TO: MSPM Distribution
FROM: John W. Gintell
DATE: October 25, 1968
SUBJECT: Segment Management Facility

The attached issue of BG.18.00, BG.18.01 and BG.18.02 document the new segment management facility which has been installed as part of the basic file system.

Sections BD.3.00, BD.3.01, BD.3.02 and BD.3.05 which document the old segment management module are superseded. Sections BD.4.05 and BD.4.05A which describe the interim search module are also superseded.

There are four points of significance to be stated:

1. The "search rules" are built into the code of the basic file system and therefore are the same for all users. They are, in fact, identical to the default search rules which were implemented in the old system.

2. The segment management facilities are available to user calls either through the old smm interface or through the new BFS entries.
   Neither of these is a guaranteed-to-be-upward-compatible file system to user interface; as a result, if there is file system redesign these calls may have to be changed.

3. If a user wishes to obtain a pointer to a segment which may not be in the hierarchy and only for the case of not being in the hierarchy does he want to create a new segment, the following logic must be used:

   a. Call to get a pointer to the desired segment.

   b. If this pointer is null, then call to create the desired segment.

4. There is a file system restriction that the length of any segment name be $\leqslant 32$. Allowance must be made in choosing names for those segments which have associated segments whose names are constructed by appending a suffix to the original name. (E.g., if "x" is a procedure name the length of x must be $\leqslant 25$ to allow for x.symbol.)

5. Unless a name is explicitly terminated it remains in the KST for the life of a process. Because of the implementation the following problem can occur: if the name "a" is initiated and then never terminated for the segment whose pathname is p1, a later attempt to initiate the name "a" for the segment whose pathname is p2 will return a pointer to the previously initiated segment p1.

Published: 10/25/68
(Supersedes: BD.3.00, 04/05/67;
BD.3.01, 05/04/67;
BD.4.00, 03/24/67;
BD.4.00, 08/18/66;
BD.4.05, 01/03/67)

Identification

Overview of the Segment Management Facilities in the BFS.
John W. Gintell

Purpose

The segment management facilities in the BFS provide an interface with segment and directory control used to make segments known, make them unknown, search the hierarchy in a specified way for segments whose pathnames are not known, and to obtain information about segments which are currently known to a process.

Introduction

In Multics a reference from one segment (normally a procedure) to another may be made by name. Multics requires that a segment be associated with the name and that a number be used for the actual reference. The name does not uniquely identify which segment in the hierarchy is required, so that it is necessary to apply rules to discover which segment is desired prior to assigning a segment number to it and bringing it to a state where it can be referred to by segment number via the hardware/software environment. The segment management facility of the BFS is responsible for implementing the mapping between these names and segment numbers. It must decide which segment to obtain and must make the necessary calls to bring it to the state where it may be written, read, or transferred to by the segment which referred to it.

The segment management facility also implements various service functions which are required by other system modules and users: supplying a pathname corresponding to a given segment number, bringing a segment whose position in the hierarchy is known to the state in which it may be referenced, or even creating a new segment in the hierarchy.

The problem of mapping names by which inter-segment referencing is made onto segments is complicated by the fact that it is desired for this mapping not to be unique within a given process. (E.g., when segment <a> in process p refers to a segment by name "n" and when segment <b> in process p refers to a segment by name "n" the two references to "n" might be directed to different segments.) For Initial Multics the mapping will be restricted so that within a ring, all references to a particular name map onto the same segment. Later it will be necessary to relax this restriction so that all references to a particular name from a segment map onto the same segment but that references to a name from different segments need not map onto the same segment.

<u>Discussion</u>:

Some definitions over and above the basic file system definitions are needed.

A <u>reference name</u> is the name by which one segment refers to another segment.

A reference name is <u>initiated</u> in a particular ring if there is a known segment which corresponds to the reference name.

A reference name is <u>terminated</u> in a particular ring when the correspondence between the reference name and a previously initiated segment is removed.

We say a <u>segment is initiated</u> if there is at least one reference name initiated for the segment.

We say a <u>segment is terminated</u> if all reference names are terminated and the segment is made unknown.

The key to the design is based on the following properties of the KST:

1) each segment which is known to a process has a KST entry which may have one or more names

2) there is a means for rapidly searching the KST by name.

In order to implement the segment management facility there must be a place to remember the names corresponding to a segment. Since the name uniqueness has been restricted to requiring that a given name map onto only one segment within one ring, names are made unique by concatenating the ring number to the actual name. It is this name which is entered into the KST entry for the segment for which it is initiated for future reference. When a new reference name is to be initiated for a segment it is added onto the end of the list of names for the segment.

Provision is made in the segment management facility for searching a selected set of directories in the hierarchy for the presence of a segment. This is used when a reference name is to be initiated but the exact location in the hierarchy of the corresponding segment is not known. Because one of the directories to be searched is the working directory, the name of the working directory is stored in the KST for easy access. The names and order of the directories to be searched are encoded into the segment management facility.

An entry to the segment management facility is available which creates a branch in a specified directory for a new, empty segment and then initiates this new segment.

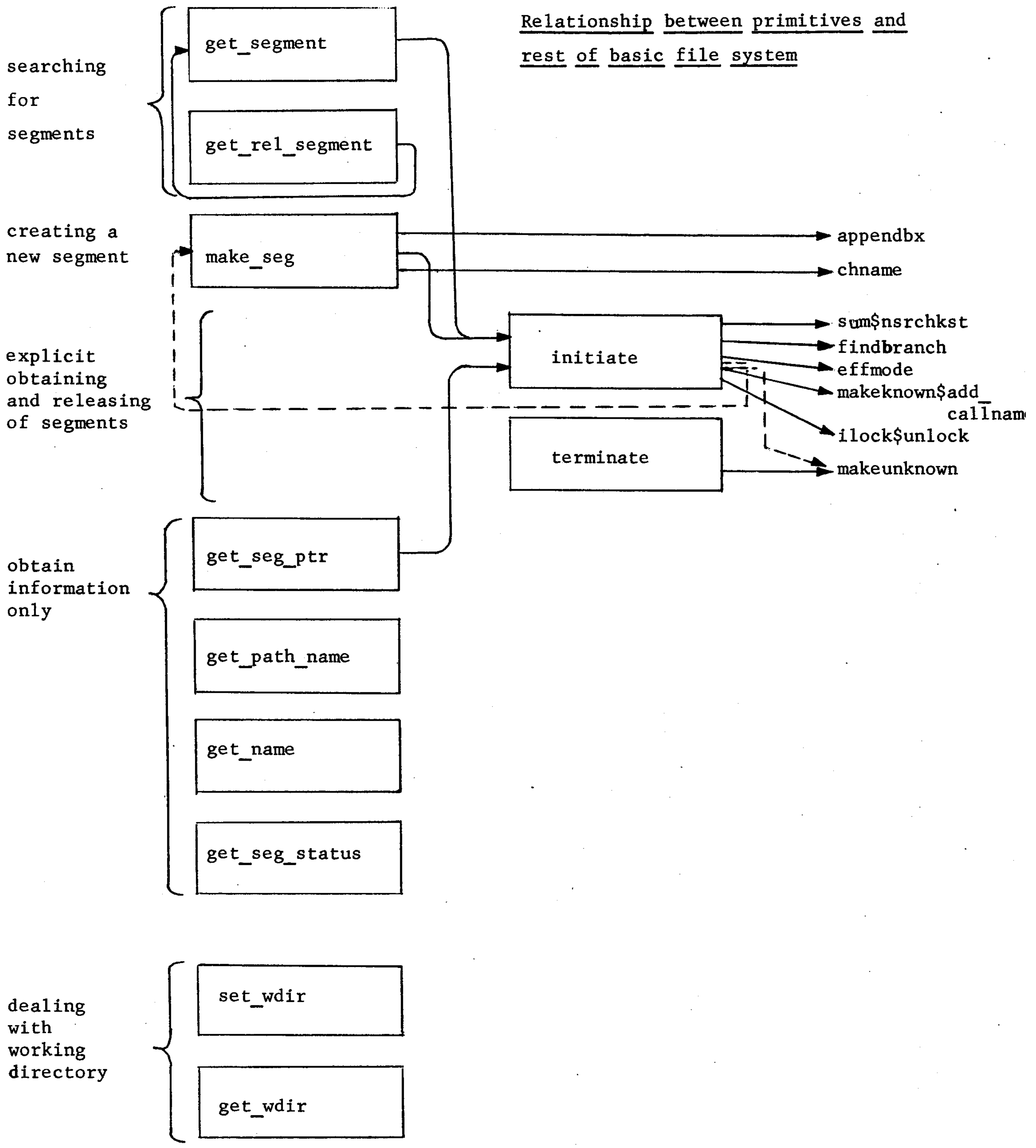
Provision is made in the entries which initiate a segment for creating and initiating a copy of the segment instead of the original to enable simple establishment of a modifiable segment which does not disturb the original.

The entries available are as follows:

1)	initiate:	initiates a reference name for a particular segment
2)	terminate:	terminates a particular segment
3)	get_segment:	searches for and initiates a segment corresponding to a reference name
4)	get_rel_segment:	find the name of an initiated segment, concatenates a suffix to the name and then get_segment's this constructed reference name
5)	make_seg:	appends a branch in the specified directory and then initiates the new segment
6)	get_seg_ptr:	returns a pointer to a segment for which the given reference name has been initiated
7)	get_path_name:	returns the pathname for a given segment
8)	get_name:	returns one of the reference names for a given segment
9)	get_seg_status:	returns information from the KST for given segment

10)	set_wdir:	enters the name of the working directory in the KST
11)	get_wdir:	returns the name of the current working directory

Those entries which initiate a segment return a null pointer if they encounter any errors. The other entries return an error code.


Relationship between primitives and rest of basic file system
searching for segments
get_segment
get_rel_segment
creating a new segment
make_seg
explicit obtaining and releasing of segments
initiate
terminate
obtain information only
get_seg_ptr
get_path_name
get_name
get_seg_status
dealing with working directory
set_wdir
get_wdir
appendbx
chname
sum$nsrchkst
findbranch
effmode
makeknown$add_callnam
ilock$unlock
makeunknown