UNIVERSITY OF MINNESOTA COMPUTER CENTER
Deadstart Systems Newsletter

10 June 1980 Vol. 6, No. 11

Send all comments, criticisms and contributions to the editor: T. W. Lanzatella, University Computer Center, 2520 Broadway Drive, Lauderdale, MN 55113. The University of Minnesota is an equal opportunity educator and employer.

TABLE OF CONTENTS

## NOTICE OF CHANGES TO THE SYSTEM

Tom Lanzatella reinstated code in DSD which generates the TAF O-display. This code was deleted in order to make room for MULTI but other space-saving changes make this one unnecessary. The Transaction Facility (TAF) will soon be offered as a standard package of the C172. Tom also installed new versions of the maintenance routines, EC3, ALX etc., correcting an error which occurred when they terminated after finding a hardware error - a frequent occurrence lately.

Kevin Matthews contributed changes necessary to accomodate his new, rewritten version of DSP. Details related to this change are explained in the original proposal (see DSN 6,1 p. 14-16). An added attraction of this change which was not mentioned in the proposal is that direct access permanent files can now be ROUTEd without first copying them to a local file.

Tim Hoffmann installed code to make the REP option on the ROUTE command work for plot files. If this change looks familiar you're right. Tim installed this same correction a couple months ago but it was lost somehow by TWL. Tim also installed two changes to RELOAD. One to enhance internal documentation, the other to ensure that RELOAD requests are charged to user number PF.

Bill Sackett installed the following changes.

1. The number of programs which SYSEDIT can place into central memory is limited. Bill increased the limit. This change is necessary in the MERITSS environment on the C720. New C720 hardware is very dependable and usually "rides through" power interruptions which bring down the C74, the C172 and ECS. Hence, an ECS-down configuration for the C720 is a practical necessity. With the ever increasing load on MERITSS, low response time gets harder to achieve and more programs have to be placed into central memory.

2. Program 1UD, the user ECS allocation routine, was changed. The restriction that jobs using ECS only get access to CPU 0 is now removed after ECS is returned.

Jeff Drummond altered TRANSIT to disallow SEND from replacing or retaining protected permanent files. Jeff also repaired an error in TRANSIT related to the relocation of a FET.

Brad Blasing submitted the following changes.

1. Program DSDI was changed to make the P-option read only the first 21B words to use as pointers rather than 200B words. This makes the CPU assignments and channel assignments correct on the B and PP displays.

2. The LINK loader was changed to always clear core thus patching up a security hole which has existed for sometime. This change also makes the LDSET(PRESET=...) upward compatible with the Cyber loader.

3. The LINK loader was changed to disable secure memory when it executes a binary unless it is execute-only. This change should have been made when we converted to NOS.

4. Brad installed a new version of CCL, the Cyber Control Language. Fixed is an execute-only procedure file problem. BEGIN processing was changed to return its scratch files when performing a job-level (initial) BEGIN.

Brad also installed the following PSR modsets received from CDC.

5. CCFD1$ - A modset to make COMCCFD handle input numbers greater than 999,999,999. The program now returns ********** in such a case.

6. SSSE2$ - Repair documentation related to a tag in COMSSSE.

7. ODIF17$ - A modset which fixes the MODIFY CG-option so that SYSTEXT is the second default.

8. QDIF18$ - A modset which fixes a mode 1 error in MODIFY when *UPDATE is the only input directive.

Steve Collins installed a new version of XEDIT with a fix for a long-standing, serious error. Occasionally, when editing a long, multi-record file, XEDIT would lose data near the end of the file when terminating.

## PROPOSED CHANGES TO THE SYSTEM

### Eliminate the Use of IQFT - by T. W. Lanzatella

Since the implementation of our use of the IQFT (Inactive Queue File Track) to store delay-type jobs, a serious operational problem has appeared. Jobs entering the system are given a job name before they appear in the input queue. After a job has been dequeued and then requeued it reenters the system with a new job name. This behavior presents a couple problems.

1. We have already had to install a special account file message to indicate the old and new job name.

2. Say a user enters two jobs with the same job name. A QUEUE(LIST...) report indicates that two jobs are present with job names differing only in the last three characters. The user decides that one of these jobs should not run and calls the operator asking for the job to be dropped. Meanwhile, both jobs begin running. The operator doesn't know which job to purge.

3. We expect, someday soon, to be installing a user queue file manipulation routine (supposedly available from Georgia Tech). I haven't seen the program but I doubt that it can handle IQFT-resident jobs or that it can deal with the jobname change problem.

I would like to eliminate these problems. The IQFT feature of delay jobs was originally mandated by a requirement to minimize the use of central memory. I don't think the volume of delay jobs justifies the volume of system plumbing necessary to accomodate the requirement. I therefore suggest that we eliminate the use of the IQFT to store delay queue jobs.

//////////

### COST and Connect Time - by T. W. Lanzatella

Ever since we converted from KRONOS to NOS and SRU accounting, connect time has not been available to users. Under KRONOS, connect time was always printed as part of the logoff message. Several users have asked that this be made available again. I don't think that the old logoff message should be reinstated because we have made a committment to adhere to SRU accounting. I propose that COST be modified so that for time sharing users an extra dayfile message indicating accumulated connect time is printed. This shouldn't be very hard to install since a lot of code from DAYFILE could probably be used.

//////////

SEND Direct Access Files - by T. W. Lanzatella

Kevin Matthews' new, rewritten version of DSP provides a nice usability enhancement in that users are now able to directly ROUTE a direct access permanent file. Program DSP will now perform the copy automatically. Few people will argue that this is not a pleasant change. In keeping with this spirit, I propose that this same feature be installed into SEND.

//////////

System Time Deadstart Dumps - by K. C. Matthews

I am currently trying to recycle deadstart dump tapes weekly, so that there will always be an ample supply. I am saving dumps of unsolved problems on a backup tape before recycling. There is currently no record of which dumps taken during system time should be saved. I would like to suggest the following procedure. If a Deadstart Dump tape is used on system time, and if you wish to save it for a while, please make an entry in the front of the NOS System Log in the computer room. Page 1 of the log will have a sheet of paper for this purpose. The DD tape number, the date, and your initials will be sufficient information. And of course, remember to place the tape in the "not to be used" half of the rack.

//////////

Remove PACKMS - by D. Nelson

PACKMS is a program used to eliminate unused records from a fortran structured random access file.

The problem is that this file structure has changed so that PACKMS no longer works. Rather than rewrite PACKMS, I propose that it be removed as a control statement for the following reasons:

1. The index key type (name/number) is only flagged in the master index for a given file. Since sub-indices need not be of the same type and due to the index structure of the two index key types, it is not possible to determine a sub-index type with 100% accuracy. Therefore, PACKMS would have to be written for the general case where all sub-indices are of the same type as the master index.

2. The user can accomplish the same results using the fortran subroutines OPENMS, READMS, WRITMS, STINDX, and CLOSEMS. Any user whose file does not fit the general form would have to use these anyway. The user is expected to keep track of the index key types of the sub-indices.

If complete removal is not acceptable, an alternative would be to make the current version of PACKMS Fetch-type.

//////////

Master User Access to Subordinate Users' Tapes - by A. B. Hastings

I propose that TAPES and RESEX be changed to allow master users (user numbers containing asterisks) implicit read-only access to tapes belonging to their subordinate users (user numbers matching the master user number except in positions containing asterisks). The master user would be able to AUDIT a subordinate user's account and obtain the tape password, but need not specify it to access the tape, just as is done now with permanent files. This change has been requested by users.

//////////

Maintainence - by T. J. Hoffmann

I would like to make the following two changes to the maintenance subsystem.

1. Currently, a 1SJ overlay goes through vast quantities of work to schedule a control point and CM, it also ignores IDLE., and schedules maintenance jobs anytime. I would like to move the scheduling to 1SP (where TRN and UQM are scheduled), and use the 1DS mechanism to schedule a job (ala X.CCCCCCC. and DELAYQ). This cleans up 1SJ, and removes the subverting of the scheduling mechanism.

2. Also, I'd like to raise the priority of the maintenance jobs from PR02 to PR30. This puts them in contention with all other jobs. This primarily affects the ECS tests which request a large amount of ECS, and with a low priority, can sit with the ECS tied up for extended periods of time.

//////////

RELOAD Enhancements - by P. Thompson

I propose to add three options to RELOAD. The first is an INQ option which allows the user to check on which of his files were requested to be reloaded. The second is a DEL option which allows a user to delete (cancel) all such requests. The third is a MI option which enables a user to reload files from one machine which were archived on another. Thus, RELOAD (INQ) would produce a list of files requested and RELOAD(DEL,MI=72) would delete all requests for that user on the Cyber 172.

//////////

Assorted Proposals - by W. T. Sackett

1. I propose that LDR issue ZLSY messages whenever overlays are loaded from the system.

2. I propose that the monitor function DTKM hang if a preserved file track is being dropped and it is not the first track in the track chain which is being dropped.

//////////

Dayfile Information When Dayfile is Too Large - by Mike Frisch

While converting a group of 7 track tapes to 9 track tapes, I ran into a problem when a large number of dayfile messages from tape errors were issued for some of my SUBMIT jobs. The systems group installed an excellent feature to automatically append the dayfile to the user's permanent file DAYFILE. However, if appending the dayfile would cause the permanent file to grow larger than an indirect access file, the new dayfile is not appended and the user might not get any dayfile information at all on the submitted job.

I propose that when the dayfile is too long to be appended to file DAYFILE and there are at least two PRU's available on DAYFILE before an attempt is made to append, that the first PRU of the user's dayfile be appended to DAYFILE and then a message appended: ** DAYFILE TOO LONG ** TRUNCATED **. It may even be useful to overwrite the last two PRU's of DAYFILE for these PRU's if there is not enough room, but this is perhaps less desirable. I usually purge DAYFILE before submitting my jobs, thus ensuring space for the two PRU's.

//////////

Control Statement Preprocessor - by Mike Frisch

When running a SUBMIT job, an error in a control statement could cost the user a lot of computer time, real time and money. I propose that a control statement preprocessor be written so that SUBMIT jobs can be checked for simple one-statement errors. For example, illegal file names or missing periods or missing right parentheses could be easily detected. The preprocessor would not have to check for multi-statement errors such as a non-loader statement appearing in a loader sequence, although errors of this type could be detected.

The preprocessor could be made even more sophisticated if equivalent code from each control statement processor were added to it. For example, if MNF's control statement processor were extracted from the compiler, it could do a more extensive pre-check of the MNF control statement.

//////////

CALLPRG Enhancements - by A. Swanson and J. Larson

We propose two enhancements to CALLPRG:

1. Allow a NODROP parameter (ND) to be specified in a CALLPRG index entry. ND = filelist would NODROP the files specified and which were made local by a CALLPRG activity. (The default would be all such files.)

2. Invent a new symbol (e.g. $) for use with the UN = and PN - parameters and which would denote the user number and pack currently assigned irrespective of any CALLPRG default.

The need for (1) is apparent when one uses CALLPRG to FETCH a user library (UL =) and subsequently and routinely wishes to NODROP it, or when a FUTURE product is similarly desired. Presently a NODROP card must be issued without fail to conveniently maintain them across OLD, NEW, etc. operations.

The utility of (2) becomes evident when one considers the use of a common CALLPRG index on several related usernumbers. When a common utility is kept on a library UN, for instance, a user would be able to create a permanent file of data and have it automatically accessed when the utility was called. An example of such a CALLPRG index entry would be:

```
WHAMO, UN = ABC1234, PN = SHA, IA = WHAMO/UTILITY,
UN = $, PN = $, IA = DATA.
```

A user currently operating under UN = XYZ1111, PN = 0, who executed X, WHAMO, would cause UTILITY to be obtained from UN = ABC1234, PN = SHA and DATA from UN = XYZ1111, PN = 0.

Both proposals are prompted by considerations of user convenience and external simplicity.

## SYSTEM MAINTENANCE: People and Procedures

Last Week's Systems Group Meeting - by T. W. Lanzatella

The following proposals were discussed.

1. John Vogel's proposal suggesting that the PURGE command respond with a message indicating what was purged was discussed and approved (see DSN 6,4 p. 33). John had previously stipulated that his proposal not be discussed until the long awaited appearance of Andy Mickel's article POLISHING OUR SYSTEM (DSN 6,10 p. 74). We decided that PURGE would not produce a long report but would send a single line dayfile message for each file purged. Users who purged several files would only see the message for the last file unless they looked at their dayfile.

2. Phil Kachelmyer's proposal to enhance CCL so that it could check for a directory at the end of procedure files was rejected (see DSN 6,10 p. 70). We decided that such an option would usually be inefficient, hard to document and too complex. Steve Reisman suggested that random access to procedure files could be effected with a procedure file which used GTR to get the desired record.

3. Marisa Riviere's two-part proposal dealing with the linkage of PSR509 COMPASS with FTN5 and the use of the "user fields" in COMPCOM was discussed at length and approved (see DSN 6, 10 p. 70).

4. Marisa's proposal suggesting that OPT=0 and OPT=1 be the second defaults for FTN5 and FTN respectively was defeated. We decided instead that they should stay at their current values (2 and 2). We also agreed with Marisa that the change in default value of OPT (2) should be made with the initial installation of FTN5 (see DSN 6,10 p. 71).

5. Mike Frisch's proposal to add a U option to LIBEDIT which would cause LIBEDIT to call LIBGEN was rejected (see DSN 6,10 p. 71). We suggested that this could be done easily with a procedure file.

6. Mike Frisch's proposal to have CALLPRG search the user's catalog was rejected (see DSN 6,10 p. 72). We concluded that the mechanism was too complex and too difficult to document. It was also pointed out that there could be no privacy for the callprg index stored under a number used by many students, i.e., anyone could purge the index.

Larry discussed the arrival of an additional 885 disk on 6 June and a 7155 controller on 20 June.

Andy Mickel led a discussion of his article "Polishing Our System" (see DSN 6,10 p. 74). After a brief recap of the article, Andy requested that the Systems Group pick-up on some of the suggestions and possibly convene a System Strategy Committee to discuss what was feasible. This met with mixed approval. Don Mears added several opinions:

1. Don would rather not change the system at all than add a lot of trifling changes.

2. Don felt that we should instead work on the areas of greatest deficiency like:
   a. insufficient control over jobs in the system;
   b. lacking communications facilities;
   c. insufficient ASCII support.

//////////

Callprg and Library Tape News - by M. Riviere

One June 14, the following changes will take place on the Cyber Library Tapes.

COPYCH will be replaced with a new version provided by Andrew Hastings, as was previously announced.

MSUIO, M77LIB, MINNLIB, MNFIOL, FN4IOL, M77IOL and IMSL will be replaced with new versions provided by M. Frisch. Most of the modifications made to these libraries are described by Michael in the UCC June Newsletter. Some other additional modifications consist mainly in the addition of routines from SYSLIB into MSUIO and the inclusion of entry points for FTN arithmetic routines on M77LIB.

COPYCH, MSUIO, M77LIB, MINNLIB, MNFIOL, M77IOL and IMSL will be updated on all computers. FT4IOL will only be updated on the Cyber 74 and 172 since it is used only on those two machines.

In addition, starting June 14, the Cyber 172 and 74 Library Tapes will be written at GE density instead of PE. If for any reason we should need to go back to a previous tape this should be taken into consideration.

Concerning Callprg modifications, on June 14, Steve Reisman will be modifying the Cyber 172/74 Callprg index in order to update the entry for SYMPL to use the level 509 version of the compiler.

On the same date I will also be updating the Callprg index on the Cyber 172/74 in order to make FTN5 a control card callable product. At the same time, I will remove the FETCH type entry that is used now for FTN5. The version of FTN5 to be used after June 14 has the default optimization value set to 2. This version of FTN5 is, as is the one now available as FETCH type, a level 509 version. A version of COMPASS, also level 509, will be retrived with the compiler, to be used by jobs with mixed Fortran and Compass code. A new updated version of the FTN4 to FTN5 conversion routines, F45, also level 509, will be available on June 14.

The next set of Callprg and Library Tapes modifications will be taking place on July 1. Modifications for that date should be submitted no later than noon June 19.

//////////

Errata to "Polishing Our System" - by Andy Mickel

Below are corrections to the article "Polishing Our System", DSN 20 May 1980, Vol. 6, No. 10, pp. 74-81.

1. Page 76, first paragraph, line 5.
   insert the word "not" after "is to try".

2. Page 76, second paragraph, line 4.
   delete both periods in this line.

3. Page 77, first paragraph, line 5.
   change "the structure" to "structure".

4. page 77, last paragraph is incorrect
   DC=SC (rescind prior routing) is
   default, not DC=1N.

5. page 78 under point 12) part b), line 3.
   change "whould" to "should".

6. page 82, Appendix C, second point 1) lines 1 & 2:
   change "one year paid for" to "one year from being paid for".

//////////

Letters to the Editor on Polishing Our System

I was delighted when the trickle of hallway discussion surrounding Andy's article turned into an avalanche of written responses. All of these responses will be presented here except when the response is in the form of an article such as Don Mears' response which follows this article.

From Lincoln Fetcher:

Hear! Hear! The important concepts are "user friendly," "simple to use," "human oriented," and I'm in full agreement with Andy about the need to strive for those attributes in our systems.

To me, it's the end user that should be the most important consideration when writing software, including the programmer him/herself. This may include compromising the "efficiency" of the computer, but so what? These machines were invented by, designed by, and intended for use by people - FOR PEOPLE! If we don't look at the purpose of computing in those terms, why do it?

Our competition - especially in the area of micros - is making great leaps toward implementing many of the attributes described in Andy's article. If we are to be competitive, especially with micros and other computers of all sizes springing up around campus, we must make our machines more people oriented. And we should be able to afford it - especially in terms of machine efficiency. It is now cheap enough to buy more hardware to accomodate the user-oriented software, so the argument of need for machine efficiency is fast dwindling.

Time spent developing this software would certaining be well spent. Just think: if we had good, informative error messages, a consistent parameter system, a simple permanent file and tape processing command system, how much consulting time would we save? Lots, I'm sure of it.

From John Mulhern:

It was reassuring to read "Polishing Our System" in DSN 6,10 p. 74. Apparently others besides myself have quarrel with some of the unpleasant quirks of our operating system. I emphasize particularly with Sara's suggestions 1, 2 and 3; Andy's suggestions 5 and 15; Jeff's suggestion 17 (!); and Bill's suggestion 20. Many of the suggestions centered around operating system responses; I personally find many system messages to be obscure or even meaningless. In a number of cases they need merely to be rewritten in English. Many other comments involved the character set mess. In as much as all my work deals with text processing I am certainly in favor of any changes which would make full-ASCII text files less outcast in the operating system.

How can we move in a consistent fashion to a more user-friendly system? What are the trade-offs in terms of additional mods to the system? I would be glad to submit a list of suggestions of my own concerning ways in which I think our system might be "polished" from the user's point of view (and omitting such pipe dreams as a structured file system, a single, full-ASCII character set, a screen editor, etc.). Is this desirable? Is there widespread interest in actually making some reasonable changes toward a more user-friendly system?

From Sara Graffunder:

I want to add my name to those who support Andy Mickel's criteria for modifying software. As a non-expert user since 1973, I have made more efforts to learn the operating system than I believe should be required of the average user. Still, I run into error messages that trip me up and problems for which the "sensible" solution doesn't seem to work. Users who don't work at UCC must have a lot more difficulty than I do.

I'd be glad to add more points to a laundry list sometime, but I don't think that the merits of any particular item are worth arguing over now. I'd just like to have some of the external matters like uninformative error messages, inconsistent parameters, and mind-cluttering redundancy given consideration equal to that given to internal matters like optimizing software. I don't expect this or any other system to be as easy to learn as how to run a microwave oven is. However, in fine-tuning this system and in selecting new ones we should look at how much alien information a user must assimilate in order to get a job done properly, then minimize it where we reasonably can.

How Andy's principles might relate to documentation:

I find CDC's documentation difficult to understand because it presents the detail in fits and starts rather than in a more helpful way. I think this is because of the amount of detail one must know to use even portions of the system. Our User's Manual explains the general concepts very well, but it took so long to write in part because of the difficulty of explaining those concepts simply, without misleading people. Much of the documentation we produce is aimed, not at explaining added-on features, but at "justifying" what's already there by explaining it simply enough for users. Writing that kind of documentation takes time, more time than it ought to.

Take the Guide to Magnetic Tape Usage as an example. Even supposing that UCC had added no software to CDC's tape software, the Tape Guide would have to exist because of the lack of a universal standard tape format and because of the NOS Manual's failure to describe what tape is, the variety of formats and how they originate, and how to get what you want onto or off of a magnetic tape. The technical detail is there, sure, but the average professor of Biometry (substitute any user you like) can't understand it.

At UCC, we can't solve the problem with diversity of tape formats. That's why I chose it as an example. With other problems, however, we could narrow the gap between the technical explanation and the understandable one by improving our software. Then we wouldn't have to write so much documentation.

From Thea Hodge:

I would like to comment on Andy's article (DSN 6,10, pp. 74-82) on Polishing Our System. That article and its appendices exemplify the kind of thinking that has made me say at user services meetings all over the country that Minnesota has one of the best service oriented systems groups in the U.S.! There is more to be done; there always will be improvements needed. But we start from a good base with "user-friendly" and interested systems staff members. In fact, I will extend that description to include all UCC staff members. Everyone here has taken to heart the admonition that service is where the action is.

Since there are, however, ways in which the software is less user-oriented than our people, I suggest as a first step that we set up priorities for the categories of items which would smooth or "polish" or "tune" the system. All of Andy's suggestions are worthy. Some needed items of change may affect more users than others, some may have more side-effects for the user, some ease a user's path more than others would. I am more concerned about awkwardnesses in XEDIT than I am about difficulties in using obscure features of CCL. I am far more bothered that the growing number of our users with access to CRT's do not have a screen editor than I am by complaints about one of our unsupported languages. In the light of Larry's questions in Appendix C, however, I suggest 1) that we make the assumption that we have one year for sure to use our current system and 2) that we very quickly set up a team of not more than 3 people to establish a hierarchical list of fine-tuning projects with very short deadlines — a tight deadline to have the list ready and a short deadline for each item on the list.

Obviously, if we decide that we are justified in planning for only N years ahead, some items will not make the list at all, even if they meet my criterion of affecting the most users most of the time. The items that make it to the top of the list should meet that criterion plus the requirement that it be do-able in a given period of time with the available staff.

There is one more factor I would like to mention. One of our sometime-users has said often, "I don't care how awkward your system is to use as long as you tell me about it." While fine-tuning the system we must continue to document every step so that we carry the user along with us. "I don't like surprises," says that user.

From Rich Franta:

I agree with many of the comments made by ABM in his DSN "book". As a matter of fact, I have had <u>many</u> calls asking what does EOI encountered or decimal base used (settl) mean. There are many messages that could be made <u>human</u> or explicit.

From Alex Riley (University of Leicester):

Your Systems Newsletter Vol. 6, No. 10 raises many interesting points. Like yourselves we are very concerned about NOS useability and I started to do some work in this area a few months ago. We hope to resurrect the project when we upgrade to Release 5.

Your contributors raised the point that in many cases one can become quite blind, through familiarity, to quite silly inconsistencies (one example given was the use of "/"). Our changes have centered around Telex and the concept of subsystems. In a nutshell, we intend to throw out the whole subsystem idea. The results may be summarised as follows:

1. Users no longer need to remember two meanings for BASIC or LIBRARY.

2. Most of the Telex command table can be scrapped since one effectively runs in BATCH the whole time.

3. The RUN command is handled as if one were in the present BASIC subsystem.

4. Other compilers (e.g. FTN,MNF,PASCAL) are added to the Telex command table and processed by Telex in such a way that:

   ```
               MNF
   ```

   will be converted to MNF,I=primary file,K.

   ```
               PASCAL
   ```

   will be converted to PASCAL,I=primary file/G+,L-.

5. BASIC becomes a normal compiler call, defaulting to primary file for input.

6. We have also removed esoteric commands (e.g. MONITOR) and time-delay code. The end result saves 1000B words from Telex as well as memory for 1RI,1RO and 1TA.

The modifications are still not fully tested and are based around Level 452. Any comments you have will be welcomed and may influence the way we implement these changes.

//////////

Reverse Polish (Part 1) - by D. W. Mears

This discussion is intended to be a response to the Andy Mickel DSN article "Polishing Our System" for the DSN Vol. 6, No. 10. Andy's basic point is that in order to combat the flight of users to new non-UCC computer systems on campus we should not make any drastic changes, but instead fine tune and polish our existing system to make it more "human oriented" and "user friendly." Furthermore, when we do this polishing we should not be afraid to improve things just because the old (bad) things have "historical significance and are perpetuated by the familiar rationale of upward compatibility with the past." He goes on to give some general criteria for ease of use and some specific suggestions for improvement of the system.

I disagree with most of the article. I do not think many people will switch to non-UCC systems' due to our systems lack of "user friendliness." Users will switch if other systems are capable of providing services we cannot provide, if other systems provide higher reliability (both in terms of system up time and products which work as documented) and if other systems can provide services of a lower cost. Polishing the system will not increase our capabilities, it will worsen our reliability (for reasons I will discuss later) and it will not lower our cost. The most it can do is make our present users slightly happier when they use our system and slightly aggravated when they attempt to use a stock CDC system.

A central concern should be providing new capabilities and services, not modifying access to existing services to make them more "human oriented." Expending a lot of effort changing things to be more "friendly" now would be analogous to working on similar improvements on our MOMS operating system in 1974. In 1974 the need was for a new service - timesharing. It has been that new service which has allowed the computer center to continue to grow. Today, we continue to need new services and capabilities to survice the attack of micro's and mini's; services such as more graphics device support (e.g. different size plotters, color plotters, high speed interactive graphics), page printers, a high speed communications network for connecting micros and terminals to our computer, full ASCII support in language processors and editors (preferably screen editors), links to other large computers which may appear on campus, and more user control over queue files and executing jobs.

The argument could be made that we should add new services and capabilities <u>and</u> change things to be more "human oriented." The problem is that this polishing of the system contributes nothing to solving the need for new services and it uses dollars and manpower which could be better used elsewhere. Also, there is a conflict between these two goals. Polishing implies keeping things essentially the same (Andy's conclusion suggests we should not even go to new releases of NOS) and making many small changes; while adding new capabilities implies making some large changes and very few small changes to the system.

I have even stronger objections to system "polishing" from a reliability standpoint. Even if only messages are changed, some existing documentation will become incorrect, and (no matter how self explanatory the new message is) some users will become confused. An example is: when the obscure message DEMAND INSTALLATION ERROR was changed to INSUFFICIENT RESOURCES ON SYSTEM a user called me on the Help-line and wanted to know what the "INSUFFICIENT..." message meant because it was not described in the old tapeuse writeup he had and he knew that it could not mean he requested too many resources because he expected to see "DEMAND..." in that case. This was a reasonably intelligent user who was confused by the improvement. We have many users who are much less familar with the system (e.g. clerk or data entry type people) who simply log in and follow a script provided by someone else. These users are even more apt to be confused and annoyed by our improving of messages and would consider the changes as a lack of apparent reliability in that they cannot rely on the system behaving the same way from day to day.

Andy suggests improving some programs so that they work differently. I argue that changing the way MNF, OLD, NEW, and ROUTE work would cause a lot of users' jobs to stop working because no matter how much we advertise the impending change some people will not hear about it. These people will see that the job that used to work has stopped working and according to the manuals they have, everything they have done is correct. Even if UCC argues that these programs are really running correctly (we just changed the rules) the user who has been burned will regard the change as creating a lack of apparent reliability.

Reliability can also be lost as an accidental side effect of these polishing mods. Polishing the system implies making small changes to a lot of programs. My experience is that mods like these are most likely to cause side effects. The reason is that when you make large changes to a program you are forced to study the whole program, but when you make small changes you can usually get by looking only at the area near the change. It seems to me that there is a real potential for installing new bugs along with the polish. One error in the system can cancel out a lot of the goodwill the polish is suppose to create.

Up to this point I have only discussed why I think we don't need polish to hold users at UCC. But there is a more basic issue. That is, what changes make the system more human oriented and user friendly? This is a very subjective issue. Should the system be oriented to the novice user or the experienced user? How do we decide what is "user friendly" and what is not? For most of the "improvements" suggested in Andy's article there exist several counter arguments why the "improvements" should not be made. Should we take a vote on each improvement and then make the change if 75% of UCC (or of users surveyed) think the change is good and 25% think the change is worthless or more confusing? I argue that when the value of an improvement which adds no new capabilities and fixes no errors is debatable, the improvement should not be made.

The following are my counter arguments to some of the improvements suggested.

Eliminate the need for EC=A9 on ROUTE.

The solution described would probably never work. The problem is that it is very difficult to guarantee that all full-ASCII files have the attribute set. Every program which copies data between files would have to set the ASCII attribute on the file it is writing if the file being read has the attribute set. These would include XEDIT, EDIT, COPYB, COPYU, PACK, and RESEQ, to name a few. I do not think we could ever get it working right everywhere. (We can't even get "AS=ON" to work correctly in PFILES now.) The result is that sometimes ROUTE will assume EC=A9 when it is supposed to and sometimes (when the wrong copy routines are used) it will not. This partial solution would be even more confusing.

MNF should not execute the binary by default.

When MNF was modernized (from compiling to core where immediate execution was a reasonable default to writing relocatable binaries) the question became: which causes more confusion, inconsistent parameters between compilers or a change of existing parameters which would invalidate most MNF job decks and much documentation? Viewed another way: do we change things to reduce the confusion of the new user or do we leave things alone to avoid confusing the experienced user.

OLD and NEW should issue a warning or should not CLEAR local files.

OLD and NEW are part of a set of de facto standard timesharing commands common to many timesharing systems. I believe that OLD and NEW drop all the local files on all these systems. The reason OLD and NEW automatically drop files is that this is a logical side effect. When OLD and NEW are used, the user usually is starting something completely different and wants the local files returned. This is easily verified by checking the system dayfile to see how often the ND parameter is used to defeat the automatic dropping of local files. I have been unable to find any basis for the contention that there is a lack of documentation warning that OLD and NEW clear local files. All of the descriptions of OLD and NEW I can find warn of this side effect. Because of this documentation, because this is a logical side effect, and because OLD and NEW are so heavily used, I have to believe all users either already know or learn very quickly that OLD and NEW drop files. For these reasons I argue that even a warning message is unnecessary.

Eliminate PACK after text mode.

This was proposed and accepted. It will be done as soon as someone gets the time.

Make CSET,ASCII default.

Most users still run programs on our system and this change would make it easy to enter programs and procedure files in lower case which would not work if the users forget to enter CSET,NORMAL (a classic example of an setting-up an accident to happen). In addition, in CSET,ASCII it is possible to enter programs which look perfect but get compilation errors because of the different translation for circumflex. I suggest that those users who do a lot of text processing and want CSET,ASCII to be default use SETVAL to set TC=ASCII to select ASCII at login time.

Minimize differences between direct and indirect.

CDC's philosophy is that since direct and indirect files have very different characteristics it is important for the user to be aware of which type of file is being used. By hiding the actual type of the file we blur the distinction between direct and indirect. The result is increased confusion. A good example of this is the job which does an ACQUIRE, modifies the acquired file, and then does a RETAIN. This continues to work until the file gets big enough to cause RETAIN to change the file to direct access. At which point the job bombs off with an attempted write on a read-only file. The trouble is that the user must worry about a new set of problems when he starts using direct access files. These problems appear spontaneously when the system automatically gives the user a direct access file.

Change "ILLEGAL COMMAND"...

I agree that TELEX should put out something other than ILLEGAL COMMAND when the problem is that the user is in the wrong subsystem or when the user enters a non-secondary command when his job is running. But, I think we should keep "ILLEGAL COMMAND" for the case where an illegal command was entered. ILLEGAL COMMAND is a common message that many interactive programs issue in this situation. I do not think that changing the message to "NOT AN INTERACTIVE COMMAND" makes the meaning more clear or understandable.

I have counter arguments for the remaining 38 suggestions, but I am running out of time and paper to respond to the other suggestions here. I will probably finish my response in another article in the next DSN.

The point I am trying to make is that I don't think we should be fixing programs which are not broken - especially when our fixes add no new capabilities and when the basis for making the changes is as subjective and nebulous as "USER FRIENDLINESS". Polishing the system will not solve the real problems and may create new problems.

//////////

Cyber 74/172 Deadstart Dump Analysis from Friday, 16 May through Thursday, 5 June - by K. C. Matthews

Sunday, 18 May

22:10 Cyber 74
Diagnostic CU1 had been failing since 19:40. The machine was given to the engineers to repair a divide problem.

Monday, 19 May

08:46(DD2002) Cyber 74
The system died. Analysis showed that bit 39 was set in one bank in several low core words. No hardware problem was ever discovered.

Tuesday, 20 May

04:25(DD2003) Cyber 172
This was a very strange dump. Both exchange packages were simply junk, and a real exchange package was in low memory. Only maintenance jobs were running - the operators shutting down the system.

Wednesday, 21 May

00:35(DD2020) Cyber 74
The Cyber 74 displays suddenly looked like junk. No deadstart attempt worked. The engineers were called in to fix a solid problem.

Monday, 26 May

16:00 Cyber 74
The Cyber 74 did not come up for production on time after the Memorial Day Holiday. A solid error in some rounded arithmetic had been detected by the diagnostic FST. Don Mears isolated a case which always failed. We were up for production at 19:38, but on the next day further problems were found.

Wednesday, 28 May

09:20 Cyber 74
Problems with floating subtracts again forced the system down. It was up again at 11:30.

13:18 Cyber 74
CPU errors were again detected. A card used in the previous repair turned out to be bad. The Cyber 74 was up again at 14:58.

13:33 Cyber 172
1SJ - the job schedular - seemed to be stuck and not scheduling. We attempted to fix this problem by changing a memory byte, but it didn't work. No dump was taken because we believed the spurt of activity after the memory change had altered things too much from the original problem state. A level 3 deadstart was performed.

16:24(DD2004) Cyber 172
PP program 1RI hung. There were several errors on the 885 on the 172, which had been introduced in the system over the previous weekend. Several attempts at deadstart were made before the 885 could be accessed. And then, it seemed to be behaving perfectly.

04:00(DD2005) Cyber 74
Attempts to use the DSD "O" display (used for TRANEX) blanked the display console. The O display was disabled at UCC, but has been re-introduced, and thus this problem is solved.

Thursday, 29 May

13:55 Cyber 172
The 885 suddenly was full of errors, as it had been once on 28 May. Again, several deadstart attempts were required to get the system up. Permanent files were removed from the disk on the weekend following this, since we didn't trust the 885 on the 172.

14:45 Both Machines
A brief power failure downed both machines. Level 3 deadstarts worked.

Friday, 30 May

01:06(DD2006) Cyber 172
Solid errors on the 885 again. And again, they disappeared after deadstart.

20:00 Both Machines

Both machines failed after another short power interruption. A level 3 worked on the 74, but a level 0 was required on the 172. The level 0 failed because there was not enough room in ECS, so both machines had to be deadstarted to initialize the ECS equipment. There have been several cases lately where one machine could not get enough ECS to continue its level 0 deadstart. This is certainly a serious bug that we should try to solve.

//////////

Cyber 170-720 Deadstart Dump Analysis (4/-1-6/8) - by R. W. Williams

| Date | Description | Tape |
|---|---|---|
| 800421 | A mysterious power surge caused only the Cyber 720 equipment to hang up. | N.A. |
| 800423 | A motor generator failure caused the system to come up late. | N.A. |
| 800428 | A software correction was in error and caused the system to abort. | Fixed |
| 800506 | LFM hung due to a problem with some corrective code. | Fixed |
| 800510 | A power failure caused the system to go down. | N.A. |
| 800514 | 1TA hung when bad data was passed to it by Telex. | DDT-10 |
| 800516 | PFM hung because two files pointed to the same disk area. | DDT-01 |
| 800519 | The dewpoint recorder sensed a humidity range error. | Fixed |
| 800523 | The dewpoint recorder sensed a humidity range error twice on this day. The range settings were incorrect and have been reset. | Fixed |
| 800529 | A power failure caused the system to go down. | N.A. |

//////////

TELEX and TELEX PDP11 Crash Analysis (5/5 to 6/5) - by D. W. Mears

There were no TELEX crashes during this period.