Published: 05/05/67

Identification

Segment Management Module Primitives
S. L. Rosenbaum

Purpose

The SMM maintains records of all segments known to the current process by symbolic call names. The SMM keeps the information concerning the segments and their names in a per-process table, the Segment Name Table (SNT). This section describes the SMM's primitives and their use.

The SMM has two main functions:

1. It makes a segment available (or unavailable) by symbolic call name to the current process, inserting (or removing) the appropriate entries in (from) the SNT.

2. It provides various service functions for the current process with respect to segments and names already available to the current process. These functions consist of retrieving and altering information already in the SNT.

Introduction

The SMM's primitives are all unprivileged and designed to be used by both system programs and the general user. As a reminder, a segment is initiated, if it is known to the current process by at least one symbolic call name, i.e., a Segment Header containing information about the segment appears in the SNT. A call name is known if it appears in the SNT, i.e., the name has a Name Header in the SNT. A call name is initiated only if it is associated with an initiated segment, i.e., its Name Header is associated with at least one Segment Header. A terminated name is a call name which is not associated with a segment. A call name is a related name if it is directly associated with a segment which references the name. The segment is called the "mother" and each of its related names is called a "daughter".

Primitives

The SMM provides the primitives listed below. If an error occurs, the SMM follows the error handling procedure described in sections BY.11.00-04.

A description of each primitive follows the list.

1. initiate - initiates the given call name for the segment located by a given path name
2. terminate - terminates the given call name
3. getseg - gets two pointers: the pointer to the segment for the given call name and the pointer to the segment's linkage section segment (designed primarily for the Linker)
4. setdel - sets a segment delete switch
5. setlock - sets a segment's usage lock
6. setnamestatus - makes an entry for a given call name
7. getnamestatus - gets information about a given call name
8. getsegstatus - gets information about a given segment
9. getsegptr - gets the pointer to the segment associate with a given call name
10. getname - gets a call name available to the given caller

The first six primitives alter the information in the SNT; the remaining primitives retrieve existing information from the SNT - signalling errors if the desired information is not in the SNT.

1. initiate

The SMM provides a primitive with which the process can initiate a given segment for a given call name for the first time. The call:

```
call initiate (name, dpath, entry, copysw, segptr)
```

creates an initiated entry which associates the call name name with the segment having the entry in the directory dpath, i.e., the path name pathname of the segment is "dpath>entry".

copysw is a switch where

copysw = 0 means use the setting of the copy switch on the branch in the file system

1 means get the segment pathname

2 means get a copy of segment pathname

The SMM puts copies of segments in the Process Directory. (The SMM sends an error signal if the copy switch in the branch is "on" and the caller has requested the original segment rather than a copy.)

The SMM looks in the SNT for a known entry for the call name, name. If it finds one which is not a related entry, the SMM initiates the name if necessary, returns with the pointer to the entry's associated segment, segptr, and generates an error. (The initiate primitive is for the purpose of creating for the first time a known non-daughter entry which is available to any segment needing a segment for the call name name. Hence, if such an entry existed in the SNT prior to the call to initiate, the SMM generates an error. It is left to the caller of initiate to determine if this error condition (the prior existence of the entry) is fatal to the process. That is, the demand that the segment for name be located in the file system hierarchy by dpath and entry might be desirable but not necessary and, in fact, the existing entry might be associated with that segment.)

If at this point, the SMM does not generate an error, it takes the following steps to create a new entry for the calling name, and to initiate it.

The SMM invokes the Directory Control primitive estblseg to get the pointer to the segment. If estblseg indicates that the segment is actually a relationship segment i.e., the segment sought has related names, the SMM creates a known (but not initiated) entry for each daughter. (See BD.3.00 for a discussion of "related names".)

The SMM expects a relationship segment to contain the actual path name of the mother segment and the following information for each daughter.

1. daughter's call name
2. global usage switch - ON/OFF
3. create switch - ON/OFF
4. search switch - ON/OFF
5. initiate switch - ON/OFF
6. directory path name
7. entry name

where the global usage switch indicates whether or not the daughter entry can be used by all segments looking for its call name (item 1) or can be used only by the mother segment. The directory path name concatenated to the entry name produces the path name of the segment for the daughter. If the create switch is "on" (indicating that the segment for daughter does not exist in the file hierarchy and must be created)- or - if the search switch is "on" (indicating that the segment may be searched for in the hierarchy by the Search Module if not found in the hierarchy by the path name specified (item 6 concatenated with item 7), then items 6 and 7 are set (or reset) when the segment is actually found. The initiate switch (item 5) indicates whether or not the daughter should be initiated at the same time as mother. Finally, the SMM updates the SNT to include the new initiated entry (and its Segment Header if necessary) and returns to the caller. The new entry is available to faulting segments in the ring from which initiate was invoked. (Note: no entry remains for the relationship segment. To actually get the relationship segment see section BX.8.13.)

2. terminate

terminates a call name in the SNT. The call:

call terminate (name, callerptr)

removes from the SNT the entry for name which is:

1. available to the segment pointed to by callerptr
2. available to segments residing in the ring from which terminate was called

If there are no other entries associated with the segment with which this entry is associated, the SMM takes additional steps to:

1. delete all related entries local to the segment (available only to the segment)
2. delete all related information from the global entries which are related to the segment, i.e., "unrelate" all other related entries
3. delete the segment's Segment Header

The SMM then updates the SNT and returns to its caller.

3. getseg

The getseg primitive is intended primarily for the convenience of the Linker and, as such, is a multi-purpose primitive. Variations of the basic calling sequence direct the SMM to:

1. obtain the segment pointer for a given call name
2. make a known entry for a call name and relate it to a segment (1 above)
3. obtain the segment pointer for a related name (i.e., initiate 2 above)

The call to getseg is:

```
call getseg (callerptr, name, relname, copysw, segptr,
             relptr)
```

where

callerptr is the pointer to the faulting segment, i.e., the procedure segment which wants a segment for name.

name is the call name for which a segment is wanted.

relname is a character string which when concatenated with name produces the call name for the related segment desired. (For example, the Linker sets relname equal to ".link" to get the linkage section; relname = ".symbol" represents the symbol table.)

copysw is the copy switch where

copysw = "00"b means use the copy switch in the hierarchy for the segment for name.relname

= "01"b means use the segment already in the hierarchy for name.relname

= "10"b means make and use a copy of the segment in the hierarchy for name.relname

The SMM returns segptr, the pointer to the segment for name, and relptr, the pointer to the segment for the related segment. (Actually, since the SMM always makes a copy of a linkage section segment for the Linker, linkptr points to the segment in the Process Directory which is a copy of the linkage section segment.) The SMM relates the call name, name.relname, to the segment segptr.

The entry for name must be:

1. available to the faulting ring

2. local to segment callerptr or global

If there is no such known entry, the SMM calls the Search Module at search. The Search Module locates the segment for name in the file system hierarchy and returns the segment's path name to the SMM. The SMM gets the pointer to the segment by calling estblseg. The SMM then creates a known and related entry for the call name of the related segment - always assuming that the segment is in the same directory as the segment for name. The SMM calls estblseg to get the segment pointer for the related segment. Via calls to appendb, estblseg again and makeunknown, (if copying is requested) the SMM puts a copy of the segment into the Process Directory, assigns this copy a unique name, gets this segment's pointer and frees the original segment. Finally, the SMM returns the two pointers to the caller of getseg (normally the Linker).

When the SMM finds an entry for name already in the SNT, it makes sure there is a related entry for the related segment (making one, as described above, if necessary). The SMM then simply returns the two pointers.

Variations

a. If the caller supplies a null pointer for relptr, the SMM does not return the second pointer. Indeed the SMM only makes sure that a known entry for the related name is in the SNT - the SMM does not attempt to initiate the entry.

b. If the caller submits a null character string for relname, the SMM returns directly to getseg's caller after getting the segment pointer for name.

Examples:

To obtain the segment pointer to the segment associated with the call name "x" which is needed by the faulting segment callerptr:

```
call getseg (callerptr, 'x'," ", "0"b, xptr, null) (I)
```

To relate the linkage section for "x" in addition to (I):

```
call getseg (callerptr, 'x', ".link", "10"b, xptr,
             null) (II)
```

To get the pointer to the linkage section in addition to (II):

```
call getseg (callerptr, 'x', ".link", "10"b, xptr,
             xlinkptr) (III)
```

(Note: If the segment for "x" has the path name ">a>b", then the SMM uses the segment ">a>b.link" for the original linkage section. The SMM copies the segment, puts the copy into the Process Directory, pdir. and assigns the entry in the Process Directory some unique name, uname. The SMM creates an SNT entry for the call name "x.link" associates the segment "pdir > uname" with the call name and relates the entry to the segment xptr.)

c. If the caller submits a null character string for name, the SMM returns the pointer to the segment for relname related to segment segptr. The caller must specify segptr and relname.)

4. setdel

call setdel (segptr, delsw)

sets the segment delete switch for segment, segptr, to delsw where:

delsw = 0 means do not delete this segment from the file system hierarchy when the segment is no longer known to the current process by any name, i.e., when there are no entries for this segment and its Segment Header is removed from the SNT.

delsw = 1 means do delete the segment when it is unknown to the current process, i.e., when its Segment Header is deleted from the SNT. The SMM sets delsw to zero when it initiates the segment for the first time, i.e., when it first creates a Segment Header for the segment.

5. setlock

call setlock (segptr, lock blocksw)

sets the segment usage lock for segment, segptr to lock where:

lock = 0 means unlocked

1 means read-locked

2 means write-locked

3 means data-share-locked

blocksw = 0 means "return if lock cannot be set and signal an error".

1 means "do not return until lock is set".

6. setnamestatus

The SMM provides the primitive setnamestatus (designed primarily for file system library procedures) which enables the user to request a very specific initiation. This primitive is available to the general user but it is recommended that the casual user use the library procedures or the file system commands.

call setnamestatus (name, dpath, entry, isw, segptr, rsw, msegptr, gsw, csw, maxsize, ssw, copysw, uname)

supplies the information which describes the initiation desired. The caller must supply the information necessary to explicitly define the desired initiation; the SMM will set the unspecified information based on the information that is supplied explicitly.

a. name is the symbolic call name to be initiated. If name is null, the SMM obtains a name which is unique to the directory, dpath.

b. dpath is the path name of the directory in which the segment resides or is to reside. If dpath is null, the SMM assumes it is the Process Directory.

c. entry is the entry name of the segment in directory dpath.

d. isw is the initiate switch where

isw = 0 means do not initiate it now

isw = 1 means do initiate it now

If isw is null then the SMM assumes isw = 0.

e. seqptr is the pointer (ITS pair) to the base of the segment. The SMM returns this value only if initiation took place.

f. rsw is the related switch where

rsw = 0 means it is not to be related, i.e., it is not to be a daughter

rsw = 1 means it is to be related, i.e., it is to be a daughter

If rsw is null then the SMM assumes rsw = 0.

g. mseqptr is a pointer (ITS pair) to the base of the mother segment. If mseqptr is null and the related switch is on (rsw = 1), the SMM assumes that the caller of setnamestatus is the mother segment.

h. gsw is the global usage switch where

gsw = 0 means this initiation is only for the mother segment

gsw = 1 means this initiation is for the use of any segment needing a segment for the call name name.

If gsw is a null bit string, the SMM makes the entry global, i.e., sets gsw = 1. (if rsw = 0, gsw ≠ 0.)

i. csw is the create switch where

csw = 0 means the segment wanted is located in the file system hierarchy by the path name "dpath>entry".

csw = 1 means the segment is not currently in the hierarchy and an entry by the name entry should be created in directory dpath.

If entry is null then the SMM will get a name for the entry which is unique to directory dpath.

j. maxsize is the maximum size that should be set for the created segment. (If csw = 0, the SMM ignores maxsize.)

k. ssw is the search switch where

ssw = 0 means the SMM should not invoke the Search Module to find the segment.

ssw = 1 means the SMM should invoke the Search Module if the segment cannot be found at "dpath>entry".

If ssw is null then the SMM assumes ssw = 1.

l. copysw is the copy switch where

copysw = 0 means the copy switch attached to the branch "dpath>entry" in the hierarchy will be used.

copysw = 1 means the segment "dpath>entry" (or the segment created) (or the segment found by the Search Module) is to be used for name.

copysw = 2 means the segment is to be copied and the copy is to be associated with the call name name.

When the SMM copies a segment, it obtains a unique name for the copy and puts it in the Process Directory with the unique name as the segment copy's entry name.

m. uname is the unique name assigned to the copy of the segment in the Process Directory and returned by the SMM to the caller.

The entry is available to the ring of setnamestatus's caller.

7. getnamestatus

The SMM provides the primitive getnamestatus for obtaining information about a known (not necessarily initiated) symbolic call name.

The call

```
call getnamestatus (name, callerptr, ringno, gsw, rsw,
                    isw, csw, ssw, dpath, entry, uname,
                    msegptr, segptr)
```

obtains information about the SNT entry which is used by the segment callerptr for the call name name. The information returned consists of:

a. ringno - the number of the ring to which the entry is available, i.e., this entry can only be used by segments which fault in ring ringno.

b. gsw - the global usage switch where

gsw = 0 means this entry is local i.e., is a daughter and may only be used by its mother, segment callerptr.

gsw = 1 means this entry is global, i.e., may be used by any segment which needs a segment for the call name name.

c. rsw - the related segment switch where

rsw = 0 means this entry is not a daughter

rsw = 1 means this entry is a daughter.

d. isw - initiated switch

isw = 0 means this name is not associated with a segment, i.e., known but not initiated.

isw = 1 means this name is initiated.

e. csw - the create switch where

csw = 0 means the segment is to be or was obtained from the file system hierarchy

csw = 1 means the segment is to be or was created by the current process.

f. ssw - the search switch where

ssw = 0 means a search is or was not desired to get the segment.

ssw = 1 means a search is or was desired to get the segment.

g. dpath - the path name (relative to the root directory) containing the entry for the segment.

h. entry - the entry name of the segment in directory dpath.

i. uname - the unique name assigned to the copy of the segment. uname is null if no copy was made.

j. msegptr - the pointer to the mother segment. msegptr is null if this entry is not a daughter.

k. segptr - the pointer to the segment for this name. segptr is null if this entry is not initiated.

8. getseqstatus

The SMM provides the primitive getseqstatus for obtaining information about a known segment. The call

```
call getsegstatus (segptr, dpath, entry, slotlist, lock,
                   delsw, numnames, numdaughters)
```

obtains information about the segment segptr.

dpath is the path name of the directory in which the segment resides.

entry is the entry name of the segment in directory dpath.

slotlist describes the segment's position in the file system hierarchy and is used for interprocess communication. Section BD.3.01 discusses slotlists.

lock is the segment usage lock.

delsw is the segment delete switch.

numnames is the number of names by which this segment is known to the current process, more exactly, the number of entries in the SNT associated with this segment.

numdaughters is the number of names related to this segment, i.e., the number of daughter entries for this segment.

9. getseqptr

The call:

```
segptr = getsegptr (name, callerptr)
```

gets the pointer to the segment, segptr, which the segment callerptr uses for the call name name. The (entry for the) segment must be initiated prior to the use of getseqptr.

10. getname

gets the ith call name

1. associated with a segment or
2. related to a segment.

(I) name = getname$segment (i, segptr)
gets the call name of the ith entry associated with the segment segptr.

(II) name = getname$daughter (i, msegptr)
gets the call name of the ith entry related to segment msegptr.

The declarations for the arguments of the various primitives follow.

```
dcl (name, dpath, entry, relname, uname) char (*) varying;

dcl (delsw, blocksw, isw, rsw, gsw, csw, ssw) bit (1);

dcl (copysw, lock) bit (2);

dcl segptr, callerptr, relptr, msegptr) ptr;

dcl uid bit (70);

dcl (ringno, slotlist (*), numnames, numdaughters, i)
     fixed bin (17);

dcl maxsize bit (9)
```