DRAFT FOR DISCUSSION: 3/12/68

Identification

A Proposal to Reduce the Number of Processes in Multics

S. Feldman

Introduction

At present, it appears that it is going to be rather expensive to create and load processes in Multics. The following is a proposal to drastically decrease the number of processes needed for the system. This is done by coalescing the Answering Service, User Control processes, and the Overseer processes. Since typewriter I/O will usually be done by a Universal Device Manager Process, this change implies that a user not in quit state will have only a single personal process.

The Present

The following is a very sketchy description of the life of an interactive process group. Initially, the typewriter channel on which the user will login is inactive and in the hands of the Answering Service. When a dial-in interrupt is received, the Device Control Module in the Device Manager signals an event. In response to the signal, the Answering Service creates a new process called the User Control Process for the prospective user. The Answering Service then turns the typewriter over to the new process by detaching it.

The User Control Process then goes through the login sequence. Assuming the login is valid and Load Control allows the user on the system, User Control accepts the user. An Overseer process is created in the new process group. After setting up the Overseer, User Control waits for an event to be signaled.

Meanwhile, the Overseer initializes its tables, initializes the I/O system for the group, and attaches the console typewriter. The Overseer makes a call to tell the I/O system that the console is the command source, and creates event channels to be signaled if a quit or hangup occurs on that device. The Overseer creates the first working process for the user, which starts out in the appropriate login responder. The Overseer also an event channel to be signaled by the working process if it wishes to be destroyed. The Overseer than waits for certain events to be signaled.

If a quit occurs, the Overseer is signaled. In response, the Overseer quits the working process(es) in the group and creates a new process. All I/O is stopped, and then the new process is started up and reads the console typewriter to find what is to be done about the quit. The Overseer may be told to put the quitted process in hold state, to destroy it, to leave it in quitted state, or to let it resume computation.

If the user decides to log out, a signal will be sent to his Overseer. The Overseer then destroys the working processes in the group and signals the User Control Process for the group. User Control destroys the Overseer and signals the Answering

Service. The Answering Service then takes control of the typewriter channel and destroys the User Control process. The cycle is now complete.

The Proposal

Since the Answering Service, User Control processes, and Overseer processes are blocked most of the time, there will be little loss in generality if we combine them. Also, these three types of processes are active at different, but related, times. Part of the sequences of process creation and event signaling would be replaced in this implementation by procedure calls. Thus, after the signal signifying a completed dial-in is received, the Answering Service would call the User Control part of this procedure, which would then attempt to log in the user. If successful, the console would be detached and the Overseer part of the process would be called. This procedure would then create the new process group and the first working process in it. This working process would start out by executing a procedure (in the administrative ring) that initialized the I/O system and attached the command typewriter. After making calls to force an event to be signaled when a quit or hangup occurs, the procedure calls out to the login responder in the user's ring.

The process would be driven using the event call mechanism. Whenever an event is signaled, the Wait Coordinator makes a call to the associated procedure using a pre-stored pointer as argument. This pointer will probably point to the entry in a data base corresponding to the typewriter channel. The

procedures called would be the dial-in entry point of the Answering Service, the logout entry point of User Contro, the quit and hangup entry points of the Overseer, the event entry point of io_control, and possibly some special system procedure entry points.

When a quit is signaled, the Overseer part of the special process would create a new process in the relevant process group and start is out in a procedure that would divert the command typewriter (by a call to io_control) and call the proper quit responder. The quit responder would then make the appropriate signal back to the Overseer depending on the action to be taken.

When a hangup is signaled, the Overseer part of the special process would save all of the processes in the group (unless the process was supposed to be scrapped upon automatic logout) and then return the typewriter channel to the Answering Service. In case of a system shutdown, User Control would first quit the processes in the group. It would then make a special call to force the Dispatcher in the Universal Device Manager Process to force it to quit and then detach the console from the present user. The typewriter channel would then be returned to the Answering Service, which would probably hang up the typewriter and shut down the channel.

In order to increase utilization of system resources at times of high login and/or logout rates, it would probably be a good idea to have several special processes such as the above, each handling a different set of typewriter channels. This does not

modify in any way the above proposal.

Data Bases

Most of the code in the relevant modules could be saved in this change. An extra data base would be needed. This data base would contain an entry for each typewriter channel under the control of the process. Each entry would contain the name of the channel (resource name), the user id of the assigned user (blank if the Answering Service), the status of the line (waiting for dial-in, logging in, not functioning, etc.), and pointers to various per-process-group data bases. These data bases could have the same form as they did before the change, and the relevant modules would just have to fetch the pointer to the data base in a different fashion.

Assumes access to all such data bases, requires access lock of at least 2 users on all answering data bases.

Conclusion

The proposed change would reduce the number of processes in Multics by a factor between two and three. Considerable time would be saved in process creation and in process loading by coalescing all of the Overseer and User Control processes and the Answering Service process into a single special process. A great deal of inter-process communication could be transformed into simple procedure calls. Also, the reduction in the number of Overseer and User Control processes would allow all of these processes to be kept loaded. This would improve system response time for logins, logouts quits, and hangups. Most of the present code could be salvaged with few non-mechanical changes.

Detailed Description of Effects on the I/O System

Almost all of the present I/O System would be unaffected by this change. The only module significantly affected would be io_control (BF.3.01), whose main task is communication between the Overseer and the I/O System. The I/O end of the module would suffer little change, but the initialization procedure and the procedure that catches quits and hangups would have to be modified. Some of the work would be done in the new Overseer procedure, and the rest would be done during initialization of the new working process created in response to the quit.

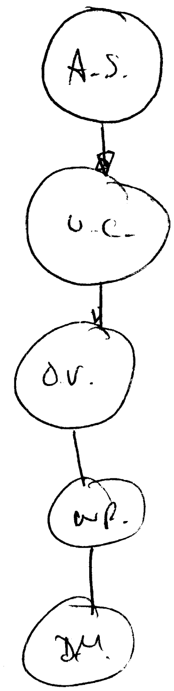
1. User Control was distinct from Answering Service, because and "per group"
Login is a interactive command. Simultaneous response would require some (maybe lots) recoding of login command to allow more logins while awaiting password.

2. Overseer was distinct from U.C, A.S group because it operates with a different access control, none than does the U.C.

3. Program requires creation of working process, etc., across an accounting and protection boundary, rather than within, for quits, etc. May make save/resume very difficult.

4. Accounting for the absent process?

5. Possible interaction with Load Control + Absentee job overseer arrangement.



better	~~AT minimum~~ minimum
1/system	← 1/system, unloaded except during dialup, hangup
1/processor	← 1/system, unloaded except during login/out.
?	← 1/system, always loaded to provide response for quits.
—	← 1/group.
1/typewriter type	← 1/system, always loaded