25/80

327

175
9-17-73

ON THE USE OF NETWORK JOB PROCESSING (NJP)

Douglas E. Engert

U of C-411A-USAEC

ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

Prepared for the U.S. ATOMIC ENERGY COMMISSION
under Contract W-31-109-Eng-38

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Atomic Energy Commission, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona	Kansas State University	The Ohio State University
Carnegie-Mellon University	The University of Kansas	Ohio University
Case Western Reserve University	Loyola University	The Pennsylvania State University
The University of Chicago	Marquette University	Purdue University
University of Cincinnati	Michigan State University	Saint Louis University
Illinois Institute of Technology	The University of Michigan	Southern Illinois University
University of Illinois	University of Minnesota	The University of Texas at Austin
Indiana University	University of Missouri	Washington University
Iowa State University	Northwestern University	Wayne State University
The University of Iowa	University of Notre Dame	The University of Wisconsin

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22151
Price: Printed Copy \$4.00; Microfiche \$0.95

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

ON THE USE OF NETWORK JOB PROCESSING (NJP)

by

Douglas E. Engert

Applied Mathematics Division

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

August 1973

MASTER

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

TABLE OF CONTENTS

	<u>Page</u>
ABSTRACT.	5
1.0 Introduction, NJP and How It Works	5
2.0 The Changeover Period.	6
3.0 Testing and Implementation	7
4.0 Suggestions for Improvement.	10
5.0 Conclusion	12
ACKNOWLEDGMENTS	12
APPENDIX A.	13
APPENDIX B.	20
APPENDIX C.	21

THIS PAGE
THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

ON THE USE OF NETWORK JOB PROCESSING (NJP)

by

Douglas E. Engert

ABSTRACT

This document describes the use of the NJP feature of ASP during a changeover period to an IBM System/360 model 75-195 computer system from an IBM System/360 model 50-75 computer system.

NJP allows one ASP system to communicate with another by sending jobs back and forth between systems. However, the three major elements of NJP, (1) a DSP called by the operator to create an NJPOPEN job; (2) a DSP created by NJP to open a line, using BTAM; and (3) a DSP that may be invoked by NJPOPEN to receive a job, or invoked by a job for transmitting, caused problems during the changeover, and it was found necessary to improve their functions. This paper brings to light several of these problems, presents solutions for some of them, and suggests improvements for the overall system. It is assumed that the reader has some knowledge of IBM systems and nomenclature.

1.0 Introduction, NJP and How It Works

Network Job Processing (NJP) permits one ASP system to communicate with another. Either a user can request that his job be sent to another processor via /*PROCESS cards, or the operator may send jobs back and forth and request that certain functions be performed by one or the other system. To perform these tasks, NJP consists of three major elements.

(a) NJP, a DSP called by the operator, which is used to create an NJPOPEN job to open a line, and to create an inquiry job to allow the operator to inquire about the status of jobs in the other system. It can also be used to route jobs to the other system.

(b) NJPOPEN, a DSP created by NJP to open a line. NJPOPEN uses BTAM and it does the read initial; if something is being sent from the other system it will issue an INTERCOM macro, which simulates the operator command to start a DSP, to start NJPIO.

(c) NJPIO, a DSP which may be invoked by NJPOPEN when receiving or invoked by a job for transmitting. (i) In the receive mode, NJPIO will issue a read continue, read the control blocks and place them in the ASP queue, notify NJPOPEN that a read initial has been performed, and then purge. (ii) For transmission, a halt I/O is issued for the read initial and the job to be sent is transmitted. On completion of the send, a read initial is done, all uncompleted functions of the job are marked complete until the next NJPIO or purge is completed, and then the job is released. The job will then be purged or the next NJPIO function is processed. NJPIO is also used to handle operator inquiries about job status.

Each of the above three functions has its own FCT and it is possible to have an NJPOPEN and an NJPIO for each line.

A more complete description of the above can be found in

- (a) the ASP Application Programmer's Manual, GH20-0322,
- (b) the ASP Console Operator's Manual, GH20-0321,
- (c) the ASP System Programmer's Manual, GH20-0323, and
- (d) the ASP System Manual, GY20-0305.

2.0 The Changeover Period

During the changeover to our present system, our local time-sharing system, RESCUE, had to be operational while the IBM timesharing system, TSO, was being implemented. Two RESCUE features, job submission and inquiry had to interface with ASP. RESCUE and ASP both resided on the 360/50, and since the 50 could not possibly drive our new 195, it was decided to run ASP on the 195 with a local main, and to run ASP and RESCUE on the 50 with no main. NJP was then used to transmit jobs from the 50 to the 195 over a 50Kb line.

NJP was modified so that it could handle RESCUE inquiries. To do this, the local module which handled inquiries as to job status and backlog were modified to work like NJPINQ, on the 195. An additional RESCUE interface routine, NJPRESCU, was added to call NJPCOMM which creates an NJPQUEUE job. Also, modifications were made to route the inquiries to the three local modules, INQJ, INQB and NJPRESCU. Input service was changed to add a

scheduler element for NJPIO so that all jobs submitted via RESCUE would be sent to the 195.

3.0 Testing and Implementation

During the initial testing of NJP, many errors were found; among these were: no check by NJPIO and NJPOPEN if the line was open; problems with restart of a line; possible abends when cancelling a line; failure of reset to work; and possible contention lockups. However, the most difficult problem was not realized until the system was placed into production. Many timeouts were recorded, most of which required cancelling the line. At first it was thought that the line was bad, but the problem was found to be in the algorithm in the NJP system.

In a teleprocessing system with BTAM, which is used by NJP, the way one machine detects the functioning of the other machine is by receiving an acknowledgment. If the response is not received in three seconds, BTAM will repetitively try, every three seconds, for a response for up to seven times. If no response is received, the error is considered permanent and NJP shuts the line down until the operator responds with a restart.

The response to a previous block of data is sent when the next READ is issued, thus for no timeouts the time between reads should be no more than three seconds. If it is not possible to process the block of data within the three second time limit, BTAM allows a WRITE "Wait-Before-Transmit" response (WACK) to be sent. The other ASP system should then issue a WRITE Inquiry to determine if the first system is still busy.

After receiving each block of data, NJPIO makes no attempt to respond within the three second time limit. This does not create a problem for any block of data except the first and the last blocks, when several or more blocks are being used. When NJPOPEN receives the first block, an INTERCOM macro is issued to call NJPIO as a DSP. NJPIO then processes the rest of the job. Since CALLDSP must be invoked and NJPIO must be scheduled by JSS before a response is sent for the first block, timeouts are frequent on the other system. Under our normal load, NJP would timeout after sending 3 to 10 jobs until a correction was made. The first attempt to correct this problem was to have a special copy of CALLDSP, to call NJPIO, run under its own

FCT. This was unsuccessful. NJPOPEN was then modified to send the WACK response until NJPIO was scheduled by JSS. Also, NJPIO was modified to issue the WRITE Inquiry macro, after a delay of three seconds, each time the WACK response was received. This cut the timeout rate from every ten minutes to every two to three hours.

Timeouts were also frequent after receiving the last block. After the last block, NJPDJ writes all of the job tables to disk and updates the JCT. Since the JCT may be busy, this could result in a wait of several seconds. The sending system would then timeout. The receiving system would finish and attempt to read the end of file record and it would timeout also.

Upon restarting the line, the sending system, not having sent the whole job, would reschedule it for later; the receiving system had already added the job to the ASP queue. Thus it is possible to send a job twice. An attempt to alleviate this problem was made by modifying NJPDJ and NJPOPEN to have NJPOPEN send WACK responses while NJPDJ was adding the job to the ASP queue.

Several other problems and the solution to some of them are described below. All problem were not corrected. Some were only circumvented or bypassed, since NJP was only to be used on a temporary transitional basis. Several of the problems were actual benefits of the NJP system. All of these are included in the enumeration listed below.

(a) We found the *RESTART line command did not work. Some bits were not reset, so NJPIO on the sending system would return to JSS and the job would be left on the wrong system. (The first time this happened about 20 jobs were left stranded on the wrong system, and the dump job DSP had to be used to move the jobs.) To get around this problem, the line was always cancelled and reopened instead of restarting.

(b) Both systems must have the same control block format, buffer size, and list of DSPs.

(c) When the NJPOPEN job is created by NJP, it does not first check to see if the line is open. It overlays the pointer to the NJODATA csect in the line table and does an OS open. This may cause problems since the operator may not be able to cancel one of these.

(d) NJPIO in receive mode does not check to see if the line is open; it uses pointers in the line table for NJODATA which usually has been deleted.

(e) NJPIO in the receive mode does not check to see if NJPOPEN is waiting for it. (This can occur if the NJPIO was called and ASP went down.) After a restart of ASP and the opening of the line, this NJPIO may become active.

(f) There was no provision for contention regulation. (If both systems did a write the line would be hung up.)

(g) If the line was cancelled at the wrong time, several things might happen.

i) During an inquire it might take ASP down and since the inquire module does not have a cleanup routine, a branch to location zero was possible.

ii) While reading in parameter buffers, ASP might go down because NJPDJ tried to free up all the parameter buffers for the job, even though it had not read all of them.

iii) Just after placing the job in the queue, ASP might go down because NJPIO does one more read to get the end of file from the other system (which may have timed out). NJPDJ did not reset the bits to keep track of which blocks must be freed.

(h) A restart of the line does not work. It can cause any scheduled job using the NJODATA csect of the restarted line to have NJPIO marked complete and put back in the queue without being sent.

(i) NJPOPEN has missing code. (If the line name is spelled wrong, it branches to some constant.)

(j) It is very easy to receive two or more copies of the same job. If the receiving system takes too long to add the job to the JCT, the sending system times out. The receiving system tries to read one more block, and it times out also. When the line is cancelled, the receiving system has already put the job in the queue, while the sending system reschedules the job, since it had not sent the final record (the end of file). Since the JCT is a well used resource, this can happen quite often.

(k) Neither NJPOPEN nor NJPIO check to see if the line is open. This may result in opening the line more than once, thus requiring a restart of ASP. (Tight operator control remedied this situation.)

Some of the problems found in NJPCOMM are that the RESET function of SEND did not work. SEND added scheduler elements to the JCT entry, thus increasing the JCT entry size. This could cause problems if it became too large for work areas such as in INQUIRE. Trying to send a job which had been reset would cause the complete bit in the scheduler element that had been reset before to be turned on. Thus there would now be an extra NJPIO scheduler element.

NJPDJ had two problems related to cancelling the line and cleaning up. If some parameter buffers had been received, but not all, an attempt to free up all these buffers was made, resulting in bad addresses and an ASP dump. At one place the wrong byte was being tested for the receive mode. (See Appendix A.) A problem that did not affect us since we were using NJP only one way was contention. A fix was supplied by our local system engineers so that if two jobs are trying to be sent, the one on the remote system would be rescheduled. Even so, there is a problem if the rescheduled job is at a higher priority than the NJPIO job; then the rescheduled job would be rescheduled again.

In our system we had previously split the functions of CALLDRVR among three FCT entries; a fourth FCT was added which was used for calling NJPIO. This solved the problem of having a call for NJPIO from NJPOPEN rejected and thus being lost. In the straight IBM version, the message could be lost if 15 messages were queued for CALLDRVR.

4.0 Suggestions for Improvement

If NJP is to function at all, the timeout problem must be solved. There are a number of ways to attack this problem, of which two are given.

The first and easiest method is to add timer routine to NJPOPEN so that if NJPIO is busy, NJPOPEN can send the WACK response. NJPOPEN should then handle all I/O, with NJPIO posting NJPOPEN, whenever it is ready for the next transmission.

The second method, which requires more effort but would save on system resources is to use RJP. Implementing the dataset type of multileaving with changes so RJP could interface with itself on another ASP system would solve most of the NJP problems. BTAM would not be needed, the timeout problem would be handled by RJP, and contention would be solved by two-way multileaving. The inquire function would then be handled by the normal ASP modules through the RJP console functions. The only 3 modules that are required are NJPIO, NJPDJ, and a data csect. NJPDJ would not need any modification, but NJPIO would need many revisions, such as interfacing with RJP, eliminating the inquire function, and handling operator commands directly. RJP would also need some changes to interface with itself, such as a sign-on routine, and possibly a dial-up routine for switched lines.

A number of restrictions have been placed on NJP which could be eliminated. These restrictions include: 1) ASP buffer size, DSP dictionary, and job tables must be compatible between the two systems, 2) a job may have only one set of job control cards, which usually limits it to one processor. Consider an example: Job MOVEDATA is submitted at location A and is to run on main to collect some data to be sent to location B. When at B, it is to run again and use this collected data. Under the present NJP system, this is a rather difficult task because the same JCL must be used. One approach to this problem under the present system is to use special cataloged procedures. Appendix C contains two procedures called ATOB and a job. While at A the job uses the procedure to unload the dataset using IEHMOVE into an ASP file. At B, IEHMOVE is used to restore it.

To overcome these restrictions, the following could be done.

- (a) Change input service to accept input records of any size up to the buffer size. These large records are only needed for datasets created by /*DATASET or //ddname DD cards.
- (b) Include on the NJP format card a parameter of DDNAME=... which would be the ASP dataset that would be sent to the other system as a new job.

- (c) Allow other datasets to be sent by NJP looking for a /*ASPDATA card in the pattern dataset. The form of this card could be:

```
/*ASPDATA,DDNAME=ddname[,RENAME=ddname][,DATASET=YES|NO][,J=YES|NO]
```

This card would cause a //ddname DD* card or /*DATASET DDNAME=ddname[,J=YES|NO] card to be inserted, followed by the data. Thus, a programmer could send selected datasets, including other pattern datasets to the other system. See Appendix B for an example of a job that uses these features.

- (d) To allow a job to be sent from one system to another via a third system, allow NJP on the sending system to determine which route to take, and if the next system on the route is the last, input service would be invoked in that system; otherwise, a module NJPVIA would be called to make up a job with a NJP scheduler element and a single dataset consisting of the data sent from the first system which would be sent on to the next system.

5.0 Conclusion

NJP as it stands today will not work under more than a minimal type load. Many changes are required, and suggestions have been given on how it could be rewritten to use RJP and eliminate many of the present restrictions.

ACKNOWLEDGMENTS

I wish to thank L. T. Bryant, Technical Editor, for his assistance in writing this paper and R. F. Krupp for programming the RESCUE-ASP interface.

APPENDIX A

THE FOLLOWING IS A LIST OF CHANGES APPLIED TO THE NJP MODULES
OF VERSION 2.6 RELEASE 5 OF ASP.

./	CHNGE NJPDJ,00,0,0,73200220		A26024 00029100
	NJODSECT TYPE=DS ECT		A26024 00029200
	NJPENTRY		A26024 00344500
	OI JDPARFDB,X'80'	SET BUF HAS BEEN READ	A26024 00481100
	L 8,NJLNETAB	POINT AT LINE TABLE	A26024 00481200
	USING NJPLINES,8		A26024 00481300
	NI NJPLFLG1,X'FE'	TELL NJPOPEN WE ARE BUSY	A26024 00481600
	L 8,NJPCSADD	POINT AT NJODATA SY	A26024 00481700
	USING NJODATA,8		A26024 00506500
	OI NJOECF,X'20'	TELL NJPOPEN TO SEND WACK	A26024 00551500
	NI JDPARFDB,X'7F'	TURN OFF BIT	A26024 00560000
NJDC10	MVI NJPDFL1,X'00'	ZERO BYTE	A26024 00573050
	TM NJPDFLG,RECEIVE	TEST FOR RECIVE MODE	A26024 00573060
	TM NJPDFLG,RECEIVE	IS THIE RECEIVE ?	A26024 00573100
	BC ALLOFF,ANLC35	YES,BRANCH	A26024 00573200
	TM JDPARFDB,X'80'	HAS THIS ONE BEEN READ?	A26024 00573300
	BC ALLOFF,NJDC40	NO LAST PARM WAS THE LAST	A26024 00573900
	NI JDPARFDB,X'7F'	TURN OFF BIT	A26024 00577100
ANLC25	EQU *		A26024 00577200
ANLC35	CLI JDPARFDB,X'00'	IS BUFFER IN CORE ?	A26024 00577300
	BC EQ,ANLC25		A23118 00780100
	B NJDC30		A23118 00780200
	CLC 4(4,1),=X'FEFEFEFE'	END OF JAB ?	A23118 00780300
	BC NE,ANLT20	NO,BRANCH	A23118 00780400
	LA 1,8(1)	INC PAST SPECIAL JAB END	A23118 00780500
	B NJDT10	GO LOOK FOR END	
ANLT20	EQU *		

./	CHNGE	NJPCOMM,00,0,0,72201129		
NJCRETA	DC	F'0'		A26004 00062100
NJCRETB	DC	F'0'		A26004 00062200
NJC111	TM	NJFLAG,RESET	WAS THIS RESET?	A26015 00105120
	BC	ALLON,NJC112	YES,BRANCH	A26015 00105130
	TM	NJCFLAG1,RDEST	WAS DEST SPECIFIED?	A26015 00106000
NJC112	EQU	*		A26015 00106125
	TM	NJFLAG,RESET	WAS THIS RESET ?	A26015 00193400
	BC	ALLON,NJC160	YES,BRANCH NOTHING TO RESET	A26015 00193500
	CH	2,=H'18'	WILL 20 SE'S FIT?	A26004 00289100
	BC	GT,NJC170	NO,BRANCH	A26004 00289200
NJC170B	EQU	*		A26004 00320100
	CLI	JCTNOSES,18	TO MANY SES ?	A26004 00320200
	BC	LE,ANL170B	NO,BRANCH MUST BE MAIN CODE	A26004 00320300
	TM	NJCFLAG,NJOB	TEST FOR SINGLE JOB	A26004 00320400
	BC	ALLOFF,ANL170C	NO,BRANCH	A26004 00320500
	LH	5,NJCJOBX		A26004 00320580
	BAL	14,NJCCOND	CONVERT JOB NUMBER TO HEX	A26004 00320590
	MVC	ANLMSG1+11(4),NJCJOBNG		A26004 00320600
	TAREL			A26004 00320650
	LA	1,ANLMSG1	POINT AT MESSAGE	A26004 00320700
	B	NJCRESP	GO TYPE MESSAGE	A26004 00320800
ANL170B	EQU	*		A26004 00320900
ANL170C	CLC	SEDSPNO,MAINSE	IS MAIN SE THE ACTIVE ONE	A26004 00321000
	EQU	*		A26004 00324500
	MVC	NJCPRTY(1),JCTPRTY		A26015 00370500
	MVC	0(8,6),=X'FFFFFFFFFEFEFEFE'	SET END OF JAB	A23118 00571000
	LA	6,11(6)	MAKE ROOM FOR TERM	A23118 00571100
	N	6,=X'FFFFFFFFC'	PUT ON WORD BOUNDARY	A23118 00571150
	L	1,NJCJAB	POINT AT JAB	A23118 00571200
	USING	JABSTART,1		A23118 00571300
	LR	3,1		A23118 00571350
	AH	3,ANLADDR	POINT AT ANLENTY	A23118 00571400
	MVC	0(L'ANLSIZE+16,6),0(3)	MOVE ANLENTY+TERM	A23118 00571500
	S	6,=A(JABBLD-L'JABSIZE)	FIND ANL ADDR OFFSET	A23118 00571600
	SIH	6,ANLADDR	SAVE NFW OFFSET	A23118 00571800
	MVC	0(JABBLDS,6),JABBLD	MOVE JAB BUILD AREA	A23118 00595000
	MVC	JABBLDS(ANLBLDS,6),ANLBLD	MOVE ANL ENTRY PART	A23118 00595500
	MVC	0(JDBBLDS,7),JDBBLD	MOVE JOB BUILD AREA	A23118 00596000
NJC210	EQU	*		A26015 00626900
	TM	NJCFLAG,NUMB	TEXT FOR NUMBER OF JOB	A26015 00628000
	CLI	NJCIMSG,X'40'	INQUIRE?	A26016 00693132
	BE	NJCBOO	NO.	A26016 00693134
	MVC	NJPIMSG(21),NJCIMSG	YES.	A26016 00693136
NJCBOO	DS	0H		A26016 00693138
	MVI	NJCIMSG,X'40'	BLANK TEMP SAVE AREA	A26016 00760165
	MVC	NJCIMSG+1(20),NJCIMSG		A26016 00760166
	CLC	0(5,1),=CL5'DEST='	TEST FOR DEST.	206A 00760254
	BC	EQ,NJC360	BRANCH IF YES.	206A 00760256
	CLC	0(2,1),=CL2'J='	INQUIRE ON A SINGLE JOB?	A26016 00760257
	BE	NJC370A	YES.	A26016 00760258
	CLI	0(1),C'B'	NO.INQUIRE ON BACKLOG?	A26016 00760260
	BE	NJC370A	YES	A26016 00760262
	CLC	0(6,1),=CL6'RESCUE'	RESCUE TERMINAL?	A26016 00760352
	BNE	NJC331A	NO.	A26016 00760354
	MVI	NJCCONS,C'R'	YES.SAVE CONSOLE NUMBER 'R'	A26016 00760355
	B	NJC320	CHECK FOR ADDITIONAL PARAMETER	A26016 00760356
NJC331A	DS	0H		00760358
	SPACE	2		00811002

*-----A26016 00811004			
* MOVE INQUIRE REQUEST INTO TEMPORARY SAVE A26016 00811006			
*-----A26016 00811008			
NJC370A	LA	4,1	INITIALIZE FOR COUNTING CHARS. A26016 00811010
	LA	3,1(1)	A26016 00811012
	LA	5,19(1)	(MAX. OF 20 CHARACTERS). A26016 00811014
NJC371	CLI	3,C' '	COUNT THE CHARACTERS. A26016 00811016
	BE	NJC372	A26016 00811018
	BXLE	3,4,NJC371	A26016 00811020
NJC372	SR	3,1	NUMBER OF CHARACTERS. A26016 00811022
	BCTR	3,0	DECREMENT FOR MOVE INSTRUCTION A26016 00811024
	STC	3,NJC373+1	COUNT->MOVE. A26016 00811026
NJC373	MVC	NJCIMSG(1),0(1)	MOVE MESSAGE. A26016 00811028
	B	NJC305	GET JOB NUMBER. A26016 00811030
NJCERR3A	LH	5,NJCJOBX	A26015 01103000
	BAL	14,NJCCOND	A26015 01103100
	MVC	NJCMSG3+13(4),NJCJOBNO	A26015 01103200
NJCERR4	LH	5,NJCJOBX	A26015 01110000
	BAL	14,NJCCOND	A26015 01110100
	MVC	NJCMSG7+13(4),NJCJOBNO	A26015 01110200
	TAREL		A26015 01110300
*NJCRETA	DC	F'0'	MOVED TO 00062100 A26004 01148000
*NJCRETB	DC	F'0'	MOVED TO 00062200 A26004 01149000
BLDSE	DC	((20*L'SESIZE+8+3)/4)F'0'	A26004 01172000
JABBLD	DC	((21*L'JABVSIZE+8+3)/4)F'0'	A26004 01173000
JABBLDS	EQU	*-JABBLD	A26004 01173010
ANLBLO	DC	((L'ANLSIZE+27)/4)F'0'	A26004 01173500
ANLBLODS	EQU	*-ANLBLO	A26004 01173510
JOBBLO	DC	((21*L'JOBVSIZE+8+3)/4)F'0'	A26004 01174000
JOBBLODS	EQU	*-JOBBLO	A26004 01174010
NJCIMSG	DC	21C' '	INQUIRE MESSAGE A26016 01221150
ANLMSG1	DC	AL1(ANLMSG1E-ANLMSG1S)	A26004 01257050
ANLMSG1S	DC	C'NJC99 JOB HAS MORE THEN 18 PROCESSES, '	A26004 01257060
	DC	C'NOT SCHEDULED FOR NJP'	A26004 01257070
ANLMSG1E	EQU	*	A26004 01257080

./	CHNGE NJPINQ,00,0,0,73200220		
	CLI NJPQIMSG,C'J'	INQUIRE J= RESPONSE?	A26016 00194100
	BE NJQ400	YES.	A26016 00194200
	CLI NJPQIMSG,C'B'	NO. INQUIRE B RESPONSE?	A26016 00194300
	BE NJQ400	YES.	A26016 00194400
NJQ400	L 6,NJPDATA	POINTER TO DATA	A26016 00247010
	LA 14,NJQNJPR	RETURN ADR. FOR NJQNJPR RJN.	A26016 00247020
NJQ410	CLC 0(4,6),=X'C5D5C4FF'	END OF DATA?	A26016 00247030
	BE NJQNJPR	YES.GO RETURN	A26016 00247040
	L 0,NJPJDS	NO.CONSOLE NUMBER.	A26016 00247050
	LR 1,6	ADR.OF RECORD	A26016 00247060
	RESPONSE DEST=(0),TXT=(1)		A26016 00247070
	LA 6,72(6)	NEXT.RECORD.	A26016 00247080
	B NJQ410	GO CHECK FOR END OF DATA.	A26016 00247090

./	CHNGE NJPIO,00,0,0,73200220		
	B ANL055		A26024 00112500
ANL055	EQU *		A26024 00131050
	MVC NJPQFROM(48),NJPFROM	SAVE PARAMETERS IN CSECT	A26016 00175120
	TM NJPLFLG1,X'02'	CONTENTION ON LINE?	A26014 00182700
	BC ALLON,NJP515	LET JSS SCHEDULE JOB LATER	A26014 00182800
	CLI NJPQCONS,C'R'	INQUIRE FROM RESCUE?	A26016 00190100
	BNE ANL350A	NO.	A26016 00190200
	ALOAD EP=NJPRESCU	YES.LOAD RESCUE INQUIRE RESPON	A26016 00190300
	B ANL350B		A26016 00190400
ANL350A	DS OH		A26016 00190500
ANL350B	DS OH		A26016 00193500
	OI NJPLFLG1,X'01'	SET NJPIO IN CORE	A26024 00212500
	STM 0,15,NJPREGSV		A26024 00216100
	BAL 5,NJPWAIT	WAIT FOR DECB TO COMPLETE	A26024 00216200
	BAL 5,NJPTIO	GO TEST IO	A26024 00216300
	LA 0,ANL340	RETURN ADDR	A26024 00306000
ANL340	TM NJOECF,X'20'	NJPOPEN WACKING?	A26024 00311100
	BC ALLOFF,NJP340	NO,BRANCH	A26024 00311200
	NI NJOECF,X'0F'	TELL IT TO STOP	A26024 00311300
	OI NJPLFLG1,X'01'	TELL IT TO STOP	A26024 00311400
	BAL 5,NJPWAIT	GO WAIT FOR IO	A26024 00311500
	BAL 5,NJPTIO	GO TEST IO	A26024 00311600
	MVC NJPQFROM(48),16(4)	SAVE 'Q' PARAMETERS.	A26016 00331100
	CLI NJPQMSG,C'J'	INQUIRE J=?	A26016 00331105
	BNE ANL090A	NO	A26016 00331110
	ALOAD EP=INQJ	YES.LOAD INQJ	A26016 00331120
	B ANL090C		A26016 00331130
ANL090A	CLI NJPQMSG,C'B'	INQUIRE BACKLOG	A26016 00331140
	BNE ANL090B	NO.	A26016 00331150
	ALOAD EP=INQB	YES.LOAD INQB	A26016 00331160
	B ANL090C		A26016 00331170
ANL090B	DS OH		A26016 00331180
ANL090C	DS OH		A26016 00332100
	MVC 16(48,4),NJPQFROM	MOVE IN 'Q' PARAMETERS.	A26016 00364120
NJP441	MVC 64(2,4),=X'1026'	MOVE IN ENDING CHARACTERS.	A26016 00365000
	LH 3,=H'66'	SET BYTE COUNT.	A26016 00366000
	NI NJPFLAG2,X'FF'-NJPTIX		A26024 00383050
	TM 24(4),X'80'	WAS 'WACK' GIVEN ?	A26024 00383100
	BC ALLOFF,ANL480	NO,BRANCH	A26024 00383150
	MVI NJPE,X'00'	RESET TIMER ECF	A26024 00383200
	LA 1,NJPTAPP	POINT AT TIMER ROUTINE	A26024 00383250
	LA 0,300	LOAD 3 SEC WAIT	A26024 00383300
	ATIME TIME=(0),ENTER=(1)	WAIT FOR 3 SEC	A26024 00383400
	LA 0,NJPE	ECF ADDR	A26024 00383450
	AWAIT ECFMASK=X'7F',ECFADD=(0),RETURN=ANL470		A26024 00383500
ANL470	ST 13,NJREG13		A26024 00383550
	LA 13,NJPREGSV		A26024 00383600
	LA 1,NJODECAD		A26024 00383650
	XC 0(4,1),0(1)	CLAER ECB	A26024 00383700
	WRITE (1),TQ,MF=E	SEE IF OTHER MACHINE IS READY	A26024 00383750
	L 13,NJREG13		A26024 00383800
	LTR 15,15		A26024 00383850
	BC NZERO,ANL470		A26024 00383900
	B NJP455		A26024 00383950
ANL480	EQU *		A26024 00383990
NJP505	TM NJPDFLG,RECEIVE	INQUIRE IN RECEIVE MODE?	A26016 00455000
	BZ NJP505A	NO.	A26016 00455002
	CLI NJPQCONS,C'R'	YES.RESCUE INQUIRE?	A26016 00455004

	BNE NJP505C	NO.	A26016 00455006
	ADELETE EP=NJPRESCU	YES.DELETE RESCUE INQUIRE MODUA	A26016 00455008
	B NJP510		A26016 00455010
NJP505A	CLI NJPQIMSG,C'J'	INQUIRE J=?	A26016 00455012
	BNE NJP505B	NO.	A26016 00455014
	ADELETE EP=INQJ	YES.DELETE INQJ	A26016 00455016
	B NJP510		A26016 00455018
NJP505B	CLI NJPQIMSG,C'B'	INQUIRE BACKLOG?	A26016 00455020
	BNE NJP505C	NO.	A26016 00455022
	ADELETE EP=INQB	YES.DELETE INQB	A26016 00455024
	B NJP510		A26016 00455026
NJP505C	ADELETE EP=NJPINQ	NO.DELETE NJPINQ	A26016 00455028
	NI NJPLFLG1,X'FE'	RESET NJPIO IN CORE BIT	A26024 00465500
	BC ALLON,NJPT49	YES,BRANCH	A26014 00490000
NJPT49	CLI 0(4),X'41'	PERMENT I/O ERROR?	A26014 00507100
	BNE NJPT50	NO,BRANCH	A26014 00507200
	TM 24(4),X'10'	CONTENTION?	A26014 00507300
	BC ALLOFF,NJPT50	NO,CONTINUE	A26014 00507400
	OI NJPLFLG1,X'02'	SET CONTENTION BIT	A26014 00507405
	MVI 0(4),X'00'	RESET ECB	A26014 00507500
	NI 24(4),X'FF'-X'10'	CLEAR CONTENTION BIT	A26014 00507600
	OI NJPDFLG,LINECAN	SET TO DO THIS JOB LATER	A26014 00507700
	BC 15,NJP490	GO SCHEDULE THIS JOB LATER	A26014 00507800
	LH 3,=H'66'	CHARACTER COUNT	A26016 00515050
	STM 0,15,NJPREGSV		A26024 00524100
	LTR 15,15		A26024 00525100
	BC NZERO,ANL2015		A26024 00525200
	LR 14,0		A26024 00525300
	BR 14		A26024 00525400
ANL2015	EQU *		A26024 00525500

./	CHNGE NJOPEN,00,0,0,73200220		
	OI NJOECFL,X'E0'	CHECK FOR WACK FORM NJPIO	A26024 00029500
	ARELEASE FDB=(7)		A26024 00048000
	ARELEASE FDB=(6)		A26024 00049000
	NI DCBXCODE,X'7F'	TURN OFF REMOTE BIT	A26014 00077100
	CLI NJPLINTF,C'1'	IS THIS TO BE REMOTE?	A26014 00077200
	BC LT,ANLNJ70	NO,BRANCH	A26014 00077300
	OI DCBXCODE,X'80'	SET REMOTE BIT	A26014 00077400
	CLI NJPLINTF,C'1'	IS THIS TO BE REMOTE?	A26014 00077450
	BC EQ,NJO080	IF '1' THEN INTERFACE A	A26014 00077500
ANLNJ70	EQU *		A26014 00077600
*	LH 5,=H'48'	SET BYTE COUNT.	A26016 00158000
	LH 5,=H'72'	SET BYTE COUNT.	A26016 00158010
	MVI NJOLFLG,X'04'	SET ECF MACK	A26024 00166500
	TM NJOECF,X'20'	START WZCKS?	A26024 00179020
	BC ALLON,ANL620	YES,BRANCH	A26024 00179040
	NI NJPLFLG1,X'FF'-X'02'	CLEAR CONTENTION IF ANY	A26014 00192500
ANL620	LA 1,NJODECAD		A26024 00195050
	XC 0(4,1),0(1)	CLEAR ECB	A26024 00195075
	LA 13,NJOREGSV		A26024 00195100
	WRITE (1),TW,MF=E	TELL OTHER MACHINE TO EAIT	A26024 00195150
	LTR 15,15		A26024 00195200
	BC NZERO,ANL620		A26024 00195250
	MVI NJOLFLG,X'00'	SET DONT WAIT ON ACTIVE BIT	A26024 00195300
	B NJO160	GO AWAIT	A26024 00195450
NJO610	EQU *		A26024 00196000
	TM NJPLFLG1,X'01'	IS NJPIO IN CORE YET?	A26024 00196100
	BC ALLON,ANL630	YES,BRANCH	A26024 00196200
	CLI NJODECAD,X'7F'	COMPLETE OK?	A26024 00196300
	BC EQ,ANL620	YES,GO WRITE WACK	A26024 00196400
ANL630	LA 7,NJOECFL		A26024 00196500

APPENDIX B

FOLLOWING IS AN EXAMPLE OF A JOB THAT COULD USE THE PROPOSED FORM OF NJP TO SUBMIT A NEW JOB TO LOCATION B WITH DATA FROM LOCATION A. NJP WOULD BE INVOKED TO SEND THE ASP DATASET CALLED TOB TO LOCATION B AS A JOB. THE /*ASPDATA CARD WOULD BE USED TO INCLUDE THE TOB DATASET. NOTE: INPUT SERVICE ON THE RECEIVING SIDE SHOULD BE CHANGED TO HANDLE RECORDS LARGER THEN CARD IMAGE.

```

//MOVEDATA JOB (W17783,,,2),CLASS=A,REGION=60K
/*PROCESS MAIN
/*PROCESS PRINT
/*PROCESS NJP
/*FORMAT NJP,DDNAME=TOB,DEST=8
// EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=DS.TO.B,...
//SYSUT2 DD UNIT=(CTC,,DEFER)
//SYSIN DD DUMMY
/*DATASET DDNAME=TOB,J=YES
//SAVEDATA JOB (W17783,,,2),CLASS=A,REGION=60K
/*ASPDATA DDNAME=SYSUT2,RENAME=SYSUT1,DATASET=YES
// EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD UNIT=(CTC,,DEFER),...
//SYSUT2 DD DSN=DS.FROM.A,...
//SYSIN DD DUMMY
/*ENDDATASET

```

IN THIS EXAMPLE, JOB SAVEDATA WOULD BE SUBMITTED AT DESTINATION B WITH THE DATA COLLECTED FROM LOCATION A.

APPENDIX C

FOLLOWING IS AN EXAMPLE OF A JOB THAT COULD BE USED TO MOVE A DATASET FROM ONE LOCATION TO ANOTHER UNDER THE PRESENT VERSION OF NJP. SINCE THE SAME JCL MUST BE USED AT BOTH LOCATIONS, TWO CATALOGED PROCEDURES HAVE BEEN SETUP TO INVOKE IEHMOVE WITH DIFFERENT DATA.

```
//MOVEJOB JOB (W17783,,,3),CLASS=A,REGION=60K
/*PROCESS MAIN
/*PROCESS PRINT
/*PROCESS NJP
/*FORMAT,NJP,DEST=B,FROM=A
/*PROCESS MAIN
/*PROCESS PRINT
// EXEC ATOB,OLDVOL=PACK01,TAPEVOL=XXXXXX,NEWVOL=DISK01,
//          OLDUNIT=3330,NEWUNIT=3330
//ONE.SYSIN DD *
//          COPY  DSNAME=DS.FOR.B,TO=2400=XXXXXX,TODD=TAPE,
//          FROM=3330=PACK01
/*
//TWO.SYSIN DD *
//          COPY  DSNAME=DS.FOR.B,FROM=2400=XXXXXX,FROMDD=TAPE,
//          TO=3330=DISK01
/*
```

THE FOLLOWING PROCEDURE WOULD BE USED AT LOCATION A.

```
//ATOB    PROC  TAPEVOL=,OLDVOL=,OLDUNIT=,NEWVOL=,NEWUNIT=
//ONE     EXEC  PGM=IEHMOVE
//SYSPRINT DD UNIT=(CTC,,DEFER)
//SYSUT1 DD UNIT=3330,VOL=SER=SCRO01,DISP=OLD
//INPUT DD UNIT=&OLDUNIT,VOL=SER=&OLDVOL,DISP=OLD
//TAPE DD UNIT=(CTC,,DEFER),LABEL=(1,NL),VOL=SER=&TAPEVOL,
//          DISP=(NEW,PASS)
//TWO     EXEC  PGM=IEFBR14,PARM='&NEWUNIT,&NEWVOL'
```

AND THIS PROCEDURE WOULD BE USED AT LOCATION B.

```
//ATOB    PROC  TAPEVOL=,OLDVOL=,OLDUNIT=,NEWVOL=,NEWUNIT=
//ONE     EXEC  PGM=IEFBR14,PARM='&OLDVOL,&OLDUNIT'
//TWO     EXEC  PGM=IEHMOVE
//SYSPRINT DD UNIT=(CTC,,DEFER)
//SYSUT1 DD UNIT=3330,VOL=SER=SCRO01,DISP=OLD
//OUTPUT DD UNIT=&NEWUNIT,VOL=SER=&NEWVOL,DISP=OLD
//TAPE DD UNIT=(CTC,,DEFER),LABEL=(1,NL),VOL=SER=&TAPEVOL,
//          DISP=(OLD,PASS)
```

THESE PROCEDURES WITH THIS JOB COULD THEN BE USED TO MOVE THE DATASET DS.FOR.B FROM ONE SYSTEM TO ANOTHER.