TO: ✓J.H. Saltzer

R. Daley

R. Rappaport

R. Graham

K. Martin

FROM: Carla Marceau

SUBJ: Save and Resume

DATE: July 14, 1967

Attached is a list of modifications and restrictions which appear to be necessary in a certain implementation of save. Do you have any additions to the list? Does it horrify you?

Purpose

It has been suggested that when process A is saved, and then later resumed, the "resumed process" have a distinct process ID from process A. This implies that:

1) no process has an indefinite life span (unless the system stays up indefinitely);

2) two "versions" of a process can exist concurrently. E.g. suppose a user saved the state of process A at some time, then continued to run process A. At a later time the user can resume the saved process without having to destroy A. (This also assumes that process A can continue to execute after its state is saved.)

There may be other implications of distinct process id's for new processes. In any case, it turns out that certain modifications to current Multics modules and certain conventions are necessary to implement the above suggestions.

List

1. Any module which squirrels away the process id of the process must be modified so that it can cope with a change in process id's. Since process directories are named after process id's, the same goes for process directory names. The Segment Management Module (SMM) currently tucks away the path name of every segment initiated in the process, including segments which reside in the process directory. It would be handy if the SMM kept the path name of the process directory in a single place and simply referenced it elsewhere.

2. The above is useful only if we establish the convention that the user does not know about process directories, i.e., does not call pdir himself. He merely refers to certain segments as temporary, and the SMM decides to put them in the process directory. The user need not be prevented from referring directly to his process directory, but he must take the consequences of such a reference.

3. Interprocess communication (in ring 1) tucks away process id's. At the time of a save these process id's would have to be replaced by symbolic process names. In this regard, it might be handy to establish the convention that in inter-process-group communication working processes of a user process-group communicate with Overseer processes exclusively. I.e., a non-Overseer process knows processes in other groups only by process-group name.

4. Process id's would thus be known only in rings 0 and 1. Ring 0 is not a problem, because processes must be run out of ring 0 before saving. Ring 1 is a problem. Consider the following race: the SMM receives a request for a pointer to a temporary segment (i.e., a segment in the process directory). The SMM calls the pdir procedure and gets a pathname for the segment. Now the process is interrupted and saved, then resumed with a brand new process id and discovers that no segment exists for the pathname it had constructed using the old process id. (The SMM is not the only module where this problem can occur.) To avoid the problem we can

   a) run the process out of ring 1 before saving (gag!); or

   b) force knowledge of process id's to be restricted to ring 0 (fascinating implications); or

   c) raise a don't-save flag at crucial times (then run process

until flag comes down before saving); or

d) give up; or

e) none of the above.