What if we don't do this? File - McHale pap.

RECEIVED
MAR 2 1972
J. H. SALTZER

TO: C. T. Clingen
F. J. Corbató
R. C. Daley
R. J. Feiertag
J. W. Gintell
N. I. Morris
R. A. Roach
J. H. Saltzer
S. H. Webber
V. L. Voydock

FROM: Dave Reed

DATE: March 1, 1972

SUBJECT: A Plan to Convert Error Tables to New Format

Stage 1

In this stage, the standard error table is updated to be in the new format, with the special segment number in its pseudo-pointer error codes. This requires changes to error_table_compiler and com_err_. Error_table_ compiler will be given an option to produce either the system type error table, or a user error table. It will comment that user error tables do not work if a person tries to compile an error table without the option. Com_err_ should be rewritten, but it will become a very simple procedure. This stage will take about 1/2 to 1 man week.

Stage 2

Here we must add trap-at-first-reference to the system. Changes are needed to link_snap, default_error_handler_, linkage_error_, and ALM. The change to alm should be made along with the other pending changes to ALM. Bob Mabee says it will take a few hours to design and implement the needed changed to ALM. The changes to link_snap and default error handler are trivial. Linkage_error_ is slightly more complicated, but not more

than 1/2 a week of work.

Stage 3

This stage opens the facility to users. The use of error_table_ compiler should be documented in the MPM and help files and is only a documentation task. After this stage, which might take a day, plus distribution time, the facility will be completely useable for users.

Stage 4

As a final task, the binder is to be upgraded to handle the trap-at-first-reference feature, but there is no reason that this cannot be deferred until the binder is upgraded to deal with the new object segment format. There are no particular problems here, so this stage should be simple, as long as the binder is being changed, anyway. There is, in fact, no real reason why this stage cannot be deferred indefinitely.