Published: 02/09/67

## Identification

Overview of System Control
J. H. Saltzer

## Purpose

"System Control"is the name applied to a group of modules and processes which control the operation of and access to Multics. The "System Control Procedure" is a module called by the Multics Trigger (see BL.0 and BL.3) when system initialization is completed. With this call, all essential features of Multics are capable of operation; Multics is now a free-standing system. Included in the category of System Control, in addition to the System Control Procedure, are the system processes (e.g. Answering Service Process, Drum Manager Process, etc.,) the Overseer, Logging in and out and Quit procedures which operate in a user's process-group, and the system load control procedures. This section describes briefly the purpose of and relations between these modules. Detailed descriptions are the purpose of other sections of BQ.

## Process-Groups

A single process in Multics is capable of only serial operation on several tasks. Since many useful computations require (at least conceptually) some parallel operations, processes are organized into groups known as "process-groups". A process-group is identified as a unit by a name, and that name is used to establish access control to the information storage hierarchy for all processes of the process-group. It is of course possible for a process-group to consist of only one process.

Typically, however, an interactive console user is assigned not one, but three processes to manage his interface with the system. One of these processes, the "Overseer" process performs the important functions of logging in and out, and "quit" handling. A second process, the "device manager", manages his typewriter console to permit type-ahead on input and buffered write-behind on output. The third process, the "working" process, performs the computations implied by the commands which the user types. The user may himself add additional working processes to the process-group to perform his own parallel computations.

Although the user can determine completely the course which his working process takes, he has no alternative but to accept the procedures provided by the system in the Overseer process. Thus it is guaranteed that the user can log in and out reliably and that his "quit" button will always work.

The name of a process-group is built up of three parts: a person identification, a project identification, and an instance identification. The project identification is the name of the project for which the user is working while logged in this time. The person identification is the name of the person who has logged in. The instance identification is two-character string which is distinct for each instance of this person-project combination which is logged in simultaneously. Thus it is possible for a user to log in simultaneously from different consoles (if he is permitted to do so) and remain an identifiable, named, entity. A complete discussion of process-groups appears in Section BQ.3.

System process-groups.

We have described so far a user process-group. There may also be system process-groups. A system process group is not associated with any particular person, but rather with the operation of the system itself. A typical example of a system process-group is the Secondary Storage Backup process-group, which copies newly created files onto magnetic tape for backup in case of failure of a secondary storage device.

Although conceivably one could have only one system process-group, containing all system processes, it is useful to have several system process-groups, since each process-group has a distinct name for access control limitation. By limiting access of a system process-group to files it needs, opportunities for catastrophe are lessened. This is an important consideration, since system process-groups are typically highly privileged and capable of causing disasters with system-wide implications.

A system process-group may be easily identified as such by inspection of its name. Since there is no person associated with a system process-group the "person identification" part of its name is null.

Section BQ.1.02 contains a census of system process-groups.

System Control Procedure

When the Trigger completes opration, it has become the first process in the system. (In the course of initializing the file system, it may have also created some additional system processes.) The Trigger then performs the call

```
        call    <Multics>|[system_control]
```

As this call transfers control from the trigger procedure to the system control procedure, we say that the trigger process has evolved into the System Control Process, and Multics is now "in operation". Although a user cannot dial up and log in yet, all essential features of Multics are operating, including all of the supervisor residing in the hard core ring--the File System, the Traffic Controller, and the GIOC Interface Module.

The System Control Process first establishes itself as an overseer process for a special system process-group, the System Control process-group. It gives itself the process-group identification name ".system.aa" for purposes of establishing file access control and communication with other system and user process-groups yet to be established.

The System Control procedure then creates the other members of the System Control process-group by using the standard Multics process creation mechanism. The System Control Process Group then stands ready to accept orders from the system operator, as described below. These orders can include directions to create other system process-groups, or to allow the Answering Service (see below) to let regular users into the system. The System Control Procedure is documented in detail in section BQ.1; other members of System Control process-group are listed in BQ.1.02.

The Answering Service Process

One of the system processes created by the System Control Process is named the Answering Service Process. This process has responsibility for monitoring all GIOC channels from which requests to log in may be accepted. When, for example, a potential user of the system "dials up" the computer through a dataphone dataset, the Answering Service Process is the one which responds to the resulting system interrupt.

When it establishes that a user is attempting to connect himself to the system, the Answering Service Process creates a user-process-group, with identification name "unknown.unknown.##", where "##" is a serial number which is distinctly assigned to each unidentified user in the system. The Answering Service creates an Overseer process for the fledgling user-process-group, and calls the Overseer procedure in that process, giving as an argument the device name of the particular data set which received the dial-up. The overseer process then takes over the responsibility of identifying the user and logging him in. The Answering Service Process returns to its vigil, monitoring all GIOC channels not currently in use and which may generate requests to log in.

The Answering Service is described in full in section BQ.2.

## Operator Control of System Process

During initialization of Multics, the operator controls the course of action taken by the General Loader and Trigger procedures from his typewriter console by commands which, while operating outside the usual command structure, appear to be normal system commands. When the Trigger passes control to the System Control Procedure, the operator's typewriter is temporarily disabled, while the System Control Procedure initializes itself and creates the Answering Service Process. The first operation performed by the Answering Service Process is to create an unidentified user process-group specifically for the typewriter console that was used during initialization. The system operator then logs in at that console. After he logs in, then, the system operator appears to be a standard user process-group, with the exception that his is the only user process-group from which the System Control Procedure will accept instructions. (The System Control Procedure discriminates against other user process-groups by making its inter-user communication data bases accessible only to the system operator's process-group.)

The general pattern of communication between the operator and the system, then, is as follows: The operator types some command to his working process, for example a command implying that all user process-groups except the operator should be logged out. The command operates by placing a request in the System Control Procedure's work queue and waking up the System Control Process (using standard inter-process communication calls.) When the System Control Process wakes up, it performs appropriate actions...in this case notifying the Answering Service not to allow more dial-ups and triggering the Automatic Logout mechanism.

By using the pattern described above, it is not necessary that that the operator who brought up the system remain logged in as long as the system stays in operation. Instead, when the time comes for a change in operators, the new operator dials up and logs himself in on another typewriter console. Then, the old operator types a command which passes the torch to the new operator by informing the System Control Process of the new operator's identification. The System Control Procedure readjusts the access to its work queue, and waits for further instructions from the new operator.

The System Control Procedure, of course, carefully validates the attempt to change operators, by checking to see that the identification of the new operator is an allowed operator's identification and that the new operator is indeed logged in. A permanent record of the transaction is made in the system log.

## Other Operators

So far, we have only discussed the role of a single operator, the one in command of the entire system. In any but the smallest systems, it is likely that there are in fact several operators. One of these is the "System Operator" who is in command of the system by virtue of being able to direct the actions of the System Control Process. In addition there may be operators responsible for mounting tapes, running printers and card readers, or perhaps moving printer forms from the stockroom to the computer room. When Multics is initialized, it is assumed that the System Operator has all of these responsibilities. This assumption is applied by placing the identification of the System Operator as the "User process-group to take orders from;" in each of the system process-groups associated with operations tasks. As the other operators dial up and log in, the System Operator can split off selected responsibilities by typing appropriate commands. A complete discussion of the interactions of the operators with the system will be found in section BM., System Operation.

## Limitation of Access to Multics

The ability to use the computation facilities of a Multics system is limited at two levels. At the first level, the user-in procedure demands identification of the potential user as one of the permitted users of the system. This level of access limitation is fundamental in order to establish accounting for usage of system resources. At the second level, even though the user is satisfactorily identified it may not be practical to allow him to log in because the system load is too great and his resource usage would degrade the response of other interactive users unacceptably.

To this end, the user-in procedure, after establishing the identity of the prospective user, inquires of the "Load Control" procedures whether or not the user should be permitted access. To allow for the possibility that this prospective user is in some sense "more important" than some user already logged in, a module known as the process-group Ranker.maintains a list of all logged-in process-groups. This list is in order according to the desirability of "bumping" each process-group off the system to make room for a higher process group.

Using these same tools, it is also possible for a Load Controller system process to dynamically adjust the system load in response to observed over- or under-utilization of system resources.

Logging in and the user_in procedure are described in BQ.3.03, and the complete load-control mechanism is described in section BQ.5.