Published: 02/27/69

Identification

System performance testing and certification
multics_test
R. J. Feiertag

Purpose

This procedure is used for making performance tests on Multics and for standard system certification. It creates a number of processes and has each process perform a set of commands specified in a file.

Usage

This procedure can be called from command level at two points:

multics_test -printer-

This entry point initializes metering to certain default values which are then printed in the console. If different metering parameters are desired they may be changed by calls to the metering commands after the call to multics_test. printer is the device name (i.e., prta40, prta34, prtb40, prtb34) of the on-line printer to be used for printing results. The default value for printer is prtb34.

multics_test$start runname number -typewriter1- ...

This entry point creates the number of processes indicated by number. Each new process is assigned a number sequentially beginning with 1. Call this number n. Each process then creates a working directory in >user_dir_dir with name runname_n. The output stream is attached to the corresponding typewriter designated by typewritern. If typewritern is "" or there is no corresponding typewritern for this process then the output stream is attached to a file with name runname_n in the user's working directory. The input stream is then attached to a file runname_n.script in the user's working directory. If this file cannot be found then the input stream is attached to runname.script in the user's working directory. This latter file must exist.

Important: The final line in these script files must be "multics_test$stop".

The user should dial up the typewriters which will be listed on the console. As each typewriter is activated it will type `ready'. When all the typewriters type `ready' the user should type a carriage return on his console. The script files will then be executed by the processes. When all the processes have stopped the output from those processes not attached to typewriters will be printed on the on-line printer along with the result of metering.

Metering is on during the creation of the new processes and during the running of the scripts. The results of these metering intervals are printed separately.

Example

Assume the user's directory is >user_dir_dir>mine>

Type the command:

```
multics_test prta40
```

This initializes metering and attaches the on-line printer.

Now type:

```
multics_test$start typical 2 tty194
```

Two new processes are created. The first will have working directory >user_dir_dir>typical_1> and its output will be attached to the typewriter designated tty194. If >user_dir_dir>mine>typical_1.script exists, the input will be taken from it and if not the input will be taken from >user_dir_dir>mine>typical.script. The second process will have working directory >user_dir_dir>typical_2 and its output will be attached to >user_dir_dir>mine>typical_2. The input will be attached to >user_dir_dir>mine>typical_2.script or if this does not exist >user_dir_dir>mine>typical.script. When the processes have completed running their scripts then >user_dir_dir>mine>typical_2 will be printed on the on-line console along with the results of metering and all the processes and the files they have created will be destroyed.

Implementation

The call to multics_test first attaches to the appropriate printer. It then makes calls to meter_set to set the metering parameters.

The call to multics_test$start first creates an event channel for future use. Metering is started and create_proc (BJ.8.01) is called for each new process. Each process is passed certain information which it needs via the pit, such as the process id and channel name of the user, the run name, the number of this process, and the working directory of the user. After the processes are created the procedure waits for wakeup from each process which indicates that the process has been initialized. Metering is stopped and meter_interpret is called. If typewriters have been attached then a message is printed on the console and a carriage return is waited for indicating that all the typewriters are ready. Then metering is restarted and each process is reawakened. The user process then waits until it has received a wakeup from each new process. meter_stop and meter_interpret are again called to obtain metering results. Then the output files and results of metering are printed out and the processes and all their associated segments are destroyed.

When each new process has been initialized it calls multics_test$cert. This procedure creates the working directory and attaches the process' output to either a typewriter or a segment and attaches the input to a script segment. An event channel is created and the user's process is awakened and is passed the event channel name and process id of this new process. This new process then goes blocked and when reawakened calls tester to act as a listener. When multics_test$stop is called by the script it detaches the input-output devices and blocks this process.