

Published: 10/25/68  
(Supersedes: BD.3.05, 12/20/67)

## Identification

Summary of Calls to the SMM "write-around"  
John W. Gintell

## Purpose

The smm "write-around" provides the interface between calls to the old segment management module (smm) and the new BFS calls so that calls made to the smm will be properly directed to the file system. These calls are available outside of ring 0.

Note: The term "callname" used in this document has the same meaning as "reference name" in the overview BG.18.00 and is retained in recognition of the nomenclature of the earlier segment management facility.

## Available Calls

1. smm\$initiate (callname, dpath, ename, copysw, segptr, status);  
initiates the segment located by path name "dpath>ename" for the call name callname.

It returns segptr, the pointer to the initiated segment and status, an indication of the results of the initiation.

status = 0 means segment segptr initiated as per request

1 segment segptr previously initiated for callname

2 unable to initiate the segment indicated by "dpath>ename"

copysw is used to determine whether an original version or a copy of the segment is to be obtained.

copysw = 0 means use the value of the copy switch in the branch

copysw = 1 means use the original segment

copysw = 2 means obtain a copy of the original segment

2. `smm$get_segment (callerptr, callname, relname, copysw,  
                  segptr, relptr);`  
gets two segments:
- one for the call name callname as wanted by the segment callerptr
  - one for the call name callname||relname and related to the segment found for a. If relname is the null string, "", then no such related segment is obtained.

copysw is applied as in initiate above to the related segment only. It returns pointers to these segments, segptr and relptr, respectively.

A null pointer indicates that no segment was found.

3. `segptr = smm$get_seg_ptr (callname, callerptr);`

returns segptr, the pointer to the segment (previously initiated for the call name callname) which is available to the segment callerptr. A null pointer indicates that no segment was found.

4. `smm$get_path_name (segptr, dirname, entryname);`

returns the path name of the segment segptr. dirname is the directory in which the segment resides and entryname is the last call name initiated for that segment.

5. `smm$set_name_status (callname, dpath, ename, msegptr,  
                      scirgco, maxsize, trewa, segptr, unname,  
                      status);`

action taken depends on the value of scirgco (which represents Search, Create, Initiate, Relate, Global and Copy).

Relate and Global are always ignored. If Initiate is off no action is taken and return is made with segptr set to null.

status is always returned set to 0.

For all other values of scirgco, segptr points to the initiated segment or is set to null if the segment could not be initiated.

For those values of scirqco for which dpath has any meaning, if it is the null string: "", it is taken to mean the name of the process directory.

If Create is on, Search and COpy are ignored and a segment whose call name is callname is placed in the directory named dpath with maximum size maxsize, access attributes trewa, and ring brackets {validation level, validationlevel, validationlevel}. The entry name will be ename or if ename = "" it will be the unique characters generated by the uniqued\_id.

If Create is off, then an attempt is made to initiate a segment named callname in the directory named dpath. If not successful and if Search is on the search rules are used to search other directories for an entry named callname.

The fixed binary value of the COpy bits are used as is copysw in the entry smm\$initiate. (If CO = "00"b then the value of the copy switch in the branch is used, if "01"b the original segment is used, if "10"b a copy of the original segment is initiated when a copy is made).

uname is returned with the unique name of the entry in the process directory for the copied segment.

6. smm\$terminate (callname, callerptr);

terminates the segment whose call name is callname when referenced by the segment to which callerptr is a pointer.

#### Argument Declarations

dc1 (callname, dpath, ename, relname)

char (\*) varying [or char (\*)];

dc1 (segptr, callerptr, relptr, msegptr)ptr;

dc1 smm\$get\_seg\_ptr ext entry ptr;

dc1 copysw fixed bin (2);

/\* = 0 means use the copy switch setting in the hierarchy

= 1 means use the original segment

= 2 means make and use a copy of the segment\*/

