To: Distribution

From: J. C. Whitmore

Date: 15 Nov 76

Subject: I/O Daemon Usage Statistics

INTRODUCTION

A few months ago, some temporary meters were installed on the MIT Multics system to gather statistics for comparing the the new and old charging algorithms. The meters were designed quickly, without much thought given to general application. However, the system administrator found the usage statistics very helpful and requested that a permanent metering capability be added to the I/O Daemon.

This MTB proposes a more complete design for gathering I/O Daemon usage statistics.

OBJECTIVES

The I/O Daemon usage statistics are intended to be used by a system administrator to evaluate how users are distributing their requests among the various request types and queues. This data is not highly dynamic (as are the file system meters) and therefore should be collected over a long period of time, e.g., a week or a month.

The usage statistics will also provide information which is not currently available from the accounting system. The contribution of each request type to the total I/O Daemon revenue is a primary example. Other data could be easily added as its need is identified.

The maintenance of usage statistics should be a site option. There will, of course, be some overhead associated with the metering code. Therefore, the system administrator should be able to decide whether the additional overhead is justified for his site.

I/O Daemon drivers may run at various authorizations and with several different process group ids. Therefore, the statistics for each request type should be split into separate segments for access control purposes.

THE PROPOSED STATISTICS TO BE GATHERED

The usage statistics will be maintained per request type (RQT) and device class (DVC). Some of the data will be global to the RQT/DVC, while other data will be associated with each queue in the RQT/DVC. The per queue data will exclude restarted and continued requests.

Global Data per RQT/DVC

Start of meter period
Time last updated
Number of restarted requests
Number of continued requests
Number of requests not charged
Number of killed requests
Number of cancelled requests
Number of requests not processed
Number of special VFU formats loaded

Per Queue Data for a RQT/DVC

Number of requests
Number of copies
Number of pages (cards for punch)
Number of lines (blocks for cards)
Total charges
Time last serviced
Real time doing requests
CPU time doing requests

IMPLEMENTATION

The usage statistics will be maintained in segments which are contained in the directory,

```
>ddd>iod>meters
```

Each segment will correspond to a RQT/DVC and will follow the naming convention:

```
<RQT>[.<DVC>].meters
```

The ".<DVC>" component will be omitted if there are no device classes defined for <RQT> in the iod_tables.

The decision for a driver to maintain meter data will be based on the appearance of the "meter= on" string for the args keyword of the RQT in the iod_tables. All device classes will be metered if the RQT is to be metered.

When metering is required, during driver initialization (in the program iodd_), the driver will attampt to initiate the meter segment for its RQT/DVC. If this is successful, and the driver has read/write access, a pointer to the segment will be stored in iodd_static to make it available to all driver programs. Otherwise, an error message will be printed and the driver will fail to initialize the RQT. When metering is not required for the RQT, a null piorter will be stored in iodd_static. For efficiency, a 1 bit switch will also be stored in iodd_static to control conditional execution of the metering code.

Most of the metering code will be added to the program output_request_. The data to be collected already exists in program variables, so the overhead of keeping the statistics should be small.

Each meter segment will have a lock to coordinate multiole drivers. If problems occur while attempting to save the data, the driver will terminate its metering (with an error message) and continue to process requests.

PRINTING THE USAGE STATISTICS

A new command, display_iod_meters (diodm), will be available to allow any process, which has read access to the meter data segments, to print the usage statistics.

```
Usage:  diodm <RQT>[.<DVC>][.meters]
```

The command will format the data and provide summaries of the per queue data. The information will be written on the user_output switch.

Notes:

1. There are no control arguments planned at this time.

2. Only non-zero data will be formatted.

3. The command will not attempt to lock the data segment or wait on the lock.

4. The command assumes that the data segment can be found in

   ```
   >ddd>idd>meters
   ```

   For testing, this assumption can be changed by the entry diodm$test.

ACCESS CONTROL

Access to the directory >ddd>idd>meters will be sma to the system administrator and s *.*.*. Only the system administrator needs to create, delete, or set access on the data segments. The directory access class must be system_low.

Access to the meter data segments will be rw to the system administrator and the driver process, and will be r *.*.*. The usage data is not inherently sensitive and may be read by all users. Only the system administrator needs to truncate (reset) the data.

Since the access class of the directory >ddd>idd>meters is system_low, the access class of all segments in the directory must also be system_low to be consistent with AIM. However, if drivers of higher authorization store their usage data in these segments (using system_privilege_$initiate), a write down path (but a noisy one) exists. The administrator who wishes to eliminate this write down path, and still get the usage data, can create subdirectories of the proper access class which contain the data segments and place links to these segments in the normal "meters" directory.

ADMINISTRATION

It is expected that the decision to meter a given RQT/DVC will not be very dynamic. Therefore, standard file system commands will be used to create, delete, truncate (reset), and set access to the meter data segments.

Care should be taken to ensure that no driver is using a meter data segment when deleting the segment. Otherwise, a seg_fault_error will occur in the driver process and the current request may be lost (or worse?).

Truncating the data segment will reset the usage data and should also be done when no drivers are using the segment. Otherwise the data could become inconsistent and the start time will be inaccurate (1 Jan 1901?).

FUTURE EXTENSIONS

If the usage statistics prove to be as valuable as anticipated, some additional data may be requested and some better commands to interface with and control the metering may be desirable. Some examples of better commands are:

A create_iod_meters command could be provided which would look in the iod_tables and create segments, in the correct directory, for each device class in each request type with "meter= on" in its args. It would automatically set rw access to the driver and r access to all users.

A reset_iod_meters command could honor the data segment lock when reseting the usage data and reinitializing the start time.