TO: MSPM Distribution
FROM: Susan Rosenbaum
SUBJECT: BD.3.05
DATE: 12/20/67

Attached is a list of the Segment Management Module primitives and a brief description of those primitives which are currently available. Section BD.3 is in the process of being revamped and will be forthcoming. Ultimately, section BD.3.05 will consist solely of a summary of the calls to Segment Management.

Major Changes to Segment Management Module Primitives:

1. Any input argument which is a character string may be declared either varying or non-varying.

2. The addition of the primitive get_path_name which returns the path name of a segment given the pointer to the segment.

3. Centralizing calls to the primitives by routing them through the segment smm.

4. New calling sequence for the primitive set_name_status. Now the user can set the maximum size and the attributes for segments he wishes to create.

5. New names for the primitives.

	Old Name	New Name
a.	initiate	smm$initiate
b.	getseg	smm$get_segment
c.	getsegptr	smm$get_seg_ptr
d.	--------	smm$get_path_name
e.	setnamestatus	smm$set_name_status
f.	terminate	smm$terminate
g.	setdel	smm$set_del_sw
h.	setlock	smm$set_lock
i.	getnamestatus	smm$get_name_status
j.	getsegstatus	smm$get_seg_status
k.	getname$segment	smm$get_seg_name
l.	getname$daughter	smm$get_daughter_name

Published: 12/20/67

Identification

Summary of Calls to the Segment Management Module
Susan L. Rosenbaum

Purpose

This section briefly summarizes all of the Segment Management Module primitives which are currently available to the user and will be updated as calls are added; section BD.3.02 fully describes each of the calls.

Available Segment Management Module Primitives

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"

2. smm$get_segment (callerptr,callname,relname,copysw,segptr, relptr);

   gets two segments:

   a. one for the call name callname as wanted by the segment callerptr

   b. one for the call name callname.relname and related to the segment found for a.

   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 off which the segment resides and entryname is the name of its entry in that directory.

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

   sets up an entry for callname in the Segment Name Table. It returns

   segptr = null if entry not initiated
   else segptr points to the initiated segment.

   uname is the unique name of the entry in the Process Directory for the copy of the segment (if copying was requested).

   status = 0 indicates request was carried out successfully. (Currently there are no other meaningful values for status.)

Implementation

```
dcl  (callname,dpath,ename,relname)
     char(*) varying [or char(*)];

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

dcl  smm$get_seg_ptr ext ent ptr;

dcl  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 */
```

```
dcl status fixed bin(17);

dcl (dirname,entryname) char(*) varying [or char(*)];

dcl uname char(15);

dcl scirgco bit(7);          /* represents Search, Create,
                                Initiate, Relate, Global and
                                COpy switches */

dcl maxsize fixed bin(9);    /* maximum size for created
                                segment - number of 1024 word
                                blocks */

dcl trewa bit(5);            /* represents Trap, Read, Execute,
                                Write and Append attributes for
                                created segment */
```