HP Service Manager Shared Memory Guide

Shared Memory Configuration, Functionality, and Scalability

Document Release Date: December 2014

Software Release Date: December 2014

invent

Introduction to Shared Memory

HP Service Manager shared memory is divided into three main areas:

- User-related memory
- Information Retrieval (IR) process
- System table cache

This document explains how Service Manager uses shared memory as well as how each area must be included in the sizing calculation.

Creation and Use of Shared Memory

Shared memory is a common area of storage that is used by all Service Manager server processes on a host machine. Using shared memory is an efficient means of sharing data between processes. For example, when a user or background process first executes a RAD application, Service Manager loads the application into shared memory. This allows other user processes to execute the same application without the overhead of storing a copy of their own into memory. Subsequent executions of the RAD application will result in Service Manager retrieving the application from shared memory thus decreasing the IO required by each process. This behavior of sharing certain data, such as applications, table definitions, and forms between all users significantly increases program performance because reading data from memory is more efficient than reading from disk memory. The total amount of memory used is additionally decreased because one copy of the application, dbdict or form is shared between all users.

The first process that initiates during the startup of Service Manager on the host machine creates the shared memory block in the size defined by the *shared_memory* parameter in the sm.ini file. The syntax of this parameter is

```
shared_memory:<value>
```

Where <value> is the amount of memory in bytes Service Manager should allocate. This shared memory will be held in the machines virtual memory and thus should not be larger than the virtual memory the machine has available. Information about this allocation can be seen in the sm.log file by locating the line that reads "Creating Resources for system xxxxx with key 0xyyyyyyyyy" (where xxxxx represents the port number you are using and 0xyyyyyyyyy is a system key identifying the shared memory). Every process that connects afterwards attaches to the shared memory that was created by the first process. Information about this attachment can be seen in the sm.log file by locating the line that reads "Attaching to resources for system xxxxx with key 0xyyyyyyyyy.". However, execution of some reports, like sm –reportsem, sm -reportshm and sm –reportlic, will not create shared memory because they are designed to only read data from existing shared memory.

Example: Shared Memory involvement during the login process

The login application is called. Service Manager will check shared memory, find that the login application is not currently stored there, retrieves the code record for the application from the database, and then loads it into shared memory. At this point, Service Manager will begin to execute the login application. Once it determines that the login.prompt form needs to be displayed, it will check shared memory for the form, find that it is not currently stored there, retrieve the login.prompt form from the database and load it into shared memory. Once the form is displayed, the user enters their userid and password, the application then attempts to validate this information against Service Manager's operator file. This method of checking shared memory, retrieving the item from the database, and storing it into shared memory continues throughout the life of the process. For this reason, the first user to log into Service Manager is responsible for loading most of the contents of shared memory, which allows each subsequent user to benefit.

Shared Memory Management

Because the memory in the allocated shared memory area is not released until the Service Manager server processes are stopped, the initial overhead of allocating does not depend on users logging in and out of Service Manager. It is assumed that while the server is operating, users will be logging in and out routinely and they will require continuous access to shared memory. Depending on the information, memory is released for reuse within the shared memory area at different times. For example, User Blocks are freed when the user logs out, and cached items are released when they are no longer needed. Cached information that has changed is flagged obsolete and a copy of the changed record is loaded into shared memory with the first user requesting the record after the change. The obsolete record is freed when the use count of the record is at 0, meaning all user processes using this record have released it from use.

For example, when a RAD application has changed, the old version is flagged obsolete if it is still in use and the new version is loaded into shared memory by the next user requesting the RAD application. This occurs because the system cannot find a current copy of the record in shared memory at that time. The obsolete record is freed when the last user still using the old code record releases it. If the IR area of shared memory reaches its size limit (default is 30 % of total allocated shared memory), the least recently used record is removed from memory to allow for new records to be loaded.

Shared memory in Service Manager is divided into various size blocks of storage. The sizes available (in bytes) are 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, and BIG_ALLOC. Any request for shared storage greater than 8192 bytes is satisfied from the BIG_ALLOC pool. Smaller sized blocks are not allocated one at a time, but instead, come from a pool of similar sized requests. For example, when a request is made for 16 bytes of shared storage and there is no 16 byte block free, Service Manager will allocate a 32K byte block from the lower end of shared memory and divide that 32K area into 16 byte blocks. The first 16 byte block would be returned to the requestor and the remaining 16 byte blocks would remain on the 16 byte block free chain for later allocation. BIG ALLOC starts at the end of the allocated shared memory. When both ends (<8K and BIG ALLOC) meet, Service Manager is out of shared memory.

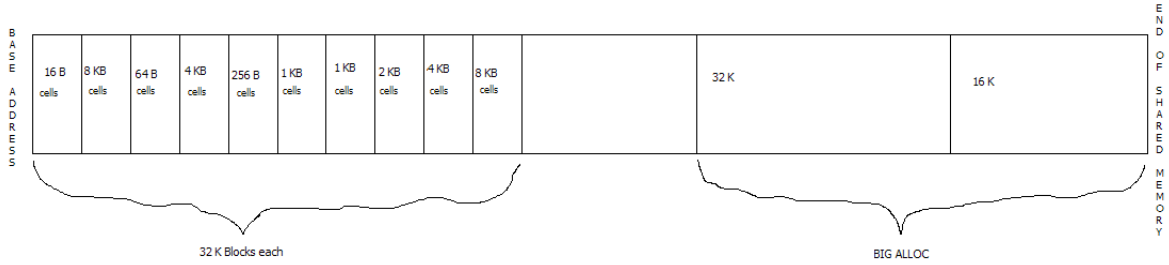

Figure 1: Shared Memory Layout

Shared Memory Sizing

Because the storage area of shared memory is not directly related to the number of users in the system, the per-user shared memory requirement can only be estimated. A user requires 512 bytes of storage for their user block, plus small amounts of storage for various overhead tasks. The user's total shared memory depends on the number of modules used, and the number of forms, dbdict records, links, format controls, and code records they access. The estimated amount of shared memory per user will be higher for a smaller user base, because the cached information in the system is usually relatively static and does not depend on the number of users in the system.

A sample calculation for shared memory is:

```
48 MB + 1MB per 10 users + Shared Memory for IR (see below).
```

Important: Shared memory in a horizontally scaled environment must be calculated per machine based on how many users this machine handles during peak times.

If the Service Manager Information Retrieval (IR) process is used, the IR Cache section of shared memory should accommodate the size of all IR files that are used.

The default size allocated for IR is 30% of the total shared memory. If this amount does not match the recommended size of shared memory for IR, use the **ir_max_shared:<*size_in_bytes*>** parameter to set aside additional shared memory for the IR process.

Shared Memory Adjustments

These suggestions offer a good baseline for most customer systems. However, it may still be necessary to periodically adjust the amount of shared memory. The amount of free storage should be monitored regularly using the Service Manager Shared Memory Report, **sm –reportshm**.

HP recommends never letting the amount of free space fall below 25%. If continuous monitoring of the shared memory report indicates that free space is constantly above 75%, the **shared_memory** parameter in the sm.ini file can be set to a lower number to ensure memory is being used efficiently.

Note: Adjustments to the **shared_memory** parameter must **not** be made when the Service Manager server is running.

Cache Sizing

Cache is stored in slots. By default Service Manager uses 2003 cache slots. The cache report (sm -reportcache) shows the cache slots used and average and maximum depths.

```
C:\scs\sm7\server\RUN>sm -reportcache
---------------------- Cache Statistics      ----------------------

Slot use: 42%; Average Slot Depth: 1; Maximum Slot Depth: 13
```

The average slot depth should be below 3 and the maximum should be below 10. If the average slot depth is too deep, increase the cache_slots parameter to a larger prime number, such as 11001. Prime numbers are most efficient for cache_slots. This is due to the fact that we calculate a hash for each item in the cache resulting in numbers between 0 and 2^{32}-1 (32-bit integer). These hash numbers have to be distributed – if possible evenly – between the available cache slots. This is done with a modulus function against the cache_slots. Using a prime number has a higher probability of distributing the cache items evenly amongst the cache slots.

Unused items are removed from cache by the sync process. During the cache_clean_interval, the sync process enters a timestamp into all cache items that have a use count of 0. During the next cache_clean_interval, if it still shows the timestamp the item will be removed from cache. If the cache item was used after the first interval, the use count overwrites the timestamp indicating the item has been recently used. Thus cache items that were unused for cache_clean_interval (seconds) * 2 will be removed from the cache.

Shared Memory Reports

You can monitor the allocation and growth of shared memory by using the Service Manager Shared Memory Report, **sm –reportshm**. Figure 2 is an example of a shared memory report. The report contains a list of all the items currently stored in shared memory in addition to the amount of storage each item is using.

```
03/11/10 13:57:11   pid (6208) HP Service Manager diagnostic report follows:

------ Shared Memory ------

Shared Memory Release     9.20

Current Size              32000000

Segment Allocation         6460760
Large Block Allocation     9060864

Unused Space              16478376  (51%)
Free Space                19875712  (62%)


Shared Memory Type  Allocations       Frees   Allocated
------------------  -----------  ----------- -----------
Not named                   110           0       56400
User blocks                   0           0           0
Messages                      0           0           0
Resource locks               86           0        8144
Database Services           139           0       17904
Cache overhead                3           0       10256
Application cache          5294        2779     3726992
DBDICT cache              12096       10974     6922560
SQL descriptor cache        544         378      438400
Join/ERD/Type cache         620           0      823296
String Type cache           532         442       23040
JavaScript Members            9           0         784
IR Expert cache             375           0       91808
```

Figure 2 Service Manager Shared Memory Report

Current Size is the amount of shared memory allocated in bytes as defined by the *shared_memory* parameter in the sm.ini file.

Free Space is the amount of shared memory that is available for use by processes. Free Space is gained when previously used Shared Memory areas are made available again. Also, Free Space is the sum of "Unused Space" (memory never before allocated) and "Freed Space" (memory in the free list chain that was previously allocated) Free Space is shown as total bytes and as an overall percentage.

Unused Space represents available shared memory that has not yet been allocated. When Service Manager starts, the entire amount of memory defined by the *shared_memory* parameter is allocated, and blocks may

then be allocated from within that shared memory as needed. Unused Space is shown in total bytes and as an overall percentage.

Using *sm -reportshm:1* to generate the Service Manager Shared Memory Report will produce a more detailed listing of shared memory allocation.

Service Manager Semaphore Report

The Service Manager Semaphore Report, **sm –reportsem**, produces a list of all semaphores and handles currently held by Service Manager. This report can be useful when debugging shared memory problems.

```
03/11/10 14:11:50   pid (7552) HP Service Manager diagnostic report follows:
--- Reportsem ---

[0]System               Available  count(17)
[1]Application cache    Available  count(4607)
[2]Shared memory        Available  count(36026)
[3]IR Expert            Available  count(24)
[4]Licensing            Available  count(240)
[5]Resource manager     Available  count(22901)
[6]User chain           Available  count(356)
[7]Cache manager        Available  count(232753)
[8]Database Services    Available  count(175968)
[9]Alert Services       Available  count(4523)
[10]Counter Services    Available  count(20)
```

Figure 3 Service Manager Semaphore Report

Shared Memory Areas

Not named = User Blocks (area moved)

Every user process allocates a user block of 512 bytes. This block of shared memory contains user specific information like userid, device, license (specific to the user, such as floating or named license, when last logged in etc.), and system monitor information.

Messages

Every message that is sent from one user to another is put into the message block section of shared memory. The amount of shared memory allocated per message depends on the size of the message.

Resource Locks

Resource locks are created whenever the lock application is executed from RAD. For example when a user edits an incident ticket, a resource lock is created for that ticket. If the 'show locks' option from the system status is selected and you have a vertically-scaled environment, the information on existing locks is taken from shared memory for all versions of Service Manager. However, in horizontally-scaled environments, the information is taken from JVM private memory of each node for versions of Service Manager earlier SM 9.31, or from the database for SM 9.31 and later versions.

Database Services

Tuning statistics regarding executed queries and database access are stored in this area. Statistics indicating possible performance issues are written to the sc.alert.log.

Cache Overhead

Contains operators and other system information; It also contains overhead or miscellaneous items that need to be stored for multi-process access

Application Cache

This area in shared memory holds code records, pre-parsed and split up by panel.

DBDict Cache

This area contains cached Table Definitions (dbdict records) as well as other cached data records such as: format, link, format control, category, scmessage.

SQL Descriptor Cache

This area in shared memory is used by the RDBMS mapping engine and contains the mapping information between the dbdict and the RDBMS table definitions.

JOIN/ERD/TYPE Cache

This area contains records from joindef, erddef, scaccess, scmandant and sctypecheck

String Type Cache

This contains the fastcounters data structure, files and fields using counters, cascade delete definitions, cached charts, list of defined triggers, JavaScript triggers that for ease of Shared Memory storage are wrapped into a STRING.

JavaScript Members

This area contains information used by or required for JavaScript applications. It does not contain JavaScript code, but rather information on the current version of the Script member.

IR Expert Cache

This area in shared memory holds the term and document lists used by Service Manager's IR expert process.

IR Expert and Shared Memory

IR Expert makes extensive use of shared memory and usually represents the largest allocation type. The default amount of shared memory used by IR is 30% of the total size allocated for shared memory. This percentage can be adjusted by changing the amount of bytes used for IR in Shared Memory using the **ir_max_shared** parameter in the sm.ini file. The optimum size of the cache is the same as the size of all ir.* files added together.

In general, IR keeps track of all documents that contain search words. The document list for a word is loaded into shared memory as it is referenced. Subsequent searches on the same word will be much faster because the first process that ran the search has already loaded the document into shared memory.

If the IR shared memory area is full, Service Manager will remove the least recently used data structure (such as a term, doc, doc list, term list, adaptive learning) from shared memory to make room for new IR information.

Increasing the overall size of shared memory will only have a performance impact on the IR Expert portion. It will NOT help with general system performance.

Horizontal Scaling: Special Considerations

Each horizontally scaled host machine has its own shared memory calculated based on the amount of users supposed to be running on that machine. The IR expert section has to be added to all machines in the group, which makes the calculation for each machine:

```
48 MB + 1 MB per 10 users + IR Size
```

In the horizontally scaled system messages to users as well as Cache updates on changed cached records are communicated to all shared memory sections of all host machines in the group.

Locks are not held in shared memory at all in horizontally scaled systems. Locks are handled differently depending on your version of Service Manager. For versions of Service Manager earlier than Service Manager 9.31, locks are communicated by using UDP Multicasting between the systems and the locking information is held in the JVM. For Service Manager 9.31 and later versions, locks are stored in the central RDBMS. For more information on how locks are managed in Service Manager 9.31, see the Lock Management topic in the Service Manager Help Server.

Troubleshooting

As soon as less than 25 % of shared memory is unused, it is recommended that the amount of shared memory be increased by modifying the **shared_memory** parameter in the sm.ini file. Unused shared memory is the amount of memory that has never been allocated. The real percentage of available memory is always higher than what the unused percentage in the Service Manager Shared Memory Report depicts.

Because shared memory can become fragmented, leaving at least 25% of free space is recommended. This will ensure that fragmentation does not prevent new space from being allocated within the shared memory area.

The ***shared_memory_address:<memory address>*** parameter can be used on all operating systems to ensure that shared memory allocation is kept in an area that will not interfere with shared libraries, stacks, heaps or other memory related data structures. You can use this parameter to specify a new address that has enough contiguous memory for the HP Service Manager process.

If you encounter a shared memory allocation failure, the address that the HP Service Manager process is using does not have enough contiguous memory. On Windows, the sm.log file lists recommendations for possible **shared_memory_address** parameter values. If the debugvmmap parameter is active, the log also contains a map of virtual memory at the time of the failure.

For more details, see the following Service Manager online help topics:

- Startup parameter: shared_memory_address
- Troubleshooting: Failed to initialize or attach to shared memory environment (Windows only)

Note: The defaults should be working for all platforms. You should only use this parameter when instructed to do so by HP Customer Support.

Shared Memory Support Information

Shared memory problems can be some of the most difficult to resolve. However, if problems with shared memory arise, you can contact HP Customer Support with the following information ready to help expedite a satisfactory solution:

- Detailed description of the issue
- Copies of all Service Manager log files

- Copies of any core files that were generated
- Service Manager Shared Memory Report
- Service Manager Semaphore Report

For more information

Please visit the HP Software support Web site at:

www.hp.com/go/hpsoftwaresupport

This Web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued customer, you can benefit by being able to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Submit enhancement requests online
- Download software patches
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Note: Most of the support areas require that you register as an HP Passport user and sign in. Many also require an active support contract.

To find more information about support access levels, go to the following URL:

www.hp.com/go/hpsoftwaresupport/new_access_levels

To register for an HP Passport ID, go to the following URL:

www.hp.com/go/hpsoftwaresupport/passport-registration

© 1994–2012 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

HP and Service Manager are registered trademarks of Hewlett-Packard Development Company, L.P. JavaScript is a registered trademark of Sun Microsystems, Inc. in the United States and other countries.