will first search locally for a resource and if it is not found, will then request it from the CMOD server.

This difference in behavior means that, in relative terms, downloading an AFP document requires less system resources as compared to a similar PDF document.

Note 1: Compression of PDF resources is not recommended. Typically, the resources are already compressed in PDF documents.

Note 2: For PDF retrieval on z/OS, the following is recommended:

1. As noted above, the resources are merged into the document on the server. The mechanism for this on z/OS is via ARSSOCKD spawning a separate Unix Systems Services address space. In a CPU constrained system, this spawned subtask should be given the same WLM characteristics as the ARSSOCKD started task itself.

2. You might wish to give the ARSSOCKD address space a name that is seven characters long or shorter. The reason for this is that these spawned subtasks will have a number appended to the end of the name of ARSSOCKD and this will allow easier visual recognition of the subtasks versus the main task For example, if the ARSSOCKD task is called ARSSOCK, then these subtasks will have names like ARSSOCK1 or ARSSOCK3, etc. There is no other reason for this recommendation other than usability.

8.5 Bypassing TCP/IP - z/OS

There are two different techniques for loading data. You can either load data using TCP/IP, in which case you can load the data from any system connected via TCP/IP to the CMOD library and object servers, or alternatively you can bypass TCP/IP and store data directly to the CMOD library and object servers (directly using DB2 and OAM/TSM), in which case you must load the data on the same system where the data is being stored.

This is illustrated in the following figure.

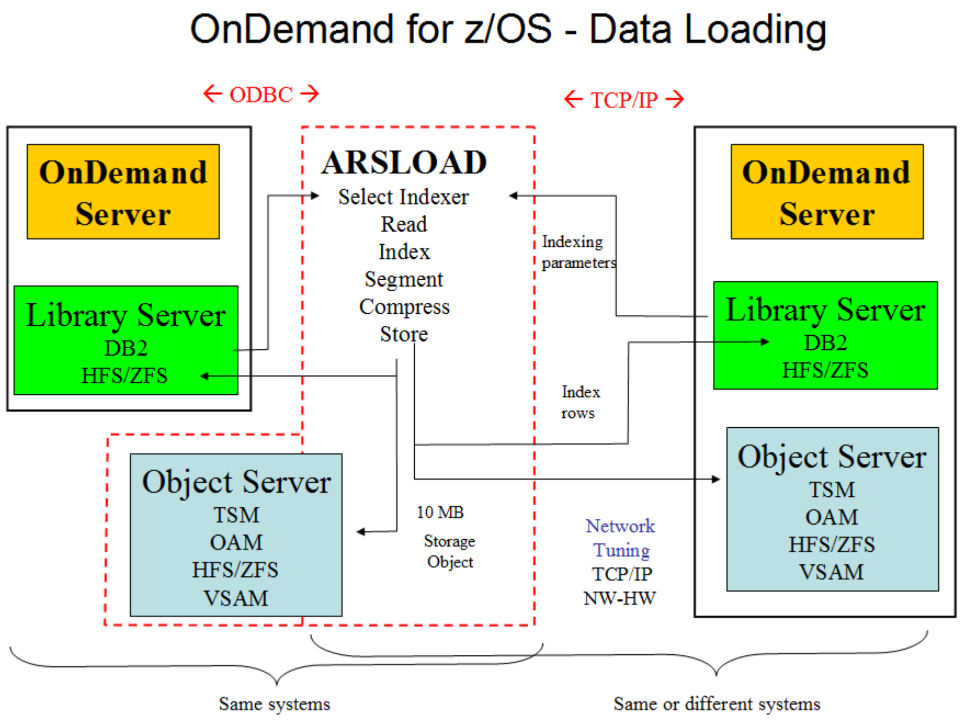


Figure 33. CMOD for z/OS - Data loading

By default, ARSLOAD will use TCP/IP to communicate with the server. This provides a great deal of flexibility with regard to where ARSLOAD is run and under which RACF user and group IDs it is run. This method of communication is advantageous when the reports to be loaded are on a separate system from the CMOD library server.

It is also possible for ARSLOAD to store data directly into DB2, OAM/TSM, or both. This method would be advantageous when:

- the reports to be loaded are on the same system as the CMOD database
- the TCP/IP network is already overloaded
- you want to offload work from the CMOD server

To enable storage without the use of TCP/IP, the following conditions must be met:

- ARSLOAD must be running with the same RACF user and group ID as the server.
- ARSLOAD must be running on the same system as the ARSSOCKD server.
- The -h parameter must specify the instance name of the server in the ars.ini file and not a host name. The instance name must also be 16 characters or less.
- A DSNAOINI DD must be present.

ARSLOAD can be forced to use TCP/IP even if all the above are true, by specifying the ARSMVS_ARSADMIN_USETCPIP=1 environment variable in the ars.cfg file.

If one or more of the above conditions are not met, ARSLOAD will use TCP/IP regardless of the setting of ARSMVS_ARSADMIN_USETCPIP.

Notes:

- If OAM is setup to store data on local disk, then storing a duplicate set of documents in the CMOD cache might not provide a performance advantage.
- Generally speaking, the OS/390 indexer is faster than the ACIF indexer. This speed difference will vary and is a function of the data, the report and document size, and whether the data is Line or AFP data.

- For the best overall performance, the indexer type needs to be matched to the type of data to be loaded, the input processing prior to document storage, and the preview processing during the document viewing process.
- When using JCL to run the ARSLOAD program to capture reports:

  1. Specify the `-s ddname` parameter to indicate the DD statement that points to the input report file that is being captured.
  2. Specify the name of a temporary file as the last parameter. The ARSLOAD program uses the temporary file for work space during the load process. The directory to be used by ARSLOAD for temporary files is determined in the following order:
     - The -c option in the ARSLOAD parameters
     - The environment variable ARS_TMP
     - The environment variable TEMP
     - The environment variable TMP
     - The current working directory if none of the above are specified
  3. Specify the -*h instance* parameter to identify the name of the instance that you want to process. This is especially necessary if there are parameters specified in the ars.cfg file that are relevant to the load process.
- The directory used by ARSLOAD for temporary files will need to be analyzed to make certain there is adequate additional space to hold the largest document that might be captured.
- The OS/390 indexer provides CMOD large object support for line print and AFP reports. In general, large object can be used for all line print and AFP reports but should be considered for most documents that exceed 100 pages in size. Non-large object documents are limited in size by available processor storage. The entire document will be stored in memory during the load and retrieval processes. If you have a large numbers of users retrieving documents at the same time, you need enough memory to store all of those documents. Excessively large documents might result in high storage requirements and significant paging activity and should be considered as candidates for large object.

8.6 Factors that affect load performance

The factors that affect the load performance are:

- Hardware: quantity and speed of CPUs, disk, I/O channels, and memory
- Network bandwidth and throughput (if using TCP/IP) for data loads
- Ultimately, the performance is limited by the speed and capacity of the available hardware (CPU, memory, disk, network, and so forth)
- Operating system tuning components: DB2, TCP/IP, Language Environment® (LE)
- CMOD tunable components
- Storage management tunable components USS, zFS, HFS, OAM, TSM, ASM
- Data components:
  - Report file size, document file size (or in the case of Large Objects, report segment size), number of documents per report
  - Number and distribution of triggers, fields and indexes per document
  - Data type and required data conversion (if any)
  - Resource collection for AFP and PDF
  - Document compressibility, a function of document data complexity and data type. Text (such as Line Data or SCS) is typically more compressible than AFP, which is typically more compressible than PDF

    - Storage object size (10 MB default) – contains 100 KB compressed object – contains compressed document
    - Exit routines/programs

8.7 Recommendations

- For Multiplatforms and z/OS, run parallel load jobs to take advantage of multi-processors, large memory pools, multiple data paths, and multiple disk drives.
    - Ensure that each parallel load is loading to a different application group.
    - Ensure that you setup a different temp directory for each of the parallel loads.   The `–c indexDir` indexer parameter (which specifies the directory in which the indexer stores temporary data) should always be specified for ARSLOAD and should be unique for each executing ARSLOAD process.
- For IBM i, start multiple output queue monitors over a single output queue to improve throughput and take advantage of multi-processors, large memory pools, and multiple disk drives.
- Each CMOD process is limited by the performance of a single processor. For example, the OS/400 indexer will use only one processor when indexing a document. Having two or more processors in your system or LPAR will not improve the performance of the OS/400 indexer. However, having two or more processors in your system or LPAR might allow you to run multiple loads jobs simultaneously. You can start multiple output queue monitors over a single output queue to improve document load performance.
- For IBM i, the use of the Merge Spooled Files (MRGSPLFOND) command can provide significant performance improvements when loading SCS spooled files. See the Merge Spooled Files section of this document for more details.
- For IBM i, depending on your retrieval patterns and system hardware configuration, it might be advantageous not to store a duplicate set of documents in the CMOD cache when using ASM given that ASM might already be using disk space. If the application group uses ASM, caches the data, and specifies to migrate data at load time, two copies of the data are stored during the load. One copy is stored in cache and one copy is stored in the ASMREQUEST directory.

  To avoid storing a duplicate set of documents in cache for non-AFP data, change **Cache Data** to **No** on the Storage Management tab of your application group definition. To accomplish this for AFP data, you might change **Document Data** to **No Cache** but leave **Resource Data** in cache for faster retrieval.
- For IBM i, every user loading data should have a home directory. If they do not have a home directory, the temporary files are stored in the root directory of the IFS.
- If the data source is on a remote system, it is possible to either load the data into CMOD on the remote system and directly store the export data to the specified CMOD library and object server, or upload the data to the specified CMOD server via FTP, then load the data on the selected CMOD system.
- For Multiplatforms and z/OS, all file systems should be dedicated file systems mounted on their own mount points.
- For z/OS, if ARSLOAD is running without TCP/IP, the -K parameter allows the DB2 connection to persist, potentially improving the load performance. This causes ARSLOAD not to detach ARSADMIN. Normally, ARSLOAD attaches and detaches ARSADMIN several times for each file loaded. This might improve performance by not requiring the LE enclave associated with ARSADMIN to be constantly created or destroyed.

  Important: Do not specify the -K parameter if you are using any CMOD for OS/390 Indexer exit routines until you verify that the exit routines function correctly in this environment. In particular, the LE enclave is no longer terminated between reports. This impacts exit routines that rely on enclave termination to perform cleanup, for example, closing files.
- For z/OS, when loading PDF reports (using the PDF indexer), placing the input report in the HFS/zFS causes the load to run nearly 50 times faster as compared to the input report being placed in a VSAM file.

9 Data retrieval

9.1 Data retrieval parameters

9.1.1 OnDemand Administrator client - folder parameters/General Tab

Note Search - If the **Annotation flags in document database** is set to **No** on the Application Group/General/Advanced tab, then this option determines when CMOD searches the database for annotations and notifies users that annotations exist for the documents that match a query. CMOD provides three search/notification methods: **Hit List**, **Retrieve**, and **Note**.

- **Hit List -** CMOD searches for annotations when the user runs a query. When annotations exist for a document, the client programs display a note icon next to it in the document list. This method has a direct performance impact on the generation of the document list.
- **Retrieve** - CMOD searches for annotations when the user selects a document for viewing. This is the default and recommended value.
- **Note -** CMOD searches for annotations when the user chooses the Note command while viewing a document.

9.1.2 OnDemand Administrator client - folder parameters/Permissions tab

Max Hits - Determines the maximum number of hits that will be retrieved and transmitted to the client. By reducing the maximum number of hits (within business reason) users are forced to enter queries that will better match the documents that they are searching for. This results in a more optimum usage of the system resources both in performing the queries and in downloading the resulting document list.

9.1.3 TCP/IP considerations

Working with the CMOD connector TCP/IP tuning and sockets

During repeated searches and retrievals on a CMOD server, many Windows sockets are opened and closed. Two default Windows settings might impact heavy traffic between the client and the CMOD server.

- When an application closes a Windows socket, Windows places the sockets port into TIME_WAIT status for 240 seconds; during this time the port cannot be reused.
- Windows limits the number of ports that an application can use to 5000.

To avoid the problems that might result, change the values for the timeout wait time and number of ports using the Windows registry editor.

- Change the value of the timeout wait time from 240 seconds to a lower number (the valid range is 30-300 seconds). The key's name is HKEY_Local_Machine\System\ CurrentControlSet\services\Tcpip\ Parameters\TcpTimedWaitDelay.
- Increase the maximum port number from its default of 5000 to a higher number (the valid range is 5000-65534). The key's name is HKEY_Local_Machine\System\CurrentControlSet\ services\Tcpip\Parameters\MaxUserPort.

For more information on TcpTimedWaitDelay and MaxUserPort, consult your Windows documentation.

There are similar problems you might face in AIX or Linux when performing repeated searches and retrievals. The default settings for these systems might also impact heavy traffic between the client and the CMOD server. To avoid the problems that might occur, you can change the values for the timeout wait time and number of ports.

Timeout Wait Time

- For AIX, you may need to change the timeout default of the tcp_finwait2 from 1200 half seconds (10 minutes) to a smaller value (recommendation is 120). This will reduce the time the operating system holds on to the socket before releasing so the next incoming connection can use it.

- For Linux, you may need to change the tcp_fin_timeout default setting to a smaller value to reduce the hold time on the sockets.

User Ports

- For AIX, you may need to increase the maximum port numbers for available connections. This is performed by adjusting the tcp_ephemeral_high and tcp_ephemeral_low default settings.

- For Linux, you may need to adjust the port ranges by altering the ip_local_port_range setting. The minimum port is normally set at 32768 as a default and the maximum port is normally set at 61000. Suggested settings are 10244 and 65535.

Two other parameters that you may want to check are the TCP buffer sizes and the TCP nodelay.

The following is an example of settings in CMOD on z/OS that you can configure in the ars.cfg file:

```
ARSSOCK_SNDBUF_SIZE=262144
ARSSOCK_RCVBUF_SIZE=262144
ARSSOCK_TCP_NODELAY=1
```

These parameters can also be set in your operating system network settings. You should verify with your network personnel that the values that are appropriate for your environment are being set correctly.

9.1.4 Factors that affect retrieval performance

Resources available to the CMOD server:

- **Disk and I/O capacity** - Each retrieve requires that the data be obtained from the archive (TSM, OAM, ASM, cache...). This data is physically located on disk or some other storage device. The storage device retrieval rate is part of the total response time observed at the client.
- **Real memory -** The data retrieved from disk needs to be stored in memory in order for it to be processed. Virtual memory allows for large amounts of data to be swapped in and out of real memory but does not remove the need for real memory.
- **CPU -** Any data transformations performed on the CMOD server require available CPU. If the CPU is not available, the server will wait until it becomes available. This will cause the total response time to the client request to be lengthened.
- **Concurrent retrievals -** Each retrieval requires resources on the server. The higher the number of concurrent retrievals, the larger the amount of resources that are needed in order to complete the work in an acceptable amount of time.
- **Network bandwidth -** The retrieved data is sent to the clients over the TCP/IP network. If the network bandwidth is not wide enough to satisfy all the concurrent requests, then the response time to the clients will be slower and data will be queued up in the server buffers, further slowing down the system.

10 Expiring data

10.1 Overview

In order to improve the efficiency of the storage process, CMOD aggregates the stored documents (typically a few kilobytes in size) into storage objects (typically 10 MB or larger in size). This aggregation allows for the enablement of extremely efficient, high volume storage, retrieval and expiration mechanisms. The storage objects can be stored in cache and/or directly in a storage manager subsystem (TSM, OAM, ASM). Most customers expire their data at some point in time. Typically, customers choose to expire data somewhere in the 5 to 10 year range. In the extreme cases, data might be expired daily or never expired. There are four typical storage and expiration scenarios that support the various customer expiration requirements. These are illustrated in the following figure.

Storage & Expiration Scenarios

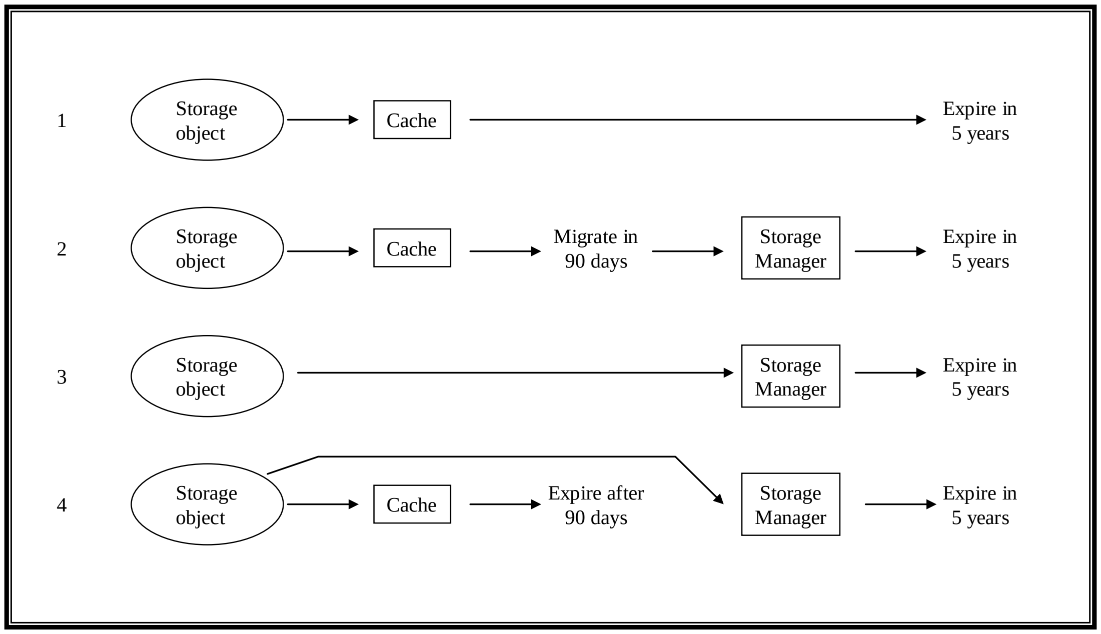


Figure 34. Storage and Expiration Scenarios

Scenario 1: The storage object is stored to cache only and is the expired from cache after some predetermined time period. Typically this methodology is employed when 1) the life of the data is short and there is no need for hierarchical storage management; or 2) the life of the data is long and the customer has a mechanism for backing up the data from cache.

Scenario 2: The storage object is first stored to cache for a short period of time after which it is migrated to a "storage manager" for long time storage. Typically this methodology is employed when 1) most of the data access will occur during the first time period and then after the data is migrated to the storage manager it will be infrequently if ever accessed; and 2) there is a performance difference when retrieving the data from cache versus retrieving it from the storage manager (which could occur if the storage manager were on a device separate from the CMOD object server).

Scenario 3: The storage object is stored directly to the storage manager. Typically this methodology is employed when 1) the performance of the storage manager equals the performance of the local file system; and 2) hierarchical storage management is beneficial. An example of this is on z/OS systems where storing directly to OAM is the most popular solution.

Scenario 4: The storage object is stored directly to both cache and the storage manager. After a short period of time the data is expired from cache and then after a much longer period of time the data is expired from the storage manager. Typically this methodology is used when 1) the cache file system allows for more efficient data retrieval; and 2) there is a need to keep the data for a longer time period; and 3) the hierarchical storage management (or other features) of the storage manager are required.

Note: Whether the data is stored in cache or in a storage manager, the main performance differences are a result of 1) the hardware speed (and I/O channels/interfaces) on which the data is stored on; and 2) the location of the hardware device with respect to the object server. If the device is connected over a TCP/IP link, then that link could possibly form a bottleneck depending on the link's throughput and the data retrieval rate required.

10.2 Expiration parameters

ARS_EXPIRE_REQLIMIT - This parameter is used if storage manager expiration has been specified. It is the number of load IDs that ARSADMIN will send to the server during expiration processing in a single request when using storage manager based expiration. Recommendation is to set it at 25. This parameter does not apply to IBM i.

10.2.1 Application Group/Storage Management tab

Storage Set Name - Determines where the storage manager maintains reports and resources loaded into the application group. The storage manager can maintain data in cache storage, or in archive storage (OAM, TSM or ASM), or in both cache storage and archive storage.

The storage set must support the number of days that you plan to maintain reports in the application group. For example, if you need to maintain reports in archive storage for seven years, then the storage set must identify a storage node (or migration policy, on an IBM i server) that is maintained by the archive storage manager for seven years.

The storage set must support the type of media required to hold reports stored in the application group. For example, if you need to maintain reports in cache storage for 90 days and in archive storage for seven years, then the storage set must identify a storage node (or migration policy) that causes the archive storage manager to maintain the data for seven years and you must select **Cache Data for __ Days** and enter 90 in the space provided.

If you do not need to maintain reports in archive storage, then select a cache-only storage set and type a number in the **Cache Data for __ Days** field. When you select a cache-only storage set, CMOD automatically sets **Migrate Data from Cache** to **No** and sets the **Expire in __ Days** field to the same value as the **Cache Data for __ Days** field. (The default value is 90 days.) IBM does not recommend using a cache-only storage set because this limits your future options for managing your data.

If you do not need to maintain reports in cache storage, then select a storage set that identifies a storage node (or migration policy) that is maintained by the archive storage manager and set **Cache Data** to **No**. CMOD automatically sets **Migrate Data from Cache** to **When Data is Loaded**.

Important: If the storage set contains cache-only storage nodes, ensure that the **Cache Data** value and the **Life of Data and Indexes** value are the same. Otherwise, the add or update operation cannot be completed.

Cache data - Determines whether CMOD stores data in cache storage. If the storage set is a **Cache only** storage set, then **Yes** is the only selection. If the storage set is an archive manager-controlled storage set (OAM, TSM, or ASM), then you have the option of additionally storing the data in cache. Typically, in a z/OS environment, data is **not** stored in cache. CMOD for z/OS customers typically use OAM as their storage manager. OAM supports the ability to store the data directly in DB2 where the storing and retrieval rates are exceptionally fast, which eliminates the need to maintain and monitor cache file systems in zFS or HFS.

Document Data - If you select **Yes** for **Cache Data**, then you can cache document data and resource data or resource data only. If you select **No Cache**, then document data is not stored in cache. If you select **Cache Document Data in Cache for xxx Days**, then document data is stored in cache for xxx number of days before the data expires.

Resource data - If you select **Always Maintain in Cache**, then resource data stays in cache forever, and does not expire. If you select **Cache Resource Data in Cache for xxx Days**, then resource data stays in cache for xxx number of days before the data expires. If you select **Restore Resources to Cache**, resources are restored to cache from the storage manager if resource data is not in cache and there is a request for the resource data.

Life of data and indexes - Determines when CMOD deletes documents, resources, and index data from the application group. The available options are:

- **Never Expires:** CMOD maintains application group data indefinitely.
- **Expires in __ Days:** After reaching this threshold, CMOD can delete data from the application group, the next time ARSMAINT (with CMOD for Multiplatforms or z/OS) or Disk Storage Management (DSM, with IBM i) is run. The default value is 2555 (seven years). The maximum value that you can type is 99999 (273 years).

Notes:

1. If you plan to maintain application group data in archive storage, then the length of time that the archive storage manager maintains the data must be equal to or exceed the value that you specify for the **Life of Data and Indexes**.

2. **Life of data and indexes** can only be used if ARSMAINT (with Multiplatforms or z/OS) or Disk Storage Management (DSM, with IBM i) is handling the expiration.

Expiration Type - Determines how data is deleted from the application group. The following options are available:

Load - The system deletes an input file (a load) at a time from the application group. If the **Database Organization** is set to **Single load per database table**, the system deletes a segment (table of index data and associated documents) at a time. The latest date value from the input file and the **Life of Data and Indexes** determines when the data is eligible to be deleted. **Load** is the default expiration type.

Note: If you intend to use **Enhanced Retention Management** or **Interoperate with FileNet P8 Platform,** then the application groups must have an **Expiration Type** of **Load**. For those that have expiration types of **Document**, **Segment**, or **Storage Manager**, utilities exist to convert such application groups to **Load**. IBM recommends that you engage IBM Services to provide these services.

With CMOD for Multiplatforms or z/OS, when the **Expiration Type** is set to **Load**, if your object server is on z/OS, and your storage manager is OAM, you can allow OAM to handle the data expiration and CMOD the index expiration using ARSEXOAM. With CMOD for i, when the **Expiration Type** is set to **Load**, you can still allow Archive Storage Manager (ASM) to handle the data and index expiration by creating an expire level in the migration policy.

Storage Manager - The storage manager (OAM, TSM, or VSAM) determines when data will be deleted from the system. **Storage Manager** expiration works with either the ARSEXPIR or the ARSEXOAM program. See the Content Manager OnDemand for z/OS Administration Guide, for details about how to configure system to use the ARSEXPIR and ARSEXOAM programs. **Storage Manager** expiration is supported only on CMOD for z/OS systems.

On IBM i, storage manager-based expiration is supported by adding an expire level to a migration policy.

Segment - The system deletes a segment (table) of data at a time from the application group. The system can delete a segment of data only after the segment is closed and every record in the segment has reached its expiration date:

- If the **Database Organization** is **Multiple Loads per Database Table**, the system uses the maximum number of rows to determine when to close a table. A segment likely contains the data from more than one input file. If the maximum rows setting is too big, the segment will not be expired until all the documents in the table reach their expiration date. If the maximum rows setting is too small then segments will constantly be created and potentially deleted (based on the expiration date). This large number of tables imposes a performance overhead during search query time and expiration time.
- If the **Database Organization** is set to **Single load per database table**, the system creates a table each time that an input file is loaded into the application group.

IBM recommends that you use **Multiple Loads per Database Table.** The system derives the expiration date from the Segment field (or the date that the data was loaded, if there is no Segment field) and the **Life of Data and Indexes**. If the Segment field contains a date in the MMYY format, data is eligible to be deleted on the first day of the month (MM). To specify the Segment field, click the Field Information tab; select a date or date/time field; then select the Segment check box.

Document - The system deletes a document at a time from the application group. To determine when to delete a document, the system uses the value of the **Expire Date** field and the **Life of Data and Indexes**. If the **Expire Date** field contains only the month and year (MMYY format), the system deletes documents on the first day of the month (MM). To specify the **Expire Date** field, click the Field Information tab; select a date or date/time field; then select the **Expire Date** check box. Individual document deletion is the most costly in terms of CPU consumption and runtime.

10.2.2 Data migration and expiration

If you are migrating data by means of unloading and then reloading the data, then you should consider what you want your future expiration policy to be.

For example, if your current expiration policy is set to **Storage Manager** but at some point in the future you would like to be able to perform holds on the data, then during the migration process (when creating the new application group, before loading any data) you should change your expiration policy from **Storage Manager** to **Load**.

When using Enhanced Retention Management or Enterprise Records (formerly FileNet Records Manager), CMOD must be in complete control of expiration processing. Therefore, if you are using TSM or OAM – you must disable the ability for either of these storage managers to expire data. Also, it is only possible to use enhanced retention management and CFS-CMOD against application groups that have an expiration type of **Load**. For those that have expiration types of **Document**, **Segment**, or **Storage Manager**, utilities exist to convert such application groups to **Load**. IBM recommends that you engage IBM Services to provide these services.

Data migration to other media

If you have chosen to not take advantage of the ability of CMOD to aggregate documents by instead loading documents ad-hoc using the storeDocument Java API, StoreDoc OLE API, or CommonStore, you will be required to migrate the data at a future point.

If you have chosen to not take advantage of the ability of CMOD to aggregate documents into 10 MB storage objects, this might result in millions of small objects stored in your storage manager. This might further result in the storage manager having performance problems when migrating these small objects to tape. A suggestion is to "aggregate" these smaller objects into larger ones.

For you to aggregate all these tiny objects into larger objects after they have been stored individually would require retrieving and reloading them as larger objects. This is something with which you might want IBM Services to assist you. They would use the ARSDOC GET command to retrieve groups of small objects and reload them as a single document stream. This could be very time consuming and expensive.

Another option is to not migrate them to tape, but to use some other random access hardware instead.

10.2.3 Application Group/Storage Management tab/Advanced button

Object size - The size of a storage object in kilobytes (KB). By default, CMOD segments and compresses report data into 10 MB storage objects. IBM recommends that you use the default value. Valid values are between 1 KB and approximately 150 MB. However, exercise caution when changing the value. Specifying too large or too small a value can adversely affect performance when loading data.

Migrate data from cache - This parameter determines how long the data is kept in cache before it is migrated to archive storage (on a potentially slower archive storage device). The data should be kept on a high performance storage device for the period during which it will be frequently retrieved.

From a user perspective there is no procedural difference in retrieving the data from either cache or archive storage. The only user-perceived difference is the response time. Various archive storage mechanisms provide different performance profiles. For example, when using OAM with the data stored in DB2 tables on disk, the response time is as fast as the cache response time. The main difference in response time is based on the type of disk used by either method. On the other hand, if the OAM data is stored on optical platters or tape, then the response time is increased dramatically. If you are using a network-attached TSM server then the retrieval rates (throughput and response times) would be governed by the TSM device and the TCP/IP connection.

Migration of indexes - Determines when CMOD migrates index data to archive storage. Choose from **No Migration** or **Migrate After __ Days**. It is important to understand that IBM recommends that indexes **not** be migrated to archive storage. Indexes that have been migrated are unable to be searched until after they are imported by an administrator. This capability should be used only under the limited circumstances described below.

Index migration is the process by which CMOD moves index data from the database to archive storage. Index migration optimizes database storage space while allowing you to maintain index data for a very long time. You typically migrate index data only after users no longer need to access the reports, but, for legal or other requirements, you still need to maintain the data for some number of months or years. If a user queries index data that has been migrated, then an administrator must take action to import a copy of the migrated table or tables using ARSADMIN (Multiplatforms or z/OS) or the Start Import (STRIMPOND) command on IBM i. After maintaining the imported index data in the database for the number of days specified in the **Keep Imported Migrated Indexes** field, CMOD deletes the data from the database.

A table of index data for a **Multiple Loads per Database Table** application group is eligible for migration when it reaches the **Maximum Rows** value causing the table to be closed. For a **Single Load per Database Table** application group, each time that you load data into the application group, CMOD closes that application group segment table and creates a new table.

You should migrate index data only after users no longer need to access the data. If a user needs to access data in migrated tables, then the process of importing the data into the database requires administrator intervention, and usually results in a significant delay in completing the query. Additional space is required in the database and in temporary storage areas to import the data.

To enable migration of index data, you must define a storage set that identifies a storage node that is maintained by the archive storage manager and update the System Migration application group to use the storage set.

10.3 CMOD data expiration

10.3.1 ASM, DSM and ARSMAINT

10.3.1.1 DSM and ASM on IBM i

In most circumstances you must run Disk Storage Management (DSM) and Archived Storage Management (ASM).

DSM

DSM performs the following functions:

- Controls the expiration of indexes and data from CMOD (if you do not use storage manager based expiration)
- Migrates data from cache to the storage manager (if the **Migrate Data from Cache** option is not set to **When loaded**)
- Expires data from cache if **Cache Data** is set to **Yes**

If you do not run DSM, your disk storage requirements for CMOD might be higher than expected. The number of objects stored in the Integrated File System (IFS) might also be higher than necessary, which results in longer save and restore times.

Note that if you have never run DSM, the first execution of the Start Disk Storage Management *(*STRDSMOND) command might last for an extended period of time.

If you want to configure CMOD so that DSM can be run less frequently, see the document titled “Reducing the need to run Disk Storage Manager (DSM)” on the web at http://www.ibm.com using ‘1417320’ for your search criteria.

ASM

ASM performs the following functions:

- Controls the expiration of indexes and data from CMOD (if you use storage manager based expiration)

- Aggregates data before migrating it to archive media (if you select the **Aggregation** option in the migration policy)
- Moves data between storage levels of the migration policy

If you do not run ASM, your disk storage requirements for CMOD are probably higher than expected. The number of objects stored in the IFS are also higher than necessary, which results in longer save and restore times.

Note that if you have never run ASM, the first execution of the Start Archived Storage Management (STRASMOND) command or the STRDSMOND command with the STRASMOND parameter set to *YES might last for an extended period of time.

For more information about expiring archives using ASM, see the document titled "Expiration processing in Common Server Archive Storage Manager (ASM)" on the web at http://www.ibm.com using '1317082' for your search criteria.

10.3.1.2 ARSMAINT on Multiplatforms and z/OS

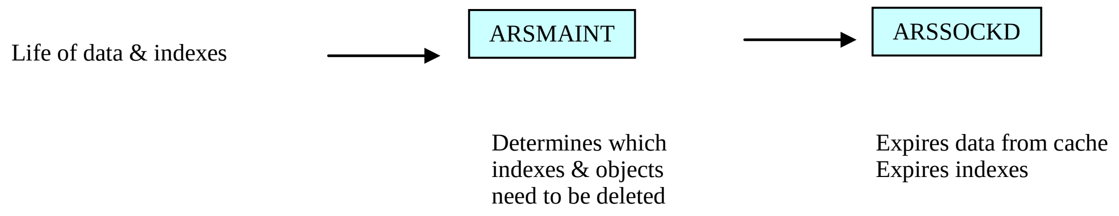
ARSMAINT manages application group data in the CMOD database and in cache storage. You typically run the ARSMAINT program on a regular schedule to migrate files from cache storage to archive storage, delete files from cache storage, optionally migrate index data from the database to archive storage, and delete index data from the database.

The application group data and the data that you have stored in cache are all managed by the ARSMAINT program. It is managed by using the storage management values from the application groups that are defined to the system. The storage management values used:

- Life of Data and Indexes
- Length of Time to Cache Data on Magnetic
- Length of Time Before Copying Cache to Archive Media
- Length of Time Before Migrating Indexes to Archive Media
- Length of Time to Maintain Imported Migrated Indexes
- Expiration Type.

The ARSMAINT program uses the Expiration Type to determine how to delete index data from an application **g**roup. The ARSMAINT program can expire a table of application group data at a time**,** a load at a time, or individual documents. You should make sure that the ARSMAINT program command runs periodically (example daily) so that CMOD deletes indexes and cache data (and the storage manager deletes archive data if applicable) when it is time to do so (so that expired documents can no longer be retrieved).

Note: Most maintenance processes should run when no other applications are updating the database or need exclusive access to the database and when you are sure that no one will be retrieving documents from the system. For example, you should not perform maintenance of the database at the same time that you load data into the system.



CMOD lets you collect statistics for all of the tables in the database with the ARSMAINT program. When you run the ARSMAINT program to collect statistics, it collects statistics on all of the tables in the database that have changed since the last time that you collected statistics. You can automate the collection of statistics by scheduling the ARSMAINT program to run with the appropriate options.

10.3.2 Storage Manager-based expiration

10.3.2.1 Storage Manager-based expiration using ASM - IBM i

For IBM i, storage manager-based expiration is the use of an expire level in a migration policy which causes ASM to delete the data, place a record into the "unload" file, and then, as the last step, use the information in the "unload" file to delete the indexes. DSM is not required to delete either data or indexes. **Note:** The Enhanced Retention Management feature is not supported when using storage manager-based expiration.

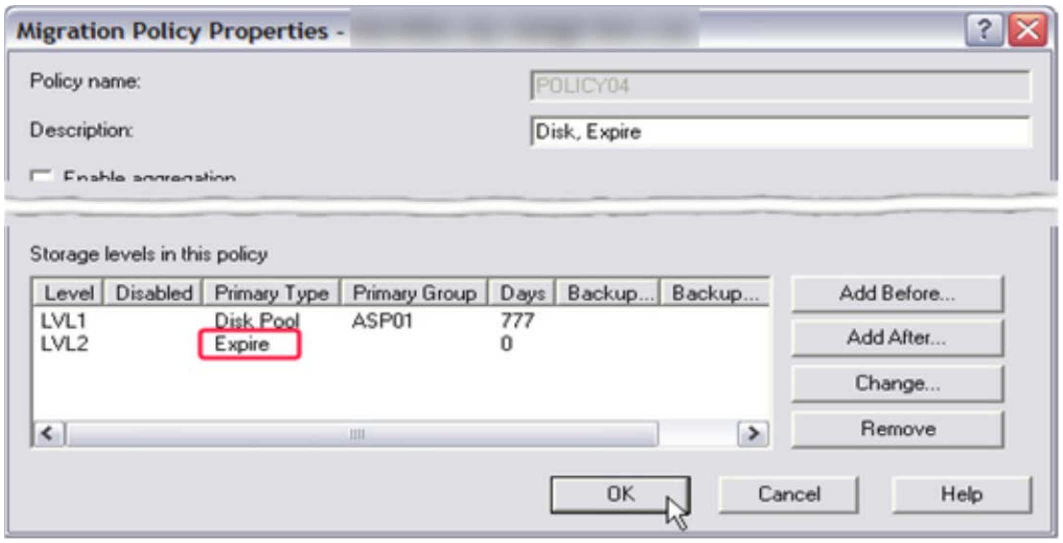
Storage manager-based expiration requires that the instance owner profile be added to the instance as a system administrator. The instance owner profile has the same name as the instance, for example QUSROND.

If you are using Tivoli Storage Manager (TSM) instead of ASM, this discussion does not apply. When using TSM, when the time specified in **Life of data and indexes** is reached, DSM deletes the indexes and sends a delete request to TSM. The next time the TSM delete processor runs, it will delete the data from TSM.

If you do not have an expire level in your migration policies, you will not see any change in how archived data is managed and expired. When the application group setting for **Life of data and indexes** is reached, DSM will delete the indexes and cause ASM to delete the data.

When using storage manager-based expiration, IBM recommends that you set the **Life of Data and Indexes** to **Never Expire** to prevent DSM from expiring data. Also, if you are using storage manager-based expiration, you do not need to run DSM for any application groups that have **Cache Data** set to **No** on the Storage Management tab.

In the example below, when ASM is run, any data that has been in ASP01 for 777 or more days will be deleted. Records are placed into the unload file, and the indexes deleted using the records from the unload file.


Migration Policy Properties -

Policy name: POLICY04

Description: Disk, Expire

☐ Enable aggregation

Storage levels in this policy

Level	Disabled	Primary Type	Primary Group	Days	Backup...	Backup...
LVL1		Disk Pool	ASP01	777		
LVL2		Expire		0		

Figure 35. Migration policy with an expire level

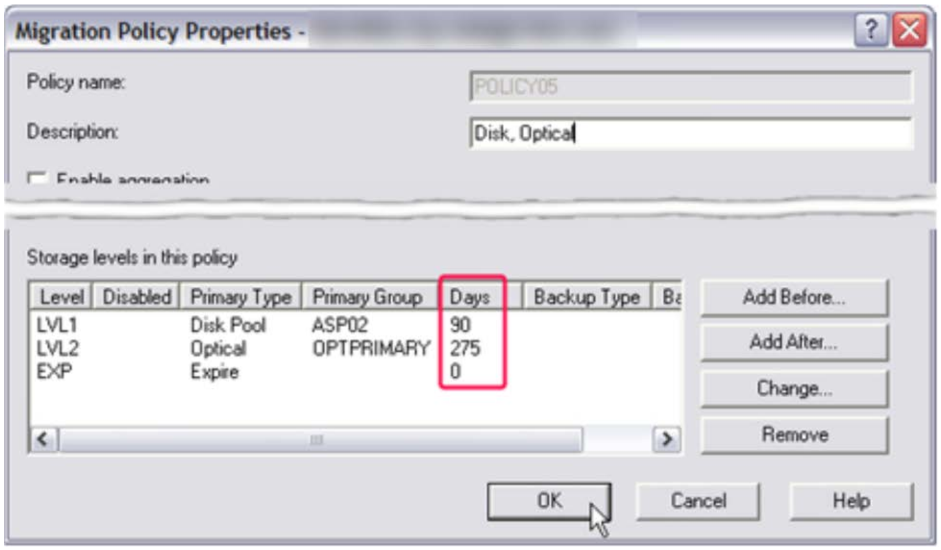
You should keep the following information in mind when using storage manager-based expiration:

- The life of archived data is based on the sum of the days in each level.
- If you do not run ASM every day, the duration in any of the levels can vary from the value specified for that level.
- In addition to the migration policy levels, data also will be kept in the 'request' level (ASMREQUEST directory) immediately after archive. If aggregation is specified in the migration policy, data will be kept in the 'aggregation' level (ASMAGGREGATION directory) until the aggregate reaches the specified size or number of days.
- The first time ASM is run the data in the 'request' level is aggregated. After aggregation the data is migrated to the first level of the migration policy. Only when the data reaches the migration policy level do the days in level start.
- After data reaches the migration policy level, that data will stay in the level until the first time ASM is run after the days value is reached. If ASM is not run daily, the data will stay in the level more than the specified number of days.

As an example, and as shown below:

- The Days setting in the first level is 90 but ASM is not run until 92 days.
- The Days setting in the second level is 275 but ASM is not run until 280 days.
- The data is then expired.

The total days in ASM will be 372 days, not the 365 day sum of the level durations.


Migration Policy Properties -

Policy name: POLICY05

Description: Disk, Optical

Enable aggregation

Storage levels in this policy

Level	Disabled	Primary Type	Primary Group	Days	Backup Type	Ba
LVL1		Disk Pool	ASP02	90		
LVL2		Optical	OPTPRIMARY	275		
EXP		Expire		0		

Figure 36. Duration at multiple levels

10.3.2.1.1 Expiring data if Segment or Document expiration is used – IBM i

If you use **Document** or **Segment** for the **Expiration Type** in any of your CMOD application groups, you might need to take action to ensure that data stored in the application group expires as expected. CMOD does not explicitly delete data stored using **Document** or **Segment** as the **Expiration Type** from Archive Storage Manager (ASM) or Tivoli Storage Manager (TSM).

Note that the default for new application groups is an **Expiration Type** of **Load**. The following information does not apply if you are using **Load** expiration. With an **Expiration Type** of **Load**, CMOD expires data at the appropriate time regardless of where it is located. Note that the System Log application group uses **Segment** expiration.

The following storage set possibilities exist:

If the storage set uses only cache storage, for example the **Cache Only – Library Server** storage set:

- Disk Storage Manager (DSM) expires both indexes and data when the **Life of Data and Indexes** is reached.

If the storage set uses TSM as the storage manager:

- DSM expires only the indexes when the Life **of Data and Indexes** is reached.
- You must ensure that the TSM copy group that is used has a retention equal to or greater than the **Life of Data and Indexes** in the application group.

If the storage set uses ASM as the storage manager (which is the most common type of storage set for CMOD for i customers):

- DSM expires only the indexes when the **Life of Data and Indexes** is reached.
- You must ensure that the sum of the duration of your migration policy levels is equal to or greater than the **Life of Data and Indexes** specified in the application group.

- You must have an expire level in your migration policy.
- You must run ASM on a regular basis to expire the data.

Note that when using ASM, a storage set also references a migration policy created in System i Navigator.

To check if you have any application groups with an **Expiration Type** of **Document** or **Segment**, run the following SQL statement. Change QUSROND to the name of your instance.

```
SELECT SUBSTR(ARSAG.NAME,1,20) AS APPLICATION_GROUP,
SUBSTR(ARSSET.NAME,1,15) AS STORAGE_SET,
CASE EXPIR00001 WHEN X'53' THEN 'SEGMENT'
WHEN X'41' THEN 'DOCUMENT' END AS EXPIRATION_TYPE
FROM QUSROND/ARSAG, QUSROND/ARSSET
WHERE EXPIR00001 <> x'4C' AND ARSAG.SID=ARSSET.SID
ORDER BY ARSSET.NAME, ARSAG.NAME
```

The results should be similar to the following. You can use these results to determine what actions, if any, you need to take to configure your CMOD system properly.

```
APPLICATION_GROUP     STORAGE_SET  EXPIRATION_TYPE
CHECKSTMTS            DISK                DOCUMENT
LABOR                 DISK                SEGMENT
UserDefinedData       DISK                DOCUMENT
System Log            SYSTEMLOG           SEGMENT
```

10.3.2.2 ARSEXOAM/ARSEXPIR (z/OS only)

When expiring documents from OAM using the ARSEXOAM/ARSEXPIR program, it is possible to improve the unload/expiration performance by using the ARS_EXPIRE_REQLIMIT parameter. This parameter controls the number of load IDs sent at a time to the server in a single expiration request. The default value is 20, meaning a separate request is sent for each 20 load IDs being processed.

Load IDs for the same application group can be grouped together up to the ARS_EXPIRE_REQLIMIT value. All load IDs in a single expiration request must belong to the same application group. For example, adding ARS_EXPIRE_REQLIMIT=100 (in the ars.cfg file for the instance) will allow up to 100 load IDs for an application group to be processed at a time.

The optimum value to use is a function of multiple variables including table size. Suboptimal values might lead to table scans. EXPLAINs with various SQL using the type of SQL involved will help in determining whether an index or a table scan will occur.

Notes related to ARSEXOAM:

1. If one object for a given load ID is deleted, all of the index entries for that load ID are deleted.
2. Index entries of all OAM objects that are recorded as being deleted by rows in the ARSOAM_DELETE table are deleted regardless of the settings in the Life of Data and Indexes section on the Storage Management tab of the application group.
3. If you plan to use Storage Management expiration, ensure that you set the expiration type of all application groups to Storage Manager.
4. The recommendation expiration type for CMOD is **Load.** CMOD will support expiration type of **Load** with the use of ARSEXOAM for expiring the indexes in CMOD.

Notes related to ARSEXPIR:

1. If one object for a given load ID is deleted, all of the index entries for that load ID are deleted.

Index entries of all objects that are recorded as being deleted by the SMF records are deleted regardless of the settings in the **Life of Data and Indexes** section on the Storage Management tab of the application group. If you want to use Storage Management expiration, ensure that you set the expiration types of all application groups to **Storage Manager**.

11 Security

11.1 Security overview

CMOD employs several different security techniques. Discussed in this chapter are:

- **SSL:** which allows for secure communication between the server and the clients
- **Stash file:** which allow for secure storage of passwords on the CMOD server
- **LDAP:** which allows for storage of “authentication” values on a remote/centralized server

11.2 Transport Layer Security (TLS) and Secure Sockets Layer (SSL)

CMOD version 8.5 and higher supports Secure Sockets Layer (SSL) and its successor, Transport Layer Security (TLS), for all transmissions between the CMOD servers and clients. SSL is the standard technology for creating secure connections between servers and clients. The secure connection allows for authentication and verification, and data encryption. Authentication and verification allow both the client and server to verify that they are communicating with the intended receiver. Data encryption prevents the packets of information travelling through the network to be viewed by any persons who have access to the network. The CMOD server can be configured to listen on either a non-SSL port or an SSL port, or it can listen on both types of ports. Note that when this topic mentions SSL, the same information applies to TLS, unless otherwise noted.

Once a CMOD client (for example, the OnDemand Windows client, ARSDOC, or ODWEK Java API) has been configured to logon to a CMOD library server with SSL, all communication to and from that client will be done using SSL.

- Between the client and the library server
- Between the client and the object server(s)
- Between the object server(s) and the library server

Important considerations

Separate port number

A separate port number is identified on the CMOD server to support the secure connection. When a client connects to this port it negotiates a connection through a handshake procedure during which the client and server agree on the session parameters to be used in order to maintain a secure connections. Session keys are generated that allow for the encryption and decryption of the data sent between the client and server.

CPU consumption

SSL improves security by encrypting and decrypting data across the network. The encryption and decryption are done at the application layer. This consumes additional CPU cycles on both the sending and receives systems. Therefore you should consider using SSL only for sessions where it is needed. Consider adding additional processor resources on the CMOD server and/or clients to manage the increased overhead.

Digital certificates

With SSL, the identities of the parties are verified through the use of digital certificates. Digital certificates have expiration dates, once a digital certificate has expired, CMOD will not be able to establish connections thru SSL. Therefore, always be aware and plan ahead to avoid expired certificates.

ODWEK

The support of SSL and ODWEK refers specifically to the transfer of data between ODWEK and the CMOD server(s) and does not imply a level of support from the browser to ODWEK. Using SSL from the browser to ODWEK has always been allowed and does not require any support from ODWEK. It is the application/web developer’s responsibility to enable such support.

11.3 Stash files

Stash files and the ARSSTASH command provide a way to securely store and pass a password to CMOD commands that require passwords in a manner that does not require the passwords to be in cleartext (unencrypted). The ARSSTASH command is used to encrypt the password by using AES-128 encryption and store it in a file called a stash file. The path to that stash file is then specified on the -p parameter to those commands that require a password. The stash file is retrieved and decrypted, and the password stored in the stash file is used. This allows the -p parameter stored in JCL or other scripts or programs to not contain a cleartext password.

Multiplatforms - Stash files are the method of choice for securely storing passwords on a CMOD for Multiplatforms server. Unified login does not work when using a CMOD for Multiplatforms server, and thus stash files are the only mechanism provided to not have passwords specified in the clear for the various CMOD commands that require passwords.

z/OS - If you are using a z/OS CMOD server, then the recommendation is to use the z/OS unified login mechanism instead of the stash file. Unified login provides the same functionality as stash files, meaning that the passwords are not stored unencrypted when at rest (e.g. in canned JCL or scripts) without introducing the additional burden of having to manage stash files. For example, when a password is changed for a user, stash files containing the encrypted password for that user must also be changed.

IBM i - If you are using an IBM i server, then you do not need to use stash files. This is because if you are signed on to the IBM i server with a user profile that is defined to CMOD and has enough authority to perform the function you are running, then CMOD will use the IBM i user profile for that function (such as ARSDOC or ARSLOAD). If this is the case, the –u and –p parameters are not required, thus preventing you from having to show or store a password in cleartext. However, if you prefer to use stash files, support for stash files was added to the ARSxxx APIs on IBM i in CMOD server version 9 and to the Add Report (ADDRPTOND), Start Monitor (STRMONOND), and Merge Spooled Files (MRGSPLFOND) commands at CMOD version 7.2.

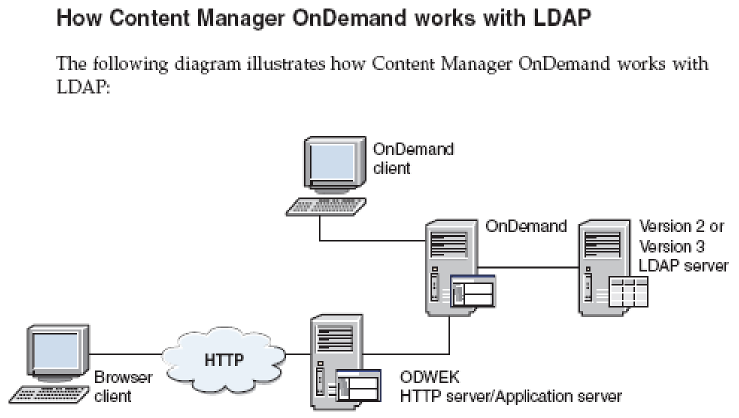
11.4 LDAP

The Lightweight Directory Access Protocol (LDAP) is an open industry standard that has evolved to share information between distributed applications on the same network, organize information in a clear and consistent manner, and prevent unauthorized modification or disclosure of private information. In recent years, LDAP has gained wide acceptance as the directory access method of the Internet, and becomes strategic within corporate intranets.

You can use LDAP to manage basic login authentication directly on the server, in other words, you no longer need to use the user security exit.

The following diagram illustrates how CMOD works with LDAP:

Content Manager OnDemand with LDAP


How Content Manager OnDemand works with LDAP

The following diagram illustrates how Content Manager OnDemand works with LDAP:

OnDemand client
OnDemand
Version 2 or Version 3 LDAP server
Browser client
HTTP
ODWEK HTTP server/Application server

Figure 37. CMOD with LDAP

If you are using LDAP, you need to take several items into consideration.

LDAP server: LDAP is running on another system and it is connected to CMOD via TCP/IP. There will thus be a time delay between when the verification request is issued by CMOD and the result is returned to CMOD. The length of this time depends on the TCP/IP network connection, the response time of the LDAP server, and the current LDAP workload.

Users with admin level security bypass LDAP. So you can compare an admin user’s response time to a production user’s response time to determine the LDAP overhead.

If the LDAP server or the connection to the LDAP server fails then users will not be able to logon to CMOD except for users with admin level security.

12 Exit programs

12.1 Overview

All exit programs and logs that are not required should be disabled or not enabled.

On Multiplatforms and IBM i, you can disable an exit program by renaming or moving the program so that CMOD cannot locate it. If CMOD detects the existence of an exit program, the exit program will be called.

On z/OS, see the relevant publication for instructions on how to enable a particular exit program, and enable it only if it is required.

12.2 All platforms exit program

12.2.1 ARSLOG – System Log exit

Both system logging and user exit logging should be turned off unless they are needed. This is accomplished by updating the System Parameters of your CMOD instance server(s) using the OnDemand Administrator client.

In addition, when you specify ARS_DISABLE_ARSLOG=1 in the ars.cfg file, the System Log Exit (ARSLOG) will not be invoked at all. Without specifying ARS_DISABLE_ARSLOG=1, even if all user exit logging is disabled in the System Parameters of the OnDemand Administrator client, an attempt will still be made to call the ARSLOG exit for certain messages. If you are not planning to use the ARSLOG exit, you should specify ARS_DISABLE_ARSLOG=1 to minimize the overhead of attempting to call the ARSLOG exit. Note that, on IBM i, all new instances are created with ARS_DISABLE_ARSLOG=1 set as the default.

12.3 Multiplatforms and z/OS exit programs

12.3.1 ARSULOAD – Load exit

-Z userExit

For the LOAD and LOAD_DB functions, specifies a user-defined string that is passed to the load user exit program. When performing migration loads, you must specify this parameter to ensure the processing of the migration load is performed.

Note: The ARSULOAD exit applies to CMOD for Multiplatforms and z/OS only.

12.3.2 ARSUPREP – Client Preview exit

The CMOD client preview exit allows you to process document data before the document is presented to the client. The following figure shows an overview of the client preview exit.

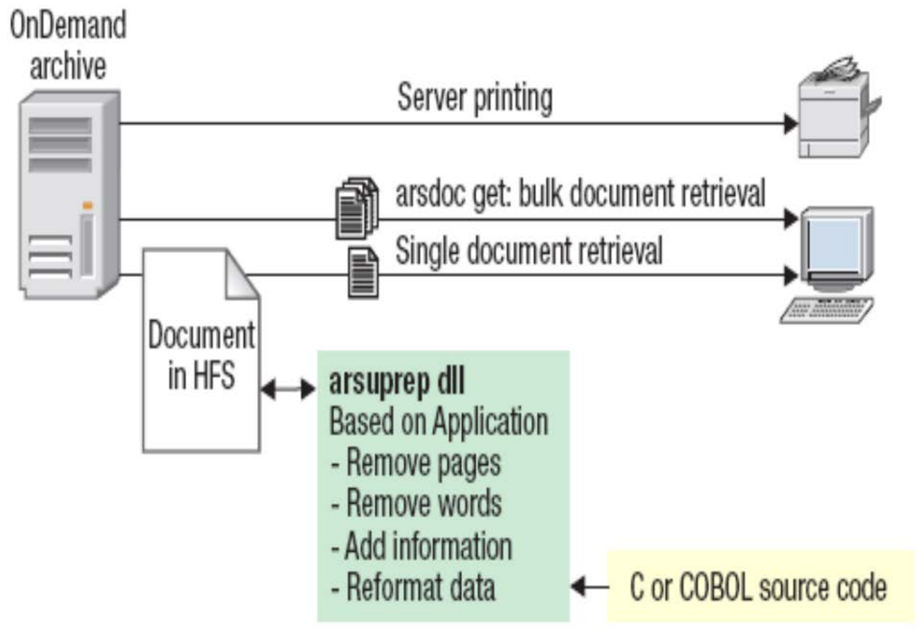


Figure 38. Client Preview exit

The client preview exit can be used to add, remove, or reformat data before the document is presented to the client. For example:

- Remove pages from the document, such as banner pages, title pages, all pages but the summary page, and so on.
- Remove specific words, columns of data, or other information from the document. That is, omit ("white out") sensitive information such as salaries, social security numbers, and birth dates.
- Add information to the document, for example, a summary page, data analysis information, and Confidential or Copy statements.
- Reformat data contained in the document, for example, reorder the columns of data.

Notes:

1. The client preview exit is not called for all document retrievals. It is not called for functions that use the Bulk Retrieval method of retrieving documents, or for server printing. Running the ARSDOC GET function without specifying the -n parameter performs a bulk retrieval, and documents retrieved will not be presented to the client preview exit.
2. The exit is not called for server printing.
3. If a request is made to retrieve a large object document, care should be taken to make certain that the client preview exit does not remove any pages from the document. The large object segment size and page navigation information are based on the number of pages that existed when the document was loaded on the server. Unexpected results might occur if this information is changed.
4. The client preview exit is enabled at the application level. The exit is invoked only for the specified applications.

Note: The ARSUPREP exit applies to CMOD for Multiplatforms and z/OS only.

12.3.3 ARSTBLSP – Tablespace creation

The ARSTBLSP program can be used to migrate existing tables of application group data from the default tablespace to separate tablespaces. IBM recommends that all customers who are upgrading from earlier versions of CMOD use this program to migrate their existing application group data to tablespaces.

During normal operation, CMOD loads index rows into a table until the **Maximum Rows** value for the application group has been reached. Such a table is said to be open for loading. When the **Maximum Rows** value is reached, the table is closed and a new table and tablespace are created. Under certain circumstances, you might desire to close a table to loading before the **Maximum Rows** value is reached. For example, migration processing (by using the ARSMAINT -e function) will not process a table that is open for loading, and therefore you might desire to migrate the table earlier than initially anticipated.

The ARSTBLSP program is typically used to copy application group tables from the USER or USERSPACE tablespace to individual tablespaces. Application group tables are stored in the USER or USERSPACE tablespace when the application group is using the dbTablespaceType="None" attribute. When the application group is using the dbTablespaceType="SMS" attribute, the application group tables are stored in their own tablespaces.

Parameters

-a action

The action to perform. The action can be one of the following values:

0 - Migrate one or more tables of application group data to separate tablespaces. Specify the application group to migrate with the –g parameter. Optionally specify a table to migrate with the –t parameter. A table must be closed before you can migrate it to a tablespace.

1 - Close a table that is still open for loading. This action causes CMOD to close the table that is currently open for loading in the specified application group. The next time that data is loaded into the application group, the data is loaded into a tablespace. Optionally specify a table to close with the –t parameter.

2 - List the tables of application group data that have not been migrated to separate tablespaces.

3 - List the tables of application group data that are open for loading. An open table must be closed before you can migrate it to a tablespace.

4 - Create a new table if there are no existing open tables. When you use this value, you must specify the -g parameter.

The following is an overview of the copy tables process:

1. Application group currently using the dbTablespaceType="None" attribute
2. Index data currently loaded into the application group is being stored in the USER or USERSPACE tablespace
3. List the table(s) that are in the USER or USERSPACE ... arstblsp -a2
4. List the table that is still open for loading ... arstblsp -a3
5. Close the table that is still open for loading ... arstblsp -a1
6. Update the application group to change the dbTablespaceType attribute from "None" to "SMS"
7. Verify the ARS_DB_IMPORT parameter in the ars.cfg file (in Windows, a string value in the CFG key HKLM\Software\IBM\CMOD for Windows\@SRV@_ARCHIVE\CFG)
   - If using the ARS_DB_IMPORT=0 (or unspecified ARS_DB_IMPORT) with the -d parameter, verify the -d directory space, permissions, etc.
   - If using the ARS_DB_IMPORT=1 parameter, verify the DB2/TSM configuration
   - If using the ARS_DB_IMPORT=2 parameter, verify the ARS_TMP directory space, permissions, etc.
8. If you made any changes to the ars.cfg file, restart the instance
9. Copy the table(s) into their own tablespace ... arstblsp -a0
10. Verify the new tables: Close, Import/Load into new table (DB2) or "Move table to tablespace" (Oracle)
11. Verify messages (System Log; ARCHIVE.LOG; application Event Log)
12. Verify temporary directory
13. Verify tablespace filesystem now contains table files

14. Verify tables (and indexes, Oracle only) in database
15. When you load new data into the application group, verify that the new index data is stored in its own tablespace(s)

Database considerations

- Oracle does not read the ars.cfg file; Oracle always uses Export/Import (0 or unspecified)
- SQL Server does not support the ARSTBLSP command (will fail)

Operating system considerations

- zLinux does not support the ARS_DB_IMPORT=2 option (will fail)

Filesystem considerations

- Huge table(s)
- Filesystem size, limits

Here is the relevant part of the ars.cfg file:

```
...
...
 #
 # Used for ARSTBLSP command and reloading migrated tables (Library Server Only)
 #   0 (import)
 #   1 (load w/TSM - DB2 only)
 #   2 (load w/DISK - DB2 only, using ARS_TMP)
 #
 #ARS_DB_IMPORT=0
 ARS_DB_IMPORT=1
```

Note: The ARSTBLSP exit applies to CMOD for Multiplatforms and z/OS only.

12.3.4 ARSUSEC User Security exit

The CMOD security system interface exit allows you the ability to authenticate users and perform resource authorization security processing that is provided with the CMOD system. On z/OS, the ARSUSEC exit invokes the ARSUSECZ security exit module.

The following activities are controlled by the exits:

User authentication (checking user security)

- Logon
- Change password
- Add user ID
- Delete user ID

Resource authorization (checking user permissions)

- Access to a CMOD folder
- Access to a CMOD application group
- Restrict access to specific documents
- Control the SQL search criteria used for searching folders

When enabling the exits to implement the required level or type of security, the user ID must be defined for both TSO and CMOD.

The diagram below presents an overview of the security system interface exits:

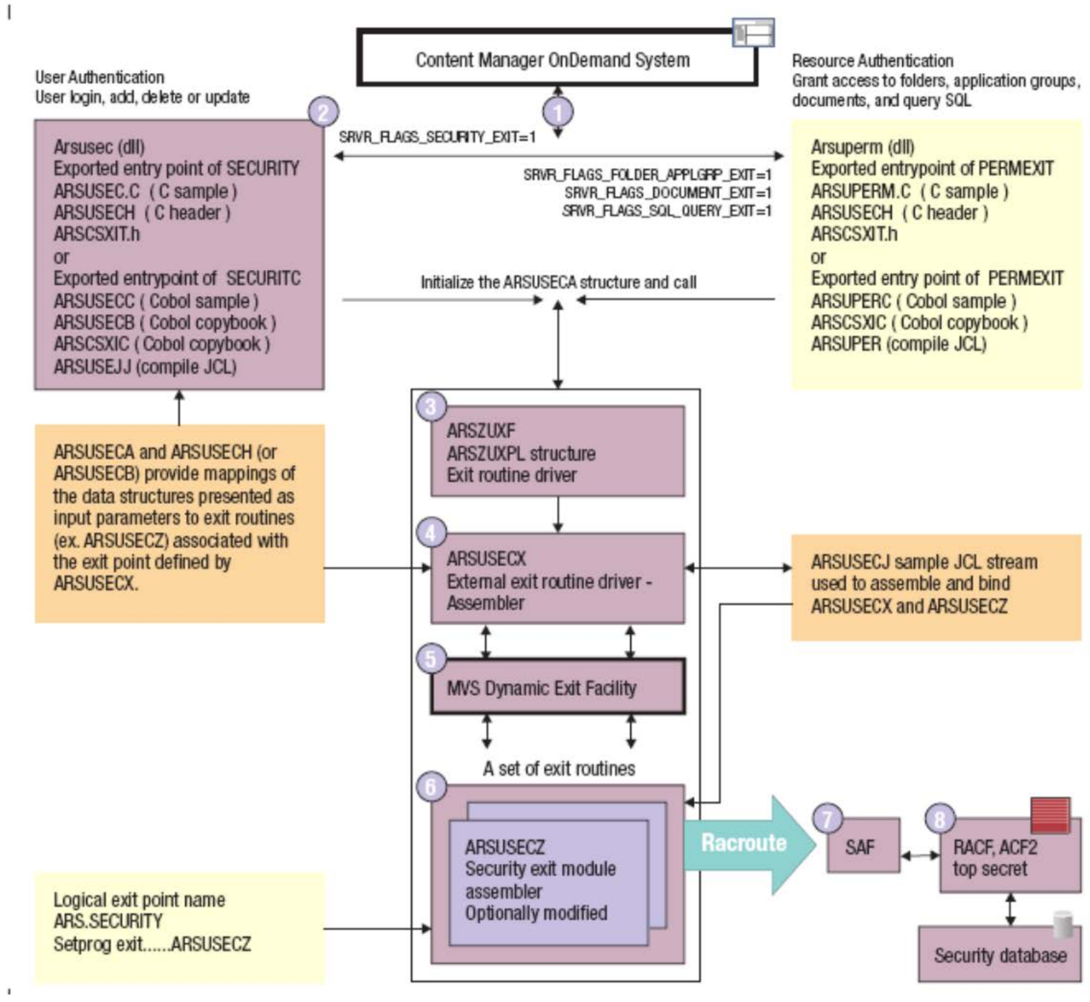

Content Manager OnDemand System
User Authentication
User login, add, delete or update
Resource Authentication
Grant access to folders, application groups, documents, and query SQL
1
2
SRVR_FLAGS_SECURITY_EXIT=1
SRVR_FLAGS_FOLDER_APPLGRP_EXIT=1
SRVR_FLAGS_DOCUMENT_EXIT=1
SRVR_FLAGS_SQL_QUERY_EXIT=1
Arsusec (dll)
Exported entry point of SECURITY
ARSUSEC.C (C sample)
ARSUSECH (C header)
ARSCSXIT.h
or
Exported entrypoint of SECURITC
ARSUSECC (Cobol sample)
ARSUSECB (Cobol copybook)
ARSCSXIC (Cobol copybook)
ARSUSEJJ (compile JCL)
Arsuperm (dll)
Exported entrypoint of PERMEXIT
ARSUPERM.C (C sample)
ARSUSECH (C header)
ARSCSXIT.h
or
Exported entry point of PERMEXIT
ARSUPERC (Cobol sample)
ARSCSXIC (Cobol copybook)
ARSUPER (compile JCL)
Initialize the ARSUSECA structure and call
3
ARSZUXF
ARSZUXPL structure
Exit routine driver
ARSUSECA and ARSUSECH (or ARSUSECB) provide mappings of the data structures presented as input parameters to exit routines (ex. ARSUSECZ) associated with the exit point defined by ARSUSECX.
4
ARSUSECX
External exit routine driver - Assembler
ARSUSECJ sample JCL stream used to assemble and bind ARSUSECX and ARSUSECZ
5
MVS Dynamic Exit Facility
A set of exit routines
6
ARSUSECZ
Security exit module assembler
Optionally modified
Racroute
7
SAF
8
RACF, ACF2 top secret
Security database
Logical exit point name
ARS.SECURITY
Setprog exit......ARSUSECZ

Figure 39. Security exits

Note: The ARSUSEC exit applies to CMOD for Multiplatforms and z/OS only.

12.3.5 ARSUUPDT – Report Specifications Archive Definition exit

The report specifications archive definition exit provides the ability to alter some of the parameters used by CMOD when document data is being captured (loaded) by the ARSLOAD program. It is a two-call procedure. On z/OS the ARSUUPDT exit invokes the ARSUUPDZ module. The following figure depicts this process:

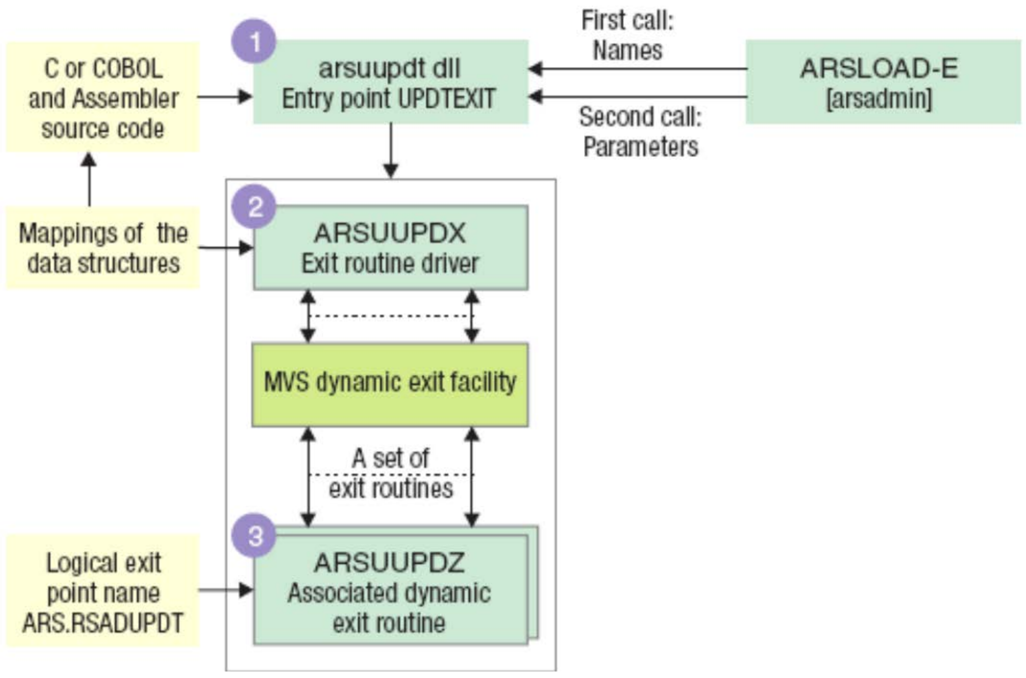


Figure 40. Report Specifications Archive Definition exit

The first call modifies the Names Process:

- Application group
- Application
- Object server
- Storage node
- DB field date format
- DB field name

The second call modifies the Parameters:

- Indexer parameters
- Input file parameters

Note: The ARSUUPDT exit applies to CMOD for Multiplatforms and z/OS only.

12.4 z/OS exit programs

12.4.1 APKACIF – ARSSPVIN

There are special considerations for APKACIF exits written in COBOL.

The ARSSPVIN sample APKACIF input exit is written as a COBOL main program. In order to prevent the Language Environment from creating and destroying the COBOL run-time environment each time ARSSPVIN is called, a CEEUOPT CSECT must be assembled and link-edited with the COBOL object code. Information about how to construct a CEEUOPT CSECT is documented in the z/OS V1R10.0 Language Environment Customization, SA22-7564-10. A sample CEEUOPT CSECT is in data set CEE.SCEESAMP(CEEUOPT). You can use this sample as a model, but you must be sure that the following option is specified: RTEREUS= (ON).

IBM also recommends that you specify the ALL31(ON) option, but this option is not required. Stack and heap storage can potentially be allocated above the line while running with ALL31(OFF) specified.

In addition, you must be sure that the resulting module is link-edited as NOT RE-ENTRANT and NOT REUSABLE. These link-editing conditions are required to allow the local variables within the COBOL exit code to retain their values across multiple invocations.

Note: The APKACIF exit applies to CMOD for z/OS only.

12.4.2 ARSPTGN – Unified Logon exit

The CMOD unified login exit (ARS.PTGN) enables you to execute the CMOD command line utilities (such as ARSLOAD) without requiring a userid and password to be specified. This facility to logon without specifying a password utilizes the ability to specify a PassTicket as a password when using a RACROUTE REQUEST=VERIFY call.

Note: The ARSPTGN exit applies to CMOD for z/OS only.

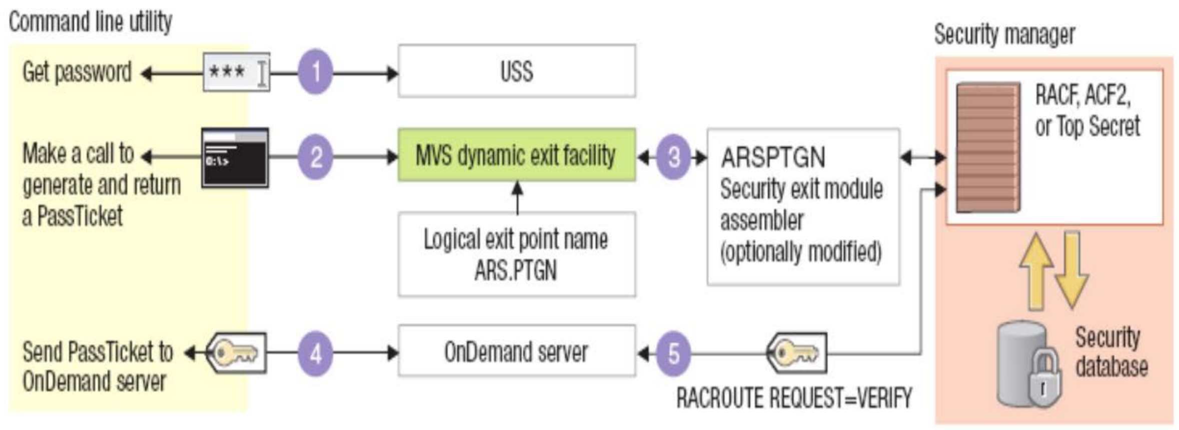

Command line utility
Get password
***
1
USS
Make a call to generate and return a PassTicket
C:\>
2
MVS dynamic exit facility
3
ARSPTGN Security exit module assembler (optionally modified)
Logical exit point name ARS.PTGN
Send PassTicket to OnDemand server
4
OnDemand server
5
RACROUTE REQUEST=VERIFY
Security manager
RACF, ACF2, or Top Secret
Security database

Figure 41. Unified Logon exit

To enable PassTickets in a security manager such as RACF, you must perform the following steps:

1. Activate the PKTDATA class.
2. Define a secured sign-on application key for each application.
3. Issue the SETROPTS RACLIST(PTKTDATA) command.

Note: The ARSPTGN exit applies to CMOD for z/OS only.

13 Potential performance impacts

13.1 Accumulation of ARSSOCKD processes

If logons, open folder, searches, and loads or document retrievals seem slow, occurring intermittently or during periods of high volume, the number of ARSSOCKD processes or threads might be considerably higher compared to periods of normal performance. Restarting the CMOD server instance might temporarily alleviate the problem. However, it is important to determine what is causing ARSSOCKD processes or threads to accumulate.

Cause

ARSSOCKD is the name of the CMOD server process for z/OS and Multiplatforms servers, and for IBM i servers prior to V7.1; QRLMCTL1 is the name of the server process for IBM i servers running V7.1 or higher. (For purposes of this discussion, we will use ARSSOCKD for simplicity.) Depending on the platform and version of the CMOD server, ARSSOCKD may exist as a single parent process that forks child processes or as a single multi-threaded process. On z/OS and Multiplatforms servers, this can be seen by issuing a process listing,

ps -ef | grep arssockd **OR** *ps -efL | grep arssockd*

for a multi-threaded ARSSOCKD process. On IBM i, run the Work with Active Jobs (WRKACTJOB) command, and then use option 12 beside your CMOD instance server job to Work with threads to see similar information. For simplicity, the information below assumes a CMOD server version that has a single parent process and multiple child ARSSOCKD processes, though the material is relevant for both implementations.

The ARSSOCKD processes consist of an accepting parent process, a license server process, and database server processes. If ARS_NUM_DBSRVR is configured as non-zero in the ars.cfg file, the ARSSOCKD processes that accumulate are known as client ARSSOCKD processes.

The purpose of a client ARSSOCKD process is to manage a request between the user and CMOD server. This might involve facilitating a database request (logon, open folder, search, load, System Log entry, and so on), a document retrieval request to the object server (CMOD cache or Tivoli® Storage Manager), or managing a cancel request issued by the user. Each user request will fork an ARSSOCKD process, while document retrieval will fork an additional ARSSOCKD. A client ARSSOCKD process will exist until the request has been satisfied or cancelled.

There are several reasons why poor performance and subsequently an accumulation of ARSSOCKD processes might occur:

- Excessive System Log messages
- Poor performing database
- Database servers are servicing a long-running request or a server-based text search
- Poor object server performance
- Poor network performance
- User volume has exceeded system capacity or configuration

Resolving the problem

For each reason listed in the previous section, the respective action should be taken in the following section.

Excessive System Log messages

If excessive logon, logoff, or application group message logging (search, document retrieval, and so on) is enabled, CMOD performance will be affected. In CMOD server versions prior to 9.0.0.2, each System Log message is an insert into the CMOD database and each transaction that has message logging enabled will wait for the System Log message to be inserted before the transaction is completed. A bottleneck will occur in this situation if a high volume of transactions are being logged. This can affect overall performance and cause an accumulation of ARSSOCKD processes in high volume environments. In addition, System Log messages might be written to the system console and the System Log user exit. (See ARSLOG – System Log exit for more information on the System Log user exit.) Problems writing to the console, for example, will have an effect on performance. Unneeded System Log message logging should be disabled.

Fortunately, regardless of the level of message logging you have defined, messages that indicate an actual error condition will always be logged. After you have disabled any excessive System Log message logging, you should review your System Log to ensure that an abundance of error messages does not exist. Lastly, ensure that the System Log application group does not have any message logging enabled.

Note that, starting in server version 9.0.0.2, a thread now collects messages and inserts them into the System Log every four seconds, by default. This change dramatically improves performance when writing messages in bulk to the System Log.

Poor performing database

With the initialization of CMOD, the server forks a number of ARSSOCKD database server processes that exist until the CMOD server is stopped. This number is determined by the ARS_NUM_DBSRVR parameter value in the ars.cfg file. These database server processes maintain a persistent pool of connections to the database. When a database request is made (logon, open folder, search, loading, System Log messages, and so on), a client ARSSOCKD process is forked for the life of this request, and will wait for a free database server process to service its database request. After the request is fulfilled, the database server process will service the next request or return to idle, while the client ARSSOCKD process that was forked will be destroyed. When an accumulation of ARSSOCKD processes occurs, this is due to the client ARSSOCKD processes being queued, and subsequently waiting for a free database server process to fulfill its request. These database server processes are busy waiting for the database to fulfill its request. Investigation into the database manager's performance will need to be performed.

Database servers are servicing a long-running request or a server-based text search

A search request or server-based full text search that yields a large result set might tie up an ARSSOCKD database server process for a long period of time. If a high number of these long searches are performed, CMOD database performance will be affected. For example, if 20 ARSSOCKD database server processes are started on initialization *(ARS_NUM_DBSRVR=20)* and 20 long searches are issued simultaneously, all subsequent requests (logon, open folder, search, System Log, and so on) will be queued until at least one of the 20 searches has finished or is cancelled. On z/OS or any of the Multiplatforms systems, reviewing the process listing generated by running *ps -ef | grep ars* can help determine if database server processes are not responding. On IBM i, reviewing the details of any QSQSRVR jobs listed after running the Work with Active Jobs (WRKACTJOB) command provides similar information. All database server processes should have relatively the same CPU TIME. Search queries can be examined by enabling database query message logging on a per application group basis.

Poor object server performance

Should the poor performance be isolated to document retrieval operations only (logon, search, open folder, and loading are normal), then an accumulation of ARSSOCKD processes or ARSOBJD processes might indicate an issue with the object server. A determination of whether document retrieval performance is poor when retrieving from CMOD cache or from Tivoli® Storage Manager will need to be made. The amount of time that the CMOD server takes to retrieve a document and AFP resource is recorded in the System Log if document retrieval message logging is enabled in the application group.

If retrieval performance is poor from only Tivoli Storage Manager, investigation into the Tivoli Storage Manager Client API and Tivoli Storage Manager server will need to be made. If retrieval performance is poor from the CMOD cache, see the following "Poor network performance" section.

Poor network performance

Poor network performance can be between the client, CMOD library server, or CMOD object server(s). Typically if network performance is an issue, performance can be said to have been acceptable at one time, but suddenly to be noticeably poor with no change to CMOD usage or volume. Therefore, investigation into a possible network problem needs to be performed.

User volume has exceeded system capacity or configuration

If the number of users has increased, performance is consistently poor, and investigation into the other potential problem areas described above has been performed, then an evaluation of the system sizing and configuration needs to be performed. The volume of users or workload may have exceeded the system capacity.

13.2 Configuring CMOD database subservers

Another explanation for slow performance might be due to an abnormally high number of threads, which is affected by the setting of the ARS_NUM_DBSRVR parameter.

Cause

The ARS_NUM_DBSRVR parameter specifies the size of the connection pool, which all database requests in CMOD must use. On z/OS, IBM i, and Multiplatforms servers other than Windows®, this parameter is specified in the ars.cfg file. On Windows, this is specified in the OnDemand Configurator.

Upon initialization, CMOD creates and maintains a number of connections to the database based on this parameter. If the connections in the pool are all currently busy, subsequent requests will be queued and seem hung, waiting for a free connection. This means setting a value too low will cause database requests to queue up and bottleneck (an accumulation of ARSSOCKD processes or threads will then occur), while setting a value too high will consume unnecessary memory, CPU, and database resources. The default value is 4, but for most production environments this number is insufficient.

The following is a list of common operations that will require a database subserver connection:

- User operations. For example, logon, open folder, folder search, and server based text search.
- Historical logging, such as the System Log and application group logging. For example, if document retrieval logging is enabled, each document retrieval request will require a database subserver connection to log the retrieval in the System Log.
- CMOD commands and daemons, such as ARSDOC, ARSLOAD, and ARSMAINT.
- OnDemand Administrator client and ARSXML operations.

Note: For a CMOD system that is using DB2, ensure that the database configuration parameter MAXAPPLS is configured to at least the ARS_NUM_DBSRVR value or AUTO.

Important: It is not recommended to set ARS_NUM_DBSRVR to 0 for a production environment. Setting 0 bypasses the connection pool scheme. CMOD must therefore create and clean up a database connection for each database operation, which is not optimal for performance and allows a limitless number of concurrent connections to be made against your database.

Resolving the problem

The optimal value will depend on each environment and will vary greatly based on the volume of concurrent users, their activities, CMOD commands and daemons running, and the overall performance of your system (network, database, object server, hardware, and so on).

One method to tune this configuration is to set it to a starting value. Then, simulate your expected user volume and monitor the count of ARSSOCKD processes/threads at regular intervals over a period of time. A continual increase in the number of ARSSOCKD processes/threads will indicate that requests are being queued and there is a bottleneck, while a steady number will mean either the pool is adequate in size or too large.

If the number continually increases no matter the ARS_NUM_DBSRVR value that is set, this indicates that your system may not be sized properly or requires performance tuning and optimization.

An example korn script to monitor the number of ARSSOCKD threads every 10 seconds:

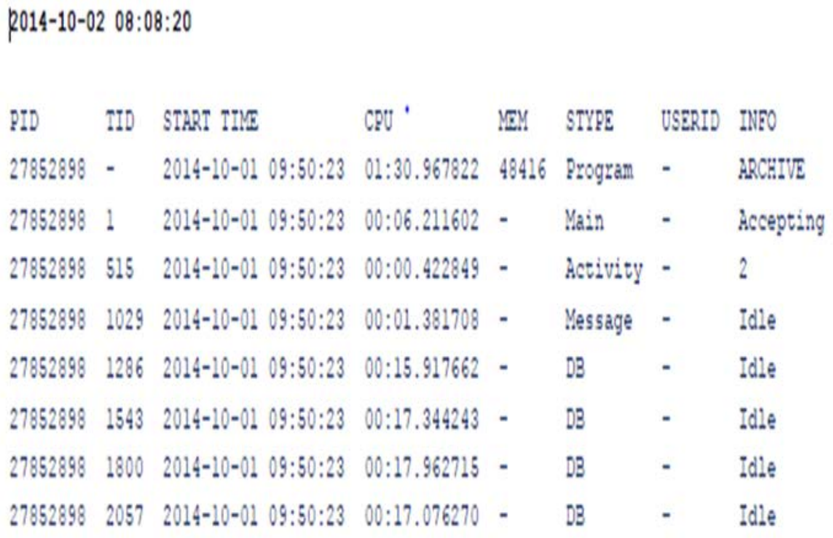
```
#!/usr/bin/ksh

while(true)
do
        echo `date` arssockd count: `arssockd -I ARCHIVE -p | wc -l`
sleep 10
done
```

Depending on the version and platform of your CMOD server, ARSSOCKD may be implemented as a single process with multiple threads per instance or multiple processes with a single thread each per instance. The following is a combination of process listing commands that will provide a process or thread count of ARSSOCKD:

```
arssockd -I ARCHIVE -p | wc -l
ps -ef | grep arssockd | wc -l
ps -efL | grep arssockd | wc -l
```

In CMOD version 9.5, the ARSSOCKD command with the –p option to display threads now includes timestamp:

```
/opt/IBM/ondemand/V9.5/bin> arssockd –I ARCHIVE –p
```



```
2014-10-02 08:08:20

PID       TID   START TIME           CPU           MEM    STYPE     USERID  INFO
27852898  -     2014-10-01 09:50:23  01:30.967822  48416  Program   -       ARCHIVE
27852898  1     2014-10-01 09:50:23  00:06.211602  -      Main      -       Accepting
27852898  515   2014-10-01 09:50:23  00:00.422849  -      Activity  -       2
27852898  1029  2014-10-01 09:50:23  00:01.381708  -      Message   -       Idle
27852898  1286  2014-10-01 09:50:23  00:15.917662  -      DB        -       Idle
27852898  1543  2014-10-01 09:50:23  00:17.344243  -      DB        -       Idle
27852898  1800  2014-10-01 09:50:23  00:17.962715  -      DB        -       Idle
27852898  2057  2014-10-01 09:50:23  00:17.076270  -      DB        -       Idle
```

Figure 42. Display of threads with timestamp

13.3 Configuring THREADSTACK LE options in CMOD

It is important to perform Language Environment (LE) tuning in CMOD. Setting Report Storage on (RPTSTG(ON)) will generate a report of the storage used while CMOD is running. You can use the storage report information to assist in setting the runtime options for the best storage tuning. This should only be used when tuning CMOD, as setting it as a default will cause a performance impact.

As part of the ongoing monitoring of CMOD, if the storage report generated by the RTPSTG(ON) reveals a large number of get segment and free segment calls for the thread stack, this can have a negative performance impact in CMOD.

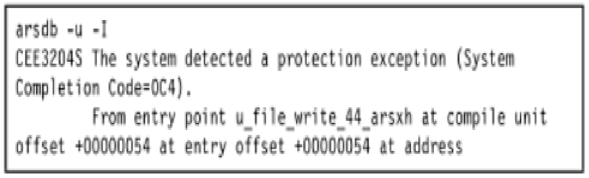
Resolving the problem

Investigation might show that the majority of the thread stack get/free segment calls were driven by Open Database Connectivity (ODBC) LE stack requirements.

A change was made to CMOD version 8.5 and 9.0 to provide a stack size for the individual CMOD DB threads. This eliminates the need to use THREADSTACK setting, which causes any and all threads to use the same defined stack size whether they need it or not, leading to an excessive use of storage.

13.4 Configuring the MAXMMAPAREA in OMVS

When installing CMOD, there may be issues encountered during this installation. One common issue you may encounter is related to the initialization of ICU:



```
arsdb -u -I
CEE3204S The system detected a protection exception (System
Completion Code=0C4).
        From entry point u_file_write_44_arsxh at compile unit
offset +00000054 at entry offset +00000054 at address
```

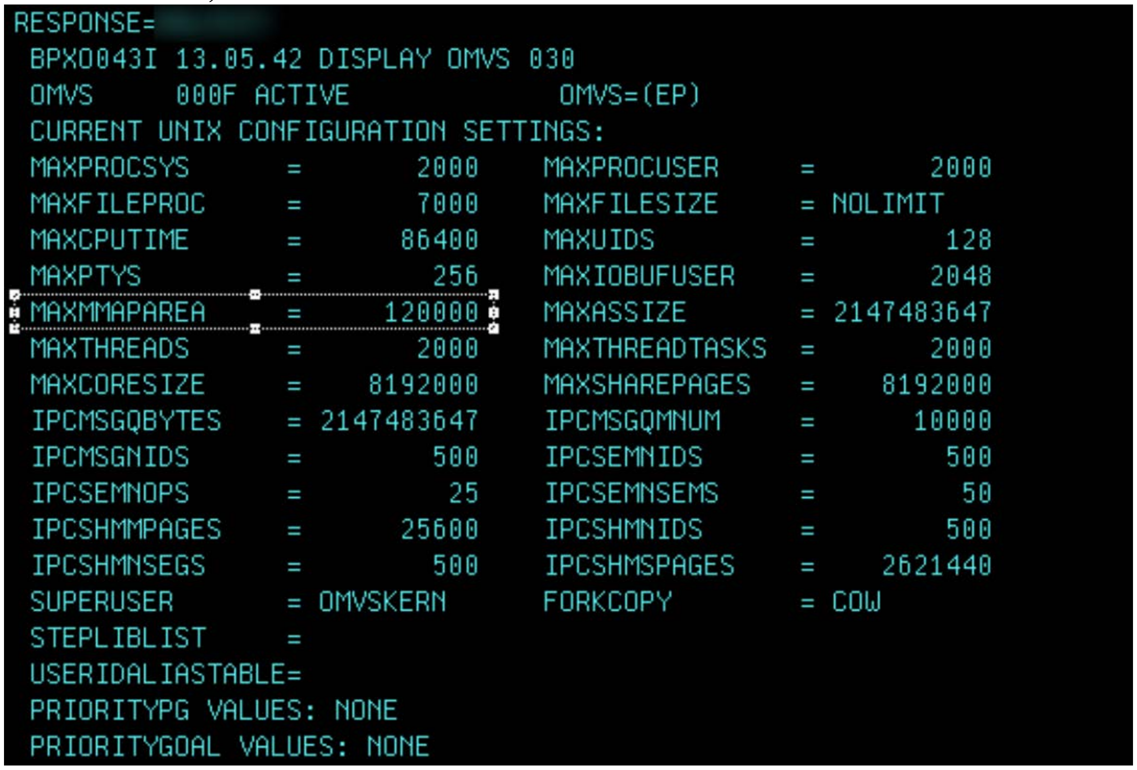
Figure 42. ARSDB error during installation

Resolving the problem

Investigation might show that the MAXMMAPAREA is set too low. This parameter specifies the maximum amount of system wide data space storage that is allocated for memory mapping of files (in 4096 byte pages). Storage is not allocated until memory mapping is active.

To determine this setting, issue:

/d omvs,o



```
RESPONSE=[redacted]
 BPXO043I 13.05.42 DISPLAY OMVS 030
 OMVS     000F ACTIVE             OMVS=(EP)
 CURRENT UNIX CONFIGURATION SETTINGS:
 MAXPROCSYS      =       2000    MAXPROCUSER     =       2000
 MAXFILEPROC     =       7000    MAXFILESIZE     = NOLIMIT
 MAXCPUTIME      =      86400    MAXUIDS         =        128
 MAXPTYS         =        256    MAXIOBUFUSER    =       2048
 MAXMMAPAREA     =     120000    MAXASSIZE       = 2147483647
 MAXTHREADS      =       2000    MAXTHREADTASKS  =       2000
 MAXCORESIZE     =    8192000    MAXSHAREPAGES   =    8192000
 IPCMSGQBYTES    = 2147483647    IPCMSGQMNUM     =      10000
 IPCMSGNIDS      =        500    IPCSEMNIDS      =        500
 IPCSEMNOPS      =         25    IPCSEMNSEMS     =         50
 IPCSHMMPAGES    =      25600    IPCSHMNIDS      =        500
 IPCSHMNSEGS     =        500    IPCSHMSPAGES    =    2621440
 SUPERUSER       = OMVSKERN      FORKCOPY        = COW
 STEPLIBLIST     =
 USERIDALIASTABLE=
 PRIORITYPG VALUES: NONE
 PRIORITYGOAL VALUES: NONE
```

Figure 44. Display of UNIX configuration settings highlighting MAXMMAPAREA

By issuing the following command: */d omvs,l*
you will obtain the Current and Highwater usage of the storage.

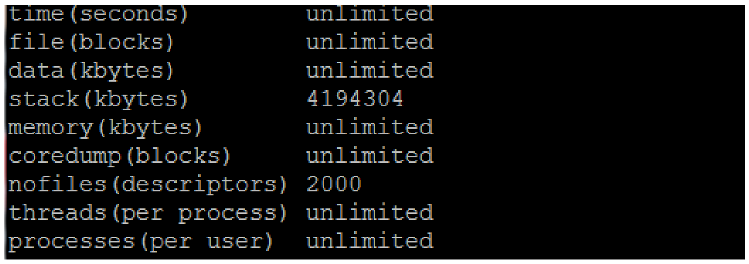
Setting this value to a higher value will prevent these types of errors/failures. Therefore, a recommendation of 120000 pages for this setting is suggested.

To set MAXMMAPAREA to 120000 pages, issue the following command:
/setomvs MAXMMAPAREA=120000

13.5 Appropriate *ulimit* settings for CMOD instance owner

Since the CMOD server is a 64-bit process, the soft and hard limits for the user running the server process should be set to unlimited.

Issue the command *ulimit –a* as the user who starts CMOD. These are the environment’s soft limits that will be inherited by the CMOD server. The output of this command will vary by platform, however it will contain the information below:



```
time(seconds)        unlimited
file(blocks)         unlimited
data(kbytes)         unlimited
stack(kbytes)        4194304
memory(kbytes)       unlimited
coredump(blocks)     unlimited
nofiles(descriptors) 2000
threads(per process) unlimited
processes(per user)  unlimited
```

The values to note in the output of the ulimit command are data, stack, memory and nofiles. The recommendation is to set them to “unlimited” to allow the CMOD server all the physical memory resources on the system. You can do this by updating the limit values for the user that starts the CMOD server. How you do that varies by platform. You can also update the values from the command line by using the ulimit command itself.

Appendix A: Information Centers/Knowledge Centers

Documentation enhancements for version 9.5

The new Knowledge Centers (KCs) replace the Information Centers for product documentation for Multiplatforms and z/OS versions 9.5, 9.0, and 8.5, and for IBM i versions 7.2 and 7.1:

- KC for Multiplatforms: http://www.ibm.com/support/knowledgecenter/SSEPCD/welcome
- KC for z/OS: http://www.ibm.com/support/knowledgecenter/SSQHWE/welcome
- KC for i: http://www.ibm.com/support/knowledgecenter/SSB2EG/welcome

Other version 9.5 information updates:

- New consolidated Indexing Reference for all platforms
- New consolidated OnDemand Distribution Facility (ODF) and Report Distribution Reference for Multiplatforms and z/OS
- Installation procedures updated
- Help text for each client updated
- All new error messages updated to include user actions and included in the Messages and Codes Guide
- Help provided in 26 languages and now includes Russian and Turkish

Other CMOD product documentation

CMOD for i version 7.2 **Publication Library** at http://www.ibm.com using '7041971' as search criteria.
CMOD for i version 7.1 **Publication Library** at http://www.ibm.com using '7016915' as search criteria.

Related information for all platforms

IBM Knowledge Center
http://www.ibm.com/support/knowledgecenter/

IBM Redbooks web page
http://www.redbooks.ibm.com

ICU - International Components for Unicode
http://site.icu-project.org/

IBM Tivoli Storage Manager Version 6.3 Information Center
http://pic.dhe.ibm.com/infocenter/tsminfo/v6r3/index.jsp

TCP/IP protocol
http://pic.dhe.ibm.com/infocenter/soliddb/v7r0/index.jsp?topic=%2Fcom.ibm.swg.im.soliddb.admin.doc%2Fdoc%2Ftcp.ip.html

WebSphere Application Server Version 7.0 Information Center
http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp

WebSphere Application Server Version 8.0 Information Center
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r0/index.jsp

WebSphere Application Server Version 8.5 Information Center
http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp

Multiplatforms

IBM DB2 Database for Linux, UNIX, and Windows Information Center
http://pic.dhe.ibm.com/infocenter/db2luw/v9r7/index.jsp

IBM DB2 Version 10.1 Information Center
http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/index.jsp

AIX Information Center
http://publib16.boulder.ibm.com/pseries/index.htm

Welcome to the AIX 6.1 Information Center
http://pic.dhe.ibm.com/infocenter/aix/v6r1/index.jsp

z/OS

IBM z/OS Complete Center
http://www-03.ibm.com/systems/z/os/zos/

z/OS V1R12.0 information center
http://publib.boulder.ibm.com/infocenter/zos/v1r12/index.jsp

z/OS V1R13.0 Information Center
http://publib.boulder.ibm.com/infocenter/zos/v1r13/index.jsp

IBM System z publications
http://publib.boulder.ibm.com/infocenter/etc/cust/index.jsp?topic=%2Fcom.ibm.storage.etc.doc%2Fetc_pubs_systemz.html

DB2 for z/OS - Technical Resources
http://www-01.ibm.com/support/docview.wss?uid=swg27011656

DFSMSdfp (OAM) - z/OS V1R12.0 Information Roadmap
http://publib.boulder.ibm.com/infocenter/zos/v1r12/index.jsp?topic=%2Fcom.ibm.zos.r12.e0zc100%2Fe0z2c1b018.htm

IBM i

IBM i and System i Knowledge Center
http://www.ibm.com/support/knowledgecenter/ssw_ibm_i/welcome

Appendix B: Additional information

CMOD Newsletters

CMOD newsletters can be found on the web on the CMOD support portal or at http://www.ibm.com using '7024130' as search criteria.

Other useful links

Content Manager OnDemand system requirements
• (MP) http://www.ibm.com/support/docview.wss?uid=swg27043162
• (z/OS) http://www.ibm.com/support/docview.wss?uid=swg27043163
• (IBM i) http://www.ibm.com/support/docview.wss?uid=swg27041914

Compatibility matrix for the Content Manger OnDemand client and servers
• http://www.ibm.com/support/docview.wss?rs=152&uid=swg21392275

Content Manager OnDemand information roadmap
• http://www.ibm.com/support/docview.wss?rs=152&uid=swg27009157

Content Manager OnDemand product overview
• www.ibm.com/software/products/ondemand

OnDemand User Group
• http://www.odusergroup.org

CMOD Redbooks and whitepapers

- Building IBM Enterprise Content Management Solutions From End to End, SG24-8226
- Content Manager OnDemand for z/OS, System Monitoring, whitepaper
- Implementing Content Manager OnDemand Solutions with Case Studies, SG24-7511
- Content Manager OnDemand Guide, SG24-6915
- Content Manager OnDemand Web Enablement Kit Java APIs: The Basics and Beyond, SG24-7646
- Content Manager OnDemand Backup, Recovery, and High Availability, SG24-6444

Related publications

- DB2 for z/OS and OS390 Version 7 Performance Topics, SG24-6129-00
- DB2 UDB for z/OS Version 8 Performance Topics, SG24-6465-00
- DB2 9 for z/OS Performance Topics, SG24-7473-00
- IBM Tivoli Storage Manager Version 5.3 Technical Guide, SG24-6638-00
- Tivoli Storage Manager for z/OS Administrator's Guide, Version 5.2 GC32-0775-02
- Tivoli Storage Manager for z/OS Administrator's Reference, Version 5.2 GC32-0776-02
- UNIX System Services, z/OS Version 1 Release 7, Implementation Guide, SG24-7035-01
- z/OS Version 1 Release 9 Implementation, SG24-7427-00
- z/OS Version 1 Release 10 Implementation, SG24-7605-00
- System Programmer's Guide to: Workload Manager, SG24-6472-03
- z/OS Distributed File Service, zSeries File System Implementation, z/OS V1R10, SG24-6580-03
- z/OS V1R9.0-V1R10.0 MVS Initialization and Tuning Guide, SA22-7591-06
- z/OS V1R10.0 MVS Initialization and Tuning Reference, SA22-7592-18
- z/OS V1R10.0 MVS System Commands, SA22-7627-20

- z/OS V1R10.0 DFSMS Access Method Services for Catalogs, SC26-7394-08
- z/OS V1R10.0 DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Object Support, SC35-0426-07
- z/OS V1R10.0 JES2 Initialization and Tuning Guide, SA22-7532-08
- z/OS: Language Environment Customization, SA22-7564-10
- Considerations for Multi-site Sysplex Data Sharing, SG24-7263-00
- System/390 Parallel Sysplex Performance, SG24-4356-03
- z/OS V1R10.0 UNIX System Services Command Reference, SA22-7802-10.
- z/OS: UNIX V1R8.0 System Services Planning, GA22-7800-10.
- z/OS Distributed File Service zSeries File System Implementation, SG24-6580-03
- z/OS UNIX System Services, Performance: http://www.ibm.com/servers/eserver/zseries/zos/unix/bpxa1tun.html