ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC06

Time and Load Measuring in the SPS/LEP Control System

J. Navratil
Czech Technical University
Zikova 4, 166 35 Praha 6, Czechoslovakia

Abstract

This paper describes the experiences with the SPS/LEP Control System during its first operational days from the communication point of view. The results show difference between hardware possibility of the local communication based on the modern technology and the possibility to use it by PC machines. There is also several figures describing the activity on the communication lines.

I. INTRODUCTION

The control system of the SPS/LEP is based on distributed microcomputer network which contains more then 300 hosts and many thousands of different devices used for the control. Backbone is created by the communication system which can be divided into two parts. The horizontal plane is based on the Token Ring system to which are connected hosts used for control of particular processes on high level and many vertical planes provide connection onto the control devices (computers, interfaces, VME creates, etc.). MIL-1553 B is used here as the standard for the communications.

Such control system is very interesting first of all by its simple design where the total computer power is obtained by the relatively small quantities of hosts (IBM PC clones) and also for the sophisticated combination of a different communications devices (TDM, gateways, bridges, etc.) which ensure flexible services in the large area and long distances (27 km) [1][2].

My activity was concentrated on the Token Ring services and because my programs had to communicate through the whole network it was interesting to know the environment in which I had to work and its basic characteristics as the response time, speed of data transport and the load of the system (from the point of view of communication). Detail results are reported at CERN SL/CO/Note 90-05.

II. NETWORK ACTIVITY

During November and December 1989 when LEP started normal operation and all parts of the control system were working normally I collected some data about the activity of all hosts in the Token Ring communication network of the Control system. The aim of the measurement was to get a general overview of how much the Token Ring network was used by users for real control work.

For this purposes a set of programs were developed which are part of the Network Management System [2]. The results and evaluation of the activity in the network was used mainly to get an overall picture about the use of particular parts of the network and for long term planning.

For described experiment I have used three programs. The first of them collects data from the selected hosts and two others are used for processing and analysing the data and printing reports. The interface counters are used as the basic information. (The counters of the number of transmitted packets and bytes are a part of the standard communication software.) Data is collected by the RPC mechanism by the standard Rply_data program which is running on almost all hosts as a standard server. All the information from the selected ring can be collected in a few seconds.

The analysing program generate 3 tables. The first gives the total overview about the measured values, the second table gives an extract of the most active hosts and tries to express the values in a pseudo graphic form. The third table shows the activity in the time picture (the same data was collected in regular time intervals).

If we have a look on the tables in detail, then the first table contains 5 originally measured values (time in sec., number of input packets, number of output packets, number of input bytes and number of output bytes) for each host. All other values in the table are calculated from these values. As the best representation of the activity of each host I have used the percentage of the total activity in the network. These figures were calculated for each measured value (input packets, output packets, input bytes, output bytes). The last calculated value for each host is the "average speed", it is the activity of the communication interface - sum of the input and output bytes divided by measured time. On the bottom of the table are summary values and an "actual average speed" in the ring.

The second and third table are self-explanatory. An extract of the hosts in the second table is done on the basis of the minimum trash value. In our case the trash value was selected as 1.5%. From the ghraphic interpretation it is immediate to get a picture of the hosts activity (the most used one). The third table shows all hosts but for better reading of the information there are shown only relative values of these hosts in which the activity is higher than the trash value. This table gives a good overview about the "long term behaviour" or the "stability of needed service". (See examples shown in tables.)

III. MEASUREMENTS OF RESPONSE TIME

There are several tools for the network communication but I have used only two of them, FTP or (TFTP) and RPC (Remote Procedure Call). Both tools ar also heavily used in many others applications. My effort has been concentrated on the RPC implemented via Network compiler [5] from the point of view of the normal user. This is because this facility was the principal tool for many programs in the Network Management System which we are developing in our section and also because RPC is a very powerful tool in itself.

Some timing measurement were done in the past. For FTP and transport of short messages by the UDP facility (which artificially simulates the possibilities of real RPC [4]), the work was mainly concentrated on analysing the communications properties of different operating systems. Another study [6] measured response time of an implemented RPC, but

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC06

Table 1 An example of distribution of network load
(time interval 4450 sec, trach level 1.5 %,
40 machines in the PCR ring)

Input host	packets abs.	rel.[%]		Output packets abs.	rel.[%]	
aldev5	5431	12.14	XXXXX	5096	13.02	XXXXX
cons11	1568	3.51	X	1673	4.27	X
cons14	3107	6.95	XXX	2928	7.48	XXX
fspcr	22739	53.30	XXXXXXXXXXXXX	227239	58.11	XXXXXXXXXXXXXX
pipcr1	2192	4.9	XX	2033	5.19	XXX
rfsba3	2263	5.06	XXX	2060	5.26	XX

Table 2 An example of hosts activity
(scan interval 360 sec., trash level 1.0 %)

Scan interval/Activity [%]

host	1	2	3	4	5	6	7	8	9
aldev5	10.9	22.5	22.5	33.4	22.7	32.7	7.1	4.8	6.5
aldis1	.	.	.	.	.	.	.	.	.
aldis2	1.4	10.7	.	.	.	.	.	.	.
colsep	.	.	.	.	.	.	.	.	.
cons11	2.8	6.2	6.6	7.5	5.8	8.4	1.9	5.8	1.7
cons14	42.9	3.8	2.6	1.3	20.3	3.5	2.5	.	2.7
fspcr	35.0	29.6	47.8	35.6	28.2	31.8	83.6	54.2	84.8
pipcr1	4.0	9.8	10.1	10.6	9.9	13.7	2.5	2.3	2.5
rfsba3	.	.	.	.	.	.	.	26.2	.
...									

at the start of my work I didn't know about this paper. The results from this study are for similar conditions approximately the same, the main difference is in the method of measuring. While the measurements described in [6] took many hours, my method takes only a few seconds and the load of the measured object was minimal. It allows these measurements to be made practically on-line without interrupting normal operations in the ring or in the host. Another new effect which has not been mentioned in [6] is a problem with ring interconnections. This is mentioned in [4] but there are only figures for a one packet pass. From the point of view of a normal user of RPC this is very important because in normal circumstances the effect increases with each packet and it is also directly connected with the global topology of the network.

Random timing measurement have shown that the response time is sometimes quite high and more dependent on the hardware of the computers which are involved in the communication than on other known aspect such as position of the host in the ring, distance or speed of the transmission system ! The results of RPC response time from one host to several others in the network are shown in table 3.

Explaining this effect is rather simple. The token ring is based on the IBM TR system with frequency slightly modified to 4,225 MHz. And if we assume that the physical level of the TR system is capable of transferring data at about 4.0 Mbps [3] then the time used for the real physical transmission of data used in the RPC (< 1 kbyte) are in the microseconds range which is negligible compared with the time consumed by all communication software layers implemented under Unix (including the RPC layers).

Timing measurement were done by a trivial client program which was running in the selected computers and a standard program which was running everywhere as a permanent server.

The results from many measurements showed that the faster reply came if both source and destination were fast computers. The following measurements showed the interesting fact that the same result was possible to get if both the Client and the Server program run in the same computer. This allowed simplification of the whole measurement and to define a Round Trip Response Time - RTRT which could be taken as a

Table 3 RPC Response time from the PCR ring

from ring	(host)	ring	to (host)	response
pcr	aldev5	lma	lmagrlma	170 ms
pcr	aldev5	lma	bile28	190 ms
pcr	aldev5	lsv	lmgrlsv	170 ms
pcr	aldev5	lsv	colsr2	220 ms
pcr	aldev5	laa	lmgrlaa	170 ms
pcr	aldev5	laa	recpca	200 ms
pcr	aldev5	lbar	aldev4	210 ms
pcr	aldev5	ldev	olive	250 ms

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S08NC06

reference characteristic for a particular type of machine. The values include time for connection. This time was about 80 – 100 ms for the I80286 processors and 60 ms for the I80386 machines (Olivetti M380/C).

All previous measurements were taken in one simple ring. Working outside simple rings brings another aspect to be taken into account. The rings are linked together by bridges and gateways. These devices (driven by software) need some time for doing their job. It was surprising to find that the delay in the bridges is, from the point of view of this measurement, negligible but the delay caused by the gateways is rather high. The values for different gateways on the network were measured for our type of request as being between 15 to 50 ms (worse for 80286 – 45 ms and the best for CISCVO/DATATRAC – 15 ms). It is necessary to emphasize that these times represented an average case from a not very loaded gateway. These figures can be much worse and they will be quite heavily dependent on the traffic through a particular gateway. Knowing the characteristic of the particular constitutive elements of the system and also knowing the topology of the network we could relatively easily synthetically estimate the best response time for the different paths in the network.

Finally I would like to bring to attention several other important problems which are connected with the use of the UNIX operating system and which put a little more light on the results.

The first problem is the Task Management. All results which I have presented here were directed towards the communication area and therefore I always had to use the shortest response time (best case from many measurements). That means that they were taken after the server process was firstly activated. The response time of the first request in which the activation (fork) is done contains another delay and therefore the first RPC response is much longer than the RTRT presented in previous tables. In practice it represents a time between 0.3 – 6 sec. depending on the load of the host. For the user it means that he must calculate this fact and if he needs fast response must keep his process in permanent activity instead of starting it repeatedly.

A second problem is more general. Performing short accurate software timing measurements within UNIX is practically impossible. There is normally a timer running with microseconds accuracy, however the system doesn't update it's software image fast enough. There is a 20 ms timing quantum which I couldn't overcome. This limitation was very annoying mainly in cases of very fast processors and this fact could have an influence on the precision of the measurements.

IV. CONCLUSION

On the basis of these results and knowledge of the network topology we can estimate the response time in any part of the network. This can be useful for future decisions or for reconfiguring the system because other applications will have similar response time overheads. The measurement also showed that the communication system has enough capacity to transfer substantially higher load through the network and also shows that the bottle-neck of the communication system is not in the hardware level of the TR but rather in the software or hardware of the hosts and gateways.

V. ACKNOWLEDGEMENT

The results of that work has been made possible by the close collaboration within the team. I would like to thanks particularly P. Lienard, L. Guerrero and A. Bland for their contribution to this work and P. G. Innocenti and K. Kostro for fruitful discussions.

VI. REFERENCES

[1] P. G. Innocenti, "The LEP Control System", International Conference on Accelerator and Large Experimental Physics Control System, Vancouver, Canada, November 1989.

[2] P. Lienard et al., "The SPS and LEP Control Network Architecture", International Conference on Accelerator and Large Experimental Physics Control System, Vancouver, Canada, November 1989.

[3] J. S. Haugdahl, "Inside the Token Ring", North-Holland 1987.

[4] K. Kostro, "Host Performance in the TCP/IP Internet of SPS", Internal report, CERN-SPS/ACC, January 1989.

[5] P.S. Anderssen, V. Frammery, G. Morpurgo, "User guide to the Network Compiler Remote Procedure Call (NC/RPC), LEP Control Note 97, May 1989.

[6] P. Van Der Stock "Remote Procedure Call Measurements", CERN SPS/ACC Note 16/12/1988.