First Draft

IPC Users Committee Report on Multics Reliability

William D. Mathews
November 17, 1970

Introduction

On October 27, 1970 the IPC Users Committee discussed the topic of System Assurance on MULTICS. It was particularly timely that this topic should be considered since the system, after nearly five years of development, has now been available to the public for one full year. As a design concept, MULTICS is the most complicated operating system ever developed and it aims at solutions to some very critical problems in the distribution of computational resources. It seems a fair speculation that a more comprehensive approach to time-sharing will not be implemented before the 1980<s. As a user<s facility, however, MULTICS has not yet reached the maturity and stability demanded by a substantial portion of its customers. It is our purpose here to outline some of the factors users see as having significant bearing on the question of system assurance and to give emphasis to opinions and suggestions which will assist in improving the level of user satisfaction with MULTICS.

Acknowledgements

We wish to thank John McManus and Roger Roach of the System Assurance Staff of the IPC for taking time from their

busy schedules to discuss this topic with us at our meeting. Thanks are also due to Tom van Vleck, Jerry Saltzer, and Akira Sekino for supplying additional information and statistics. Tireless analysis and tabulation of system crashes was performed by Kathy Huber of Project TIP.

MULTICS Growth

Performance statistics gathered during the past year indicate that MULTICS capability has steadily grown. In October 1969 the system was saturated when 20 users were logged-in, in October 1970 forty users could log-in before saturation. With this increased capacity has come about a corresponding decrease in cost to the user. Table 1 summarizes the results of running a sample user script consisting of 66 commands.

Date	System	Users	CPU Secs
09/03/69	3.8	18	67.810
01/26/70	5.5	21	50.136
03/25/70	6.0	28	48.839
07/21/70	8.7	35	30.604

Table 1. MULTICS Performance Analysis summary for a simulated script of 66 commands.

It can be argued that the improvement in the system as far as the user is concerned is even greater than the factor-of-two indicated in this table since the script involved uses the FORTRAN compiler which has not been subjected to any substantial improvements during the year while the average user utilizes programs written in PL/1.

PL/1 has been dramatically improved during the year and these improvements are noticed by the user both in the speed of compilation and the efficiency of the compiled code. Actual figures demonstrating this improvement in PL/1 could not be found at the time of this writing.

Another measure of the growth in service capability can be found in the summary of average number of processes initialized. Table 2 gives this information for weekdays and weekends in April and October 1970.

TYPICAL MULTICS USER LOAD

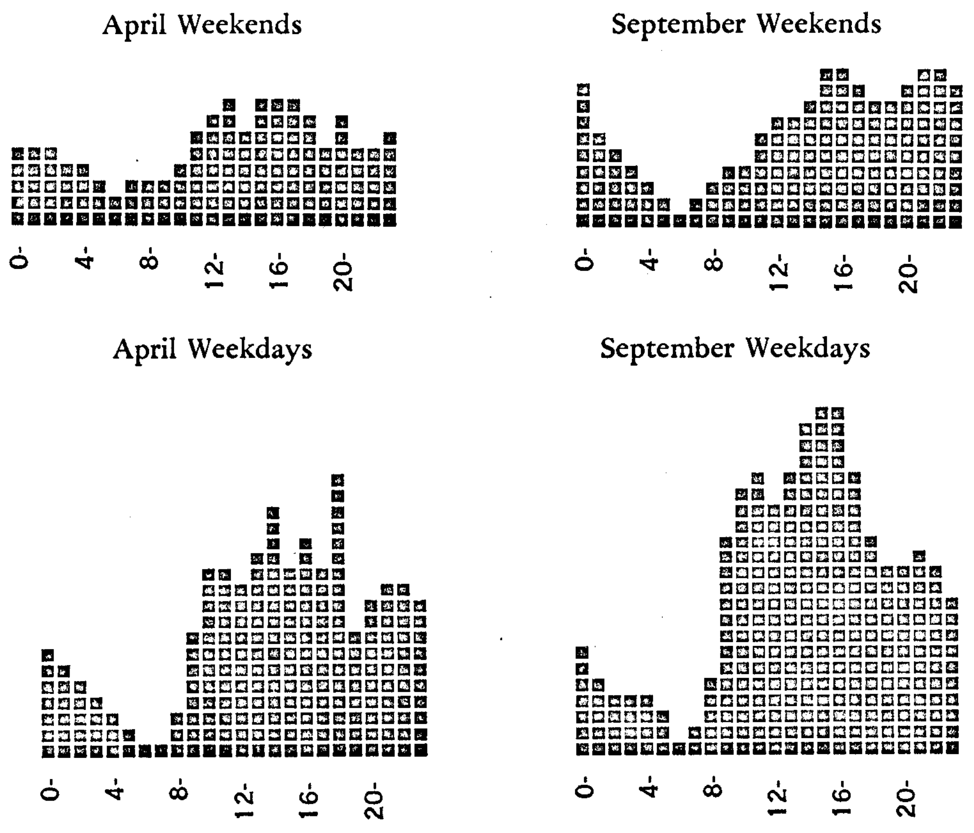


TABLE 2 GOES HERE

The User<s view of System Assurance

That a system is up during advertised hours is only the most primitive form of system assurance. Other forms of assurance that the user looks for are the usability of the system, the repeatability of strange or buglike situations, the consistency of basic utilities such as typewriter I/O, printing, card to disk or tape to disk activities. Positive or negative feelings about a system begin with the act of dialing the computer. All these functions have thresholds of acceptability. Listed roughly in order of importance, the user seeks assurance that:

1. The system is operating during scheduled times.
2. The user can log-in.
3. The basic services are not malfunctioning.
4. The system works as advertised in the documentation.
5. Auxiliary support systems are operating reliably.
6. Consultation is available.

These topics will be treated in order in the following discussion.

System Operation during Scheduled Times

>>Crashes<< frequently prevent the system from operating during scheduled hours. It is to the credit of Project MAC and IPC staffs that dilligent information has been maintained on each crash and its probable cause. This

information has been kept in a most professional manner. It was also recognized by the staffs of IPC and MAC that the incidence of system crashes was far beyond the expectations of the user community and far beyond the expectations of the designers as well. It should be pointed out that a crash on MULTICS invariably destroys the user<s process. This is especially serious for persons who are inputting files on-line and for persons debugging programs who have spent considerable time establishing the proper testing situation only to have their work completely destroyed. The impact of a crash at such a time is often psychologically severe. During the summer of 1970, goals were set for MULTICS operation in a memo by Jerry Grochow. In brief, these goals called for operation such that there would be:

a) An average of one crash per day due to software.

b) A maximum of three crashes per day due to software.

c) An average of one crash per day due to hardware.

d) A maximum of three crashes per day due to hardware.

e) A maximum of one disk reload per month.

f) System up 75% of its scheduled time.

Statement on Reliability Goals

Subsequently, a statement was made by Wes Burner of IPC on August 13, 1970 which read as follows:

Dear Multics User:

In early July we established a committee known as the Multics Planning Group that includes representatives of all organizations connected with the implementation of the Multics service. This group drew up a set of acceptable and attainable goals involving the reliability of Multics to be reached on or about September fifteenth. These goals encompass not only performance, but also increased stability of software and of operational procedures. Tasks were defined, personnel were assigned and measurement criteria established, all with the September milestone in mind. At that time, it was also agreed that, should the efforts of people dedicated to the project<s success not achieve the established reliability and performance standards, then users of the Multics service must be advised that there are reservations involved. In view of current progress, I see no reason to believe that there will be any qualifications necessary. We are well on target towards achieving our objectives.

W. J. Burner, Director

Information Processing Center

MULTICS System Reliability Index

Eventually, these goals were looked at in a different manner. A single number per week measure, known as the MULTICS System Reliability Index (MSRI) was computed for a period of eight weeks. This index had in its favor the fact that it took into account not only system crashes but also other failures in the system such as bugs in the standard service commands. On the other hand, no goal was set against which this index could be compared, the measure was only calculated for a short period of time, and finally, the measure did not definitively indicate any improvement in the system. Table 3 shows MSRI for the weeks it was computed. Low numbers indicate reliable operation.

Week	MSRI
1	1684
2	1765
3	1096
4	789
5	756
6	1817
7	1047
8	1033

Table 3. MSRI for the eight weeks starting 07/18/70.

Tabulation of Crashes

To gain more perspective on the situation, we have tabulated the information on crashes for a forty week period. During the forty week period ending November 7, 1970, there were 915 crashes. Multics crashes averaged 3.268 per day or 22.875 per week during this period. These crashes can be looked at in a number of ways. In the following tables, we look at them by four-week period to see if there is any trend, by month for those who think in terms of months, by day of the week, by hour of the day, and by type of failure. Table 4 shows the distribution of crashes by four week periods.

Period	Crashes	Crashes per day
1	75	2.679
2	72	2.571
3	119	4.250
4	95	3.393
5	122	4.357
6	98	3.500
7	95	3.393
8	71	2.536
9	88	3.143
10	80	2.857

Table 4. Distribution of the 915 crashes in the forty week period ending 11/07/70 tabulated by four-week periods.

By month, we can report the average number of crashes from February 1 to October 31. The figures, contained in Table 5, are only slightly different from those in Table 4.

Month	Crashes per Day
February	2.7
March	2.5
April	4.5
May	3.2
June	4.7
July	3.9
August	2.7
September	2.3
October	3.0

Table 5. Average number of system crashes per day by month.

Generally, system crashes are highly correlated with the number of users being served. Thus, weekends and early morning hours are relatively crash-free. Table 6 shows the distribution of crashes by day of the week.

Day	Total C<s	Avg. C<s
Sunday	73	1.825
Monday	135	3.375

Tuesday	165	4.125
Wednesday	142	3.550
Thursday	150	3.750
Friday	169	4.225
Saturday	81	2.025

Table 6. Multics crashes by day of the week for the forty week period ending 11/07/70.

Sundays are more than twice as crash-free as Fridays. A similar spread can be noticed when hour of crash is tabulated. A factor of three or more can be observed in the difference between the number of crashes in the late evening and at midday. Table 7 summarizes crashes for the 39 week period ending October 31 and gives the probability that there will be a crash within that time slot.

Hour	Crashes	Probability
0	30	.1099
1	21	.0769
2	22	.0806
4	8	.0293
5	8	.0293
6	7	.0256
7	21	.0769
8	35	.1282
9	35	.1282
9	61	.2234
10	72	.2637
11	55	.2015
12	66	.2418
13	63	.2308
14	82	.3004
15	55	.2015
16	49	.1795
17	37	.1355
18	48	.1758
19	26	.0952
20	27	.0989
21	48	.1758
22	30	.1099
23	23	.0842

Table 7. Crashes by hour in the 39 week period ending October 31, 1970 together with the probability that a crash occurred in a given hour.

Notice that a user working at 2 PM has a 30% chance that the system will crash under him within the hour. Compare this table with Table 2 which shows the typical user load.

Finally, distributed by cause of crash, we can see that >>hardware<< is a prime cause of the crashes, >>software<< second, then >>other<< which includes operator, engineer and installation caused crashes.

Time Period	HDW%	SFW%	OTH%
1	60	29	11
2	49	20	31
3	52	21	27
4	49	30	21
5	70	15	15
6	36	42	22
7	28	64	8
8	50	32	18
9	32	55	13
10	58	30	12

Table 8. Percentage of contribution to total crashes by hardware, software, and other causes for the ten four-week periods ending 11/07/70.

In summary, for this period, hardware accounted for 49% of the failures, software for 33% and other for 18%. Although the proportion of hardware failures has been decreasing recently and software failures increasing, the total down time due to software failure has been decreasing due to quicker recovery.

Crashes from the User<s Point of View

From the individual user<s point of view, the nuances about cause of crash, duration, frequency, etc are largely

lost. He is faced with the more primitive data of his senses. His dataphone which was alive and well has now turned itself off. His reaction is more likely to be emotional than statistical. A tabulation of total user sessions against crashed sessions shows that something on the order of one session in five meets with disaster.

Month	Total	Crashed	C/T
April	7653	2052	.268
May	7237	1465	.202
June	7924	2380	.300
July	7269	2511	.345
August	8161	1891	.232
September	7598	1205	.159

Table 9. Total sessions, crashed sessions, and the ratio c/t, for the six months ending September 1970.

General Remarks about Crashes

Some general remarks about system crashes are in order. First, both hardware and software crashes seem to occur in bunches. Some hardware problems occur several times before their cause can be found. Software bug flocks (swarms?) occur expecially after a new installation. John McManus speculates that there may be some cyclic behaviour but that the cycle may be too large to detect. Another point is that several of the bugs in the present system are thought to be load or timing dependent. Only during periods of heavy stress do these conditions bring the system down. Naturally, this is also the moment when most users are likely to be affected.

Personal Experience

Doug Miller has related his experience as part of a course which is using Multics during the fall term. He stated that some 35 or so persons were participating in the class until the first hands-on computer problem. The problem was judged by him to be a fairly minor one, one which could easily be completed in an evening or two. In fact, no one in the class was able to complete the work due to system crashes and he himself logged into the 360/67 after Multics gave him trouble for four days in a row. Only 12 persons remained in the course after this experience. Most of the loss, Miller felt, could be attributed to discouragement with the computational facility. It should be noted as a postscript that the week mentioned was unusually bad even for Multics, a new teletypewriter device interface module (TTY DIM) had just been installed and was on its shakedown cruise. Nevertheless, the experience warns us that instructors should be informed of the situation and their expectations should be tuned to the frustrations inherent in current Multics use.

Opinions and Suggestions

Although we are subjecting the data to further scrutiny, it is our impression at this time that no convincing trend has been established to form a basis for optimism. The opinions and suggestions of the users committee on this matter of system crashes follow:

1. The Ides of September.

Since September 15 has come and gone and no major improvement in Multics basic reliability has been achieved, the user community at large should now be informed that while gradual improvement will continue, no dramatic improvement is foreseen.

2. New Goals.

New Goals for reliability should be set and the users should be kept closely informed of progress. We suggest that such goals should not postulate herculean effort or boundless energies on the part of the development and maintenance staff but should instead attempt to progress by measured steps. It is our belief that the goals previously set were in fact unattainable in the given time frame.

3. New Measures.

We suggest that an important measure of the basic availability of Multics could be based on the number of console hours lost due to failure. Such a measure should also carry penalties for each time the system goes down. For example, if the system goes down with 20 users and stays down during a two hour period when there would normally be 22 and 24 users, the total loss should be something like 66 user hours.

4. Multics Badness Index.

Since bugs congregate in droves, and since if the system is bad it is likely to be very, very bad, the recorded message should indicate not only that the system is up or down but how likely it is that it will stay up once it comes up. A recent recording stated that >>Multics is currently down due to unknown causes and will be back at 1:30<<. In fact, this was the fifth time it had gone down that day. The user should be given some warning of this.

5. Future News.

Most of the users do not receive Multics Checkout Bulletins and, indeed, it is unlikely that they could make much sense of them in their current form. But there is the feeling that someone knowledgeable could (and should) capsulize the information contained therein and summarize future installations, their target dates, and the probable disruptive severity of such installations.

6. Special Test Sessions.

If some of the current bugs are load dependent, would it not be possible to have special test sessions in which no guarantees were made to the user but during which use of the computer is free? This might generate the desired load and some good will at the same time.

7. Classroom Use

A statement should be made at this time to guide instructors in the choice of system to support classroom work during the next semester. It is our recommendation that this statement should be based on current operations, not on hopes or lights at the end of the tunnel. We feel that classroom applications which do not specifically demand Multics use as part of the substance of the course should be encouraged to look elsewhere until the system settles down.

8. Maintenance Emphasis.

The current division of responsibility between system assurance and development might be profitably redrawn. It is our understanding that the IPC System Assurance staff detects bugs but does not set priorities on the fixing of those bugs. As the system becomes more and more of a service facility, the maintenance staff will undoubtedly have to be given more power to determine just which bugs must be fixed.

Available to the User

Even when the system is up, it is not always available in the user<s sense of the word. During the daytime, the system is often filled to capacity and a heavy load often obtains well into the evening. In line with this it should be pointed out that the Performance Analysis figures show that when the system is upgraded from 256K to 384K of core, the resulting increase in capacity is about equal to six months worth of tuning. That is, in January running under 384K was about equivalent to operation in July under 256K. In light of the feeling that the demand is now present on campus to saturate a 90 user machine, we wish to convey our urgent recommendation that steps be taken at once to bring about an increase in the size of available core.

Private or User Crash

In addition to gross malfunction which takes the system down, the individual user often finds himself in a situation in which, as far as he is concerned, the system is not providing him a real service. This happens sometimes through ignorance but at other times through bugs in the standard service system which are not fatal to Multics itself but possibly fatal or damaging to the user<s process. Since it is very difficult for the user to determine if a problem has been caused by him or he has been victimized by some unknown process, clues should be published which will allow users to determine the facts.

Standard Service System Problems

There is considerable anxiety in the user community about the compilers on Multics. PL/1 and FORTRAN, perhaps because of their complexity, continue to have bugs and to incorporate bugs into the users programs. This is particularly disquieting when a previously debugged module develops a bug because it was recompiled. When such bugs develop, the time frame of bug fixing is a major concern to the user. He does not wish to hear, for example, that >>EDM will be fixed in two weeks.<< When standard service system commands are not functioning this must be given highest notice such as a message of the day warning. If the bugs cannot be fixed, the command should be withdrawn from service.

Status of FORTRAN

The current status of FORTRAN on Multics has been called into question. It is the opinion of the users committee that unless the inadequacies of the current FORTRAN compiler can be overcome at once, it should be decommissioned as a standard service and users should be made properly aware of the difficulties of transferring FORTRAN coded subroutines onto Multics until a new compiler arrives.

Quality of Supporting Services

As of November 1, users are being charged for printouts on the on-line printer. While we applaud every attempt to make the system pay for itself, three objections are raised to this policy. (1) The cost of the printout is not given on the tailsheet. (2) The printout is of such unspeakably miserable quality that to charge for such a questionable service seems to the user to be encouraging the continuation of evil. (3) Satisfactory methods of obtaining a rerun or a rebate have not been publicized. We suggest that recovering the cost of printout should wait until the product is worthy of the name. The present course of action seems to aggravate a very sensitive spot.

Card Processing

Although it is very difficult to obtain good documentation on problems involving card input and output, the following statements have been made: (1) About 10% of the cards submitted to Multics fail to end up on the system (links are not properly established in the user<s directory, cards are mangled in the card reader, various sequence errors are complained of, the cards directory is full) (2) About 10% of the cards punched out from Multics cannot be loaded back on (blank cards are in the middle of the decks, parts of decks are missing, whole decks are missing, bad sequence numbers are generated). Some of these problems involve programs, others operations. This has serious implications for users wishing to transfer from CTSS to Multics since cards are the only common currency for transferring such data. We hope that this situation will soon improve.

Illustration
Crash Pattern by Hour during a 273 day period.

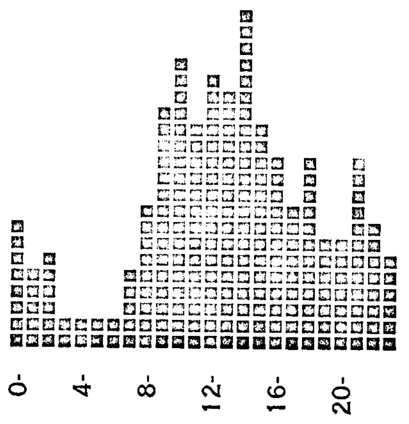
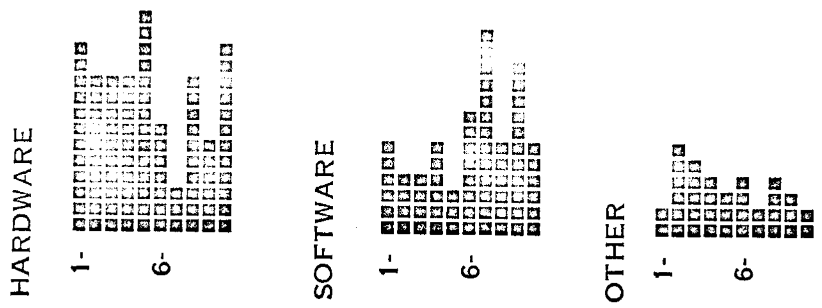


Illustration
Distribution of Crashes by Cause. Each square represents 5 percentage points.


HARDWARE
SOFTWARE
OTHER