Research Report 1286

CONVERTING PLANIT LESSONS TO ENHANCED FORMAT

Charles H. Frye
The Northwest Regional Educational Laboratory

MANPOWER AND EDUCATIONAL SYSTEMS TECHNICAL AREA

U. S. Army
Research Institute for the Behavioral and Social Sciences

September 1978

U. S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the

Deputy Chief of Staff for Personnel

JOSEPH ZEIDNER
Technical Director

L. NEALE COSBY
Colonel, IN
Commander

Research accomplished under contract
to the Department of the Army

Northwest Regional Educational Laboratory

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Research Report 1286	2. GOVT ACCESSION NO. AD-A110 541	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* CONVERTING PLANIT LESSONS TO ENHANCED FORMAT		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Charles H. Frye		8. CONTRACT OR GRANT NUMBER(s) DAHC19-78-C-0022
9. PERFORMING ORGANIZATION NAME AND ADDRESS Northwest Regional Educational Laboratory, 710 S.W. Second Avenue, Portland, Oregon 97204		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2Q762722A764
11. CONTROLLING OFFICE NAME AND ADDRESS Army Research Institute for the Behavioral and Social Sciences, 5001 Eisenhower Avenue Alexandria, VA 22333		12. REPORT DATE September 1978 13. NUMBER OF PAGES 59
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

PLANIT
TACFIRE

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This research report relates the experience of converting 11,000 lines of Fire Mission PLANIT lesson scenario from conventional form to the enhanced PLANIT FORMAT. Authoring difficulties are discussed along with proposed solutions. Among the problems discussed is the proper sequencing of operations, particularly in the areas of input and output.

This report concludes with several recommendations aimed at improving the human factors elements in authoring and in debugging the newly authored lessons.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

Research Report 1286

CONVERTING PLANIT LESSONS TO ENHANCED FORMAT

Charles H. Frye
The Northwest Regional Educational Laboratory

Submitted by:
James D. Baker, Chief
MANPOWER AND EDUCATIONAL SYSTEMS TECHNICAL AREA

Approved by:
Edgar M. Johnson, Director
ORGANIZATIONS AND SYSTEMS
RESEARCH LABORATORY

U.S. ARMY RESEARCH INSTITUTE FOR THE BEHAVIORAL AND SOCIAL SCIENCES
5001 Eisenhower Avenue, Alexandria, Virginia 22333

Office, Deputy Chief of Staff for Personnel
Department of the Army

September 1978

Army Project Number
2Q762722A764

Embedded Training

ARI Research Reports and Technical Reports are intended for sponsors of R&D tasks and for other research and military agencies. Any findings ready for implementation at the time of publication are presented in the last part of the Brief. Upon completion of a major phase of the task, formal recommendations for official action normally are conveyed to appropriate military agencies by briefing or Disposition Form.

FOREWORD

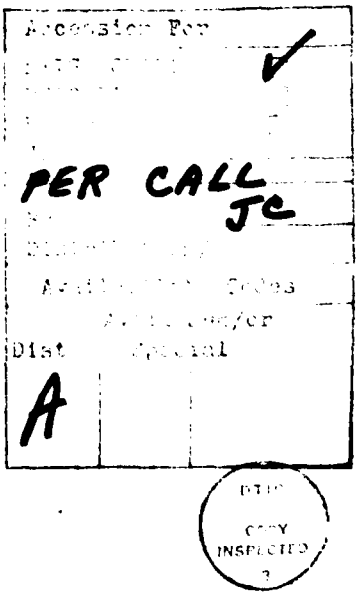
The Unites States Army Research Institute for the Behavioral and Social Sciences (ARI) engaged the Northwest Regional Educational Laboratory under contract DAHC19-78-C-0044 to convert a PLANIT lesson scenario from conventional form which could be executed under the PLANIT/TACFIRE NORMAL mode to the Enhanced PLANIT format.

PLANIT, or Programming Language for Interactive Teaching, is a portable, time-sharing computer software system and authoring language used for the preparation and delivery of computer-assisted instruction (CAI).

Charles Frye spearheaded the effort of converting approximately 11,000 lines into the Enhanced PLANIT format.

This Research Report is a comprehensive and specific study into the modification of the PLANIT system and is a complement to Research Report 1285, "Extension of Computer-Assisted Team Training through Coordinated Lesson Scenario" by Charles Frye.

JOSEPH ZEIDNER
Technical Director



CONVERTING PLANIT LESSONS TO ENHANCED FORMAT

BRIEF

This Research Report attempts to relate the experiences and insights gained from converting approximately 11,000 lines of Fire Mission PLANIT lesson scenario from conventional form suitable for execution in the PLANIT/TACFIRE NORMAL mode to the Enhanced PLANIT format. Many authoring techniques are discussed along with the "do's" and "don'ts" which were learned along the way.

Specific expected authoring difficulties are discussed along with proposed solutions that will make the resulting lessons manageable and convenient for the user. Among the more severe of the expected problems are those which require proper sequencing of operations, especially in the areas of input and output.

This report concludes with several recommendations which are aimed at improving the human factors elements both in authoring and in checking out the newly authored lessons. A final section of recommendations suggests a course of action that will lay the groundwork for the proposed changes.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

LIST OF FIGURES

CONVERTING PLANIT LESSONS TO ENHANCED FORMAT

INTRODUCTION

PLANIT (Programming Language for Interactive Teaching) is a portable, time-sharing computer software system and authoring language for the preparation and delivery of computer-assisted instruction (CAI). In addition to several commercial installations, PLANIT has been installed in two different ways on the ANGYK-12 computer, specifically for use in training TACFIRE system operators. The system was first installed on the TACFIRE system in a manner closely resembling commercial installations, where the TACFIRE terminals were made to function like conventional CRT computer terminals. Later, an enhanced version of PLANIT was made operational on the TACFIRE system which permitted the practice of normal TACFIRE terminal operations to be both elicited and monitored during the training sessions.

In the time after the initial installation and prior to the enhanced installation, a lesson series was authored which taught many of the principles of fire mission processing, using the TACFIRE hardware (and the PLANIT system) as a vehicle for delivering the instruction. Although the authoring was done with TACFIRE terminal hardware in mind, the lessons would execute just as well on commercial PLANIT installations since none of the special TACFIRE push-switch, indicator lights or special screen activity was subject to author control in that installation. The Fire Mission lesson series was fairly comprehensive, encompassing 11,000 lines of lesson scenario, including two course tests.

The enhanced version of PLANIT subsequently became operational on TACFIRE, and the task was undertaken to convert the Fire Mission course to the Enhanced PLANIT format. This was desirable for at least three reasons:

- To upgrade the Fire Mission course such that the trainees were required to practice TACFIRE operations rather than just learn about them.
- To chart the difficulty that authors of Enhanced PLANIT lessons would face and recommend ways that the difficulties could be ameliorated.

- To prepare authoring guidelines that would assist prospective authors surmount the difficulties.

Regarding the three points, above, the first has been accomplished and initial tests of the results have been encouraging, the second is the subject of this report, and the third will be the predominate subject matter of the forthcoming final report.

Thus, this report will document the findings during the conversion effort which will impact future authors who use the Enhanced PLANIT facility. It will describe the problems which were encountered and the solutions which were adopted. While most of the solutions had to do with authoring techniques, some required changes to the Enhanced PLANIT installation. Since resources for system changes were limited, not all of the desired improvements were implemented. Those yet to be made will appear in this document in the form of recommendations.

ASSUMPTIONS

There are two important assumptions which affect one's understanding and comprehension of this document:

1. It is assumed that the reader is already a proficient author of conventional PLANIT lessons.

2. It is assumed that the bulk of PLANIT lessons which are to be authored in the enhanced format will be composed and checked out on conventional PLANIT installations prior to being used on TACFIRE hardware.

In the first case, the lesson conversion problems are of such a nature to make the discussion unwieldy if PLANIT authoring proficiency cannot be assumed. PLANIT authoring documentation is readily available.

In the second case, TACFIRE computer time is not usually available in the amounts needed to author training scenarios. The ability to complete the authoring on commercial equipment provides such a resource advantage that it must be maintained, and even improved where possible. This does not rule out the option of authoring (or performing certain authoring functions) on the TACFIRE hardware as well. Of course, complete lesson compatibility must also be maintained between the two kinds of hardware.

It is also true that the TACFIRE terminals are not especially well suited to authoring tasks. Commercial terminals make the authoring task easier.

ABBREVIATIONS

Certain PLANIT and TACFIRE terms will be used so frequently that common abbreviations will be used. In order to avoid possible misunderstanding, these will be listed here.

PLANIT abbreviations:

- Q - PLANIT Question frame
- MC - Multiple Choice response format
- D - PLANIT Decision frame
- P - PLANIT Programming frame
- G2 - Group 2 of a PLANIT frame
- G3 - Group 3 of a PLANIT frame
- G4 - Group 4 of a PLANIT frame
- F: - PLANIT Feedback (short message) command prefix
- R: - PLANIT Repeat response (and short message) *command prefix*
- C: - PLANIT CALC line and Correct answer command prefix
- B: - PLANIT Branch command prefix
- $ - Carriage control end-of-line terminator (to supress carriage return/line feed and prompt characters)
- CALC - Calculation mode of PLANIT
- MIOP - Machine Input/Output Program which adapts PLANIT to operate on the target machine

TACFIRE abbreviations:

- ACC - Artillery Control Console (primary TACFIRE terminal)
- RD - Receive Display (upper screen on the ACC)
- CED - Compose/Edit Display (lower screen on the ACC)
- ELP - Electronic Line Printer (for hard-copy printout)
- SA - Switch Assembly (panel of push switches on the ACC to the left of the screens)

IMPLEMENTATION OF PLANIT ENHANCEMENTS

The reference section includes citations for comprehensive reference manuals for the enhancements package that was implemented with the TACFIRE PLANIT installation. However, since those documents deal only with the enhanced implementation on TACFIRE and not with commercial machine counterparts, some additional discussion of that implementation might be useful here.

First, it should be noted that the Enhanced PLANIT implementation on TACFIRE has two modes of operation, NORMAL and CONTROL. The NORMAL mode operates in virtually the same way as the earlier implementation on TACFIRE did, and that was deliberately designed to resemble commercial implementations as much as possible. Composing a response on the CED screen is a bit awkward, and the method chosen to simplify the process left only six lines on the screen to accomodate output messages. This is typically less than commercial terminals allow, but within this constraint, the authoring process for sending the information out and processing the reply is the same for the NORMAL mode (of the Enhanced PLANIT on TACFIRE) as for the typical commercial implementation. Such lessons are immediately transportable, with the most severe potential problem being the need to step through output messages, six lines at a time.

The Enhanced PLANIT implementation did not change the way PLANIT ran on TACFIRE; rather it added a CONTROL mode. The CONTROL mode provides a means of illuminating indicator lights and sensing switch actions on the SA in addition to accepting keyboard responses. It also permits assignment of output messages to either screen (RD or CED) or to the ELP (or both). The author includes a SPECIAL command directive in the lesson scenario to switch lesson execution from the NORMAL mode to the CONTROL mode (or vice versa), and only while in the CONTROL mode are the effects of the enhancements noticeable.

Next, it is important to the discussion to understand how the enhanced author directives are recognized by the computer during execution of the lesson. The PLANIT system source code was not changed in any way as a direct result of the enhancements implementation. The enhancements are completely contained in and performed by MIOP. There were many reasons for not wanting to change the PLANIT system code to implement the enhancements package, mostly having to do with the desire to maintain one machine independent version of PLANIT, thus avoiding costly maintenance and assuring compatibility of lessons across machines. Also, it was recognized at the outset that commercial installations would need to be used to prepare

PLANIT/TACFIRE lessons, even for the enhanced (CONTROL) mode of execution. Some changes were eventually made to the PLANIT source code which improved the enhancements facility, but these changes were also deemed beneficial to all PLANIT users alike. They will be discussed in more detail as solutions to specific problems.

Thus, the enhancements facility within the PLANIT/TACFIRE implementation constitutes a local addition which is contained entirely within the MIOP installation package (which is written locally in any case). In no way does it change the PLANIT system which interfaces with it. The same PLANIT is used for the TACFIRE installation as is used in every other installation. As new versions of PLANIT become available, they are mated with the present MIOP package and continue to run as before.

In order to implement the recognition of the enhancement author directives, the PLANIT/TACFIRE MIOP interprets the consequences of the PLANIT SPECIAL command call (which is provided for special local needs). One important use of that command causes MIOP to switch to the CONTROL mode. Once in the CONTROL mode, MIOP begins to monitor all of the text that the lesson causes PLANIT to send, through MIOP, to the terminal. However, the Enhancements Document directs the author to place certain directives in that text stream which PLANIT is sending to the terminal, and the occurrance of the characters $$$ at the beginning of a line of text, although meaningless to PLANIT, alerts MIOP that the line is not text but rather a list of one or more enhancements directives.

While the technical nature of this discussion may seem to go into detail unnecessarily, the understanding of this particular procedure helps to explain many of the authoring problems as well as providing the means for preparing lessons on commercial machines which lack most of the special TACFIRE terminal features. Notice that the MIOP's written for commercial installations will place no significance on the $$$ characters, but will simply allow them and the following command directives to print on the terminal along with the other text. The author will not actually see the effect of the directives on the commercial terminal, but the directives will be seen, printed on the terminal and one can then infer what would have taken place on TACFIRE.

MIOP receives no information from PLANIT regarding the lesson structure. It is simply a courier of message data between the central computer function of PLANIT and the terminals (and other peripherals). Therefore, enhancement command directives are not (and cannot be) associated with any part of the lesson structure (e.g. G2 of a Q frame); they are simply regarded as a part of any output text from the lesson, whether it comes

from G2 of a Q frame, the text portion of the F: or R: command, the text of a CALC PRINT or ALIGN command, an internal PLANIT error message, or any of several other means of sending text to the terminal. So long as the lesson is executing and a SPECIAL call has switched MIOP to the CONTROL mode, MIOP is examining the beginning of each new line of output text for the $$$ string of characters. In the case of TACFIRE, output lines on which command directives are discovered are not sent on to the terminal, but are interpreted and the prescribed actions are taken instead. These actions can direct subsequent text to a different screen (or the ELP), blank either screen, turn switch lights on (or off), transmit selected portions of the CED screen back to PLANIT for response processing, or release PLANIT from the CONTROL mode (switching it back to the NORMAL mode).

On the response side of the interface, PLANIT typically notifies MIOP that lesson execution will be suspended while the response is being typed on the terminal. When it has been completed, as signaled by the pressing of some kind of transmit (send, return, etc.) key, MIOP is to move the typed message to PLANIT and "raise a flag" that indicates the execution is ready to proceed. In the case of PLANIT/TACFIRE, the response may come from any of several input sources, including keyboards or a variety of push switches. Since the PLANIT response "read" command is only implicit within the language structure (i.e. occasioned by the first answer line in any G3), the enhanced version of MIOP uses a clever sequence of activities to transmit pertinent data at appropriate times. However, if the author does not completely understand the prescribed sequence, the lesson can easily be written such that the hoped-for response is not the one actually transmitted, and in consequence, the lesson will make some incorrect assessments of the student's performance. It can lead to a great deal of confusion, such as waiting for the student to respond when the student has no indication that a response is expected. (This can present a real impasse situation!)

Thus, the proper sequencing of some of the directives can be just as important as the directives themselves. For example, suppose the directive was given to write to the RD screen, followed by some text which appears on the RD screen. Suppose this was followed by another directive to write on the RD screen, followed by some more text. MIOP will do exactly as commanded, flashing the first RD display for the amount of time you can blink, then replacing that with the second batch of RD text. Obviously, the first batch of text will not be readable. Response sequencing is probably even more error-prone. There are solutions to these problems, and some specific solutions will be given. However, the

author will want to know the general solution to the entire sequencing requirements since lesson scenarios will inevitably be devised for which pat solutions are not appropriate. For these, the author will have to rely on understanding the circumstances of the data transfer, whether by directives, by sequencing, or a combination of both.

Fortunately, much of this discussion which may seem confusing in print, turns out to follow some pretty good rules of common sense in the execution pattern of the lesson, so that a little experience with the system makes it a lot more clear. Since the recent lesson conversion experience is very relevant to this report, it may be comforting to know that the tryout of the converted lesson material on TACFIRE equipment occurred in intervals a month apart, after batches of nearly 3000 lines of lesson were converted, and the tryout was done on the opposite side of the country by different individuals, yet there were remarkably few conversion errors. Reports from the tryouts were relayed over the telephone so that similar errors could be avoided from then on.

The discussion will now turn to more concrete authoring and lesson conversion problems, but references will be periodically made to the above considerations of the inner workings of the enhanced MIOP as a means of explaining the root of the problem and the nature of the solution.

SCREEN SIZE CONSIDERATIONS

TACFIRE terminals usually have 7-line screens. The ACC has two such screens, the RD and the CED. The MIOD terminal has seven lines, as also do the VFMED's (except for the newer 14-line units).

The first PLANIT/TACFIRE installation, as well as the NORMAL mode of the current installation, allocates six of the seven screen lines for output information, reserving one line for the response. If more than six lines are sent at a time, the system will display the first six and light the PRIORITY MESSAGE switch. When the user has read those six lines, the PRIORITY MESSAGE switch is pushed to display the next six lines (or the remaining lines if less than six). This scheme was implemented to provide as much correspondence as possible between the TACFIRE screens and commercial printing terminals (or scrolling CRT's).

The present PLANIT/TACFIRE installation, while maintaining that scheme in the NORMAL mode, makes no such provisions in the CONTROL mode and reserves no lines. Thus, if eight lines are sent to a 7-line screen, the first seven will be on the screen and the eighth will be lost. It becomes the responsibility of the author to make sure that the screen does not

overflow. No special significance is attached to the PRIORITY MESSAGE switch (or any other switch) other than that given it by the lesson.

Responses are taken from switch actions and from text on the CED screen. In the case of text response from the CED screen, the lesson can direct that any line or part of a line from that screen be considered. No distinction is made regarding how the characters got onto the screen. They may have been typed by the user or displayed by the lesson. It is common for the lesson to fill the CED screen with format messages containing blank spaces which are to be filled in by the user. In that case, the response sent back to the lesson will probably be a combination of characters displayed by the lesson and those typed by the user.

The rationale for the above methods of processing is not to facilitate lesson authoring, but to make the equipment behave in authentic TACFIRE fashion while still retaining authoring capabilities. This then provides opportunity to include realistic TACFIRE terminal practice within the context of the training course.

When authoring, one must be careful not to overflow display screens. This is not because of possible damage, rather it simply means that some of the instruction meant for the user would be unreadable.

In the case of the lesson conversion task, the text was organized into segments of six lines each. The organization was done with full realization at all times of which six lines were going to appear together. Many blank lines were added to the text in order to achieve their desired screen formatting. At the end of each sixth line, an abbreviation was included, "(TCA)" which meant, "Take Continue Action." TCA was explained at the beginning of the lesson to mean "press the PRIORITY MESSAGE switch." All such occurrences had to be reorganized, often making them into separate frames. With the extra (seventh) line available, some TCA's could simply be omitted. However, for those that would run over seven lines, a Q frame was inserted such that the allowable lines of text were in G2. A convention was established and explained at the beginning that designated the characters "(MORE)" on the last line to indicate that more text was waiting to be displayed. Then the command directive $$$ PM-ON was added which illuminated the PRIORITY MESSAGE switch, and the user had been advised that the PRIORITY MESSAGE switch should be pressed to see the next display. This, however, also required the insertion of a G3 to accept and process that response (being no longer handled automatically by MIOP). The resulting execution looked nearly the same at the terminal other than the fact that the display was switched from the CED to the RD for

reasons to be discussed later.

This organization appears over and over again in the converted lessons. There are numerous examples of it. However, the example would also show some logoff and reentry provisions which might not yet be understood, so the example will be deferred to another section.

In general, the text on any screen might be a composite of one or more PLANIT frames. For any given frame, the author must consider the possibility of wanting to branch into it, and, if that occurs, how much accumulated text will go onto the screen. A workable solution for the lesson conversion effort (and probably for new authoring as well), was to adopt a self-imposed limit of the number of lines of text in any G2 of a Q frame. In the case of the converted lessons, the chosen limit was six. Then, adopt another limit that no more than 7-n lines of text can be sent to a display prior to any entry to another frame (in this case, one line). This provides the necessary assurance that screens will not overflow.

Even with a rule such as this, there will need to be exceptions. There will be times when more than one line must be displayed prior to entry into another frame. In those cases, the next frame can be checked to be certain that the accumulated total does not exceed seven, or if it does, an additional Q frame can be inserted into the lesson flow which only has the (MORE) and $$$ PM-ON in G2, and the response line in G3. This will stop execution while the display is being read, and enable execution again when the PRIORITY MESSAGE switch is pushed.

Thus, in the lesson conversion effort, all of the TCA logic was removed, and hundreds of Q frames were augmented or added to include the PRIORITY MESSAGE switch logic.

Note that no such considerations need to be made for the ELP since it displays continuously, so long as paper is available.

LESSON EXECUTION ENVIRONMENT CONTROL

The CONTROL mode of PLANIT/TACFIRE requires what amounts to an environment for proper execution. This environment includes such considerations as lesson logoff, interruption, reentry, lesson checkout aids, and response processor default conditions.

In order to establish the desired lesson environment with the minimum of extra authoring effort, a set of control frames were devised which were added to each of the lessons. The control frames were intentionally designed to require a minimum of tailoring for each new lesson. A copy of these frames is shown in Figure 1.

```
 1-  1FRAME 1.00 (D.)
 2-  2C:CHANGE FUNCTION TO FN  C:CHANGE ABSOLUTE TO AB
 3-  2C:SET MATRIX(LK,20) C:LK=1000+LK C:SET KEYWORD ON
 4-  2C:LK(1)=ARRAY(2,5,9,14,16,22,27,30,34,50,52,58,60,64)
 5-  2C:FN GO(I)=GOTO 999 FOR(QQ=I)
 6-  2C:FN LOOK=LK(SUM SUM 0**DF(I) FOR(I=1,20 J=QQ+.001))
 7-  2C:FN DF(I)=SUM AB(K)+K FOR(K=100(LK(I)-J))
 8-  2C:FN XIT=GOTO 998 FOR(QQ=FRAME)
 9-  2IF LINK(10) LQ 2  C:QQ=2  B:999
10-  2ELSE C:QQ=LINK(10)  C:QQ=LOOK  B:999

11-  1FRAME 996.00 (D.)
12-  2C:LINK(10)=0  C:FM6=997  B:FM6

13-  1FRAME 997.00 (D)
14-  2F:THE NEXT LESSON SEGMENT, FM6, IS NOT IN THE SYSTEM.
15-  2F:YOU WILL NEED TO GET THE MONITOR TO LOAD IT FOR YOU.
16-  2F:THEN START AGAIN JUST AS YOU DID TODAY.  HAVE A GOOD DAY.
17-  2F:$$$ RELEASE  C:FINISHED

18-  1FRAME 998.00 (Q)
19-  20SESSION FINISHED.  PRESS 'PRIORITY MESSAGE'.          (MORE)
20-  2S$$$ SA-CLEAR CE-CLEAR PM-ON $
21-  3S SP
22-  4F:$$$ RELEASE  C:FINISHED

23-  1FRAME 999.00 (D) LABEL=START
24-  2C:SPECIAL(1,1,TERMINAL,1,5,0,0,0)
25-  2F:$$$ RD-WRITE SA-CLEAR CE-CLEAR COPY-OFF  B:QQ+0
26-  $$$$
```

Figure 1. Control frames for use with Enhanced PLANIT lessons.

The five control frames in Figure 1 were added to each of the converted lessons (with minor variations). The first frame must occur first in each lesson; the remaining four may occur immediately after the first or at the end of the lesson.

If other data are to be initialized for the lesson, the commands can be added to the first frame.

Some modifications are required to make the control frames fit the lesson. Referring to Figure 1, in line 4, the entries in the LK array must be changed to coincide with the desired reentry frame numbers (i.e. the numbers of the frames where reentry will place the learner at a meaningful point in the context of the lesson. In line 9, the frame number "2" must be changed to the number of the first subject matter frame of the lesson (i.e. the next one to be executed following the initial control frame). In lines 12 and 14 the name must be changed (from "TEST" in this example) to agree with that of the lesson to which the branch is being made. If no further lesson branches are to be made, then frames 996 and 997 can be omitted, and the lesson will end with a branch to frame 998.

Note that the control frames are set up to accommodate up to 20 reentry points. If more than that number are desired, the "20" figure must be changed in lines 3 and 6.

Now working through the example in Figure 1, line 2 is simply a couple of name changes for convenience. Lines 3 and 4 establish the LK array with reentry frame numbers listed in the beginning cells and some large number ("1000" in this example) in the remaining ones. The KEYWORD processor is also SET ON, a condition that will be discussed later.

Lines 5 through 8 define four functions. The "GO" function can be used by the author while checking the lesson to cause execution to begin at any arbitrary frame, but only after executing frame 999 which establishes the CONTROL mode. Note that the CONTROL mode only applies to lesson execution. Thus if the author intends to try out arbitrary pieces of the lesson, provision such as this must be made to cause execution to make the SPECIAL call which reinstates the CONTROL mode. This allows the author to enter CALC and type, "GO n" where n is some frame number, after which execution will begin at frame 999 and from there to frame number "n".

The LOOK function, defined on line 6, compares the current frame number value in item QQ to the array LK, selecting the largest value from array LK that is less than or equal to QQ for the returned value. Thus, for any frame QQ, the

value of function LOOK will be the appropriate reentry frame number such that after having had the lesson interrupted for some reason, the student will resume at a convenient point to pick up the context again. Function DF in line 7 is simply called by function LOOK, and has no other purpose. Function LOOK uses an "indicator" calculation, taking advantage of the special properties of zero raised to a power. The result can only be zero or one, providing the logic for an "indicator function," one which has many uses among mathematical algorithms. This function will show up later as a means of optimizing conditional branching within the lesson.

Function XIT in line 9 allows the user to exit from the lesson. Even though PLANIT provides the FINISHED command for this purpose, the CONTROL mode makes it virtually unusable by the student. Thus, a SA switch was chosen to be used as a means of escaping from the lesson while in the CONTROL mode. This is something the author provides in the lesson rather than something built into PLANIT/TACFIRE. In this instance, the SPARE switch was described as the one to push to leave the lesson. In order to implement that provision, a test for the SPARE key is included in the majority of Q frames in the lesson, and if a match occurs, a call is made to function XIT (i.e. C:XIT). The execution of this function assigns the value of the current frame number to item QQ, and then branches to frame 998 where a final message is printed, the lesson is "RELEASED" from the CONTROL mode (i.e. returned to the NORMAL mode), and then execution is FINISHED. Many of the following examples will show the test for the SPARE switch action.

Lines 9 and 10 show the use of a LINK array cell to provide appropriate reentry in the event of a system failure. Notice in the above discussion that PLANIT's built-in capability for providing reentry is not being used, at least not in the usual way. Typically, the author of a PLANIT lesson would provide reentry information by "dotting" the frame type. However, for lessons in the CONTROL mode, execution must encounter a SPECIAL call near the beginning of any session. One could scatter SPECIAL calls throughout the lesson at every reentry frame but it would be awkward since D frames would need to be added for that purpose. The SPECIAL call must precede the sending of text to the terminal, thus the "dotting" of Q frames would not be suitable. The frames in Figure 1 provide a satisfactory solution to the problem. Notice that two frames are dotted, frame 1 (line 1) and frame 996 (line 11). These will always be the only dotted frames in the lesson when this scheme is being used. One remaining piece of this logic requires that the statement, "C:LINK(10)=FRAME" appear frequently in the lesson, usually as the first line of G4.

With the above preparation in the lesson, LINK(10) will be continually updated as the lesson progresses. Then if some failure should occur, the logic of frame 1 (Figure 1) will use the current value of LINK(10) to update the QQ value which will then be used to retrieve the appropriate reentry frame number from function LOOK, and after entering the CONTROL mode in frame 999, execution will resume at that frame. Notice that the restart logic is no different than if the SPARE switch had been pressed. When entering the lesson for the first time, LINK(10) will have the value of zero, causing QQ to be assigned the value of the first content frame of the lesson.

The author can also use these conveniences to enter and exit the lesson while trying it out. In addition to the GO function, the author can elect to type EX START (executing at the frame labelled START). This will cause the lesson to resume at exactly the point where the SPARE switch had last been pressed (but only after executing frame 999 to be placed back into the CONTROL mode). Or, the author may elect just to type EX, which would make execution resume at a marked reentry frame, just as a student would do. These three check-out conveniences (EX, EX START and GO n) give the author the needed flexibility to move freely about the lesson, checking the desired parts.

Frames 996 and 997 are not necessitated by the CONTROL mode. Rather, they provide a way of informing the student if the next lesson in the series is not currently available in the system. Note on line 12, if the name FM6 is assigned to a lesson in the computer, the branch will be made to it. If not, FM6 is also defined to have the value, 997, a frame number to which the branch will be made. Also note that LINK(10) is given the value, zero, so that the first frame of the next lesson (to which that value will be passed) will function properly as prescribed on lines 9 and 10.

It is important that the PLANIT SET operator <u>not</u> be used with the lesson name, FM6 or the item name, QQ. The logic would not work properly if it was.

Finally, the command form "B:QQ+0" might raise some questions. The more obvious form, B:QQ would also work. This form was used as a matter of efficiency. In the case of B:QQ, PLANIT would first make an extensive search looking for a frame label or lesson name of QQ before using its CALC value. If the QQ+0 form is used, the intent is obvious, and PLANIT will not spend time on false searches.

Line 25 shows the starting conditions which were chosen for the ACC in this lesson series. Text is to go to the RD screen, no SA lights are on, the CED screen is blank and the ELP will not be making a copy of that which goes to the screen.

By adding the above five frames to each of the PLANIT/TACFIRE lessons, most of the authoring and checkout features are regained which the CONTROL mode impaired. The example was organized to show one frame at the beginning of each lesson and the remaining four at the end. This arrangement, although satisfactory for card input, would be a nuisance for terminal input since large numbered frames cannot arbitrarily be inserted from the terminal. In that case, the above frames could just as well be the first five (Frames 1, 2, 3, 4 and 5) using the same "dotting" arrangement and labels. With this approach, the author would maintain a "starter" lesson comprised of the five frames, which would be copied and used as a foundation for each new lesson segment.

The above five frames provide a number of conveniences for authoring and execution. However, some authors might want to tailor the five-frame sequence to their own needs or devise different ones. These five frames are not being suggested as the best, but they are suitable, relatively efficient and suit most of the needs.

Finally, the KEYWORD answer processor was placed in the "SET ON" status (in line 4). The SET ON condition indicates that the KEYWORD status will not change throughout the remainder of the lesson unless it is explicitly changed. There are several answer processors (KEYWORD, PHONETIC, TEXT, FORMULAS, etc.), all of which are normally in the OFF condition and changed in given frames where matching variations are desired. PLANIT/TACFIRE does not change that procedure except that the nature of the answers to be matched ordinarily requires a KEYWORD ON condition. Therefore, giving it the "SET ON" status at the beginning saves many KEYWORD ON commands later on. In this case, the status of KEYWORD is expected to be "ON" upon entry into any frame. Thus, if the status is explicitly changed in any frame, it would be wise to include the command, C:SET KEYWORD ON as the first line in the G4 of that frame, returning the status to its expected value. The wisdom of using the KEYWORD status as "SET ON" will become apparent when one examines the recurring need to match switch action mnemonics.

AUTHORING COMMAND REPERTOIRE

The PLANIT facilities for lesson building are not changed in any way whatever when being used to build PLANIT/TACFIRE lesson scenarios. However, several reasons suggest that the author may wish to avoid use of certain of the language features, including:

- Conditional Q frames
- Multiple-Choice (M) frames
- Question (Q) frames which contain only numeric answer tags in G3
- Commands which produce multiple printout lines (e.g. STATUS, PRINT CALC, etc.)
- Use of control characters to enter and exit CALC during lesson execution
- All uses of interactive CALC during lesson execution

The above restraints pertain only to the CONTROL mode. In the NORMAL mode, PLANIT functions as it normally does. However, the CONTROL mode introduces certain problems regarding the terminal reply which demand special consideration.

Looking at the above list in more detail, the reader should consider the expected operations in light of the two-step terminal read. Recall that the first response from the terminal is always the identification of the switch actions, the last of which served as a transmit key to deliver the contents of the screen (CED) to PLANIT (see section A-3.0 of the Design Description Document for PLANIT System Enhancements). Thus, the first match to be made will be on the switch mnemonics, not the Conditional Q choice or the Multiple Choice tag. So, the matching will not occur on the expected answer set, and, in fact, a match would seldom if ever occur. Figure 2 illustrates how a Multiple-Choice frame would have to be written to get around the two-step read problem, which only reinforces the recommendation that the use of the frame be avoided. Notice that some logical path must be provided to process the switch mnemonic information (line 11 in Figure 2), and all spontaneous output must be avoided (e.g. CHOOSE ONE OF THE ABOVE LETTERS, ENTER YOUR ANSWER, NUMERIC ANSWER PLEASE, etc.).

```
 1-  1FRAME 1.00 (M)
 2-  2WHAT IS PLANIT?
 3-  2$$$ SA-CLEAR CE-CLEAR $
 4-  3A.  A HEAVENLY BODY
 5-  3B.+ A COMPUTERIZED TRAINING SYSTEM
 6-  3C.  A DESK CALENDAR
 7-  4    C:RELATED ON
 8-  4AC  F:WRONG
 9-  4B   F:RIGHT
10-  4*   F:TYPE ON THE TOP LINE OF THE SCREEN. B:1
11-  4-   B:$$$ GET $
12-  4-   F:NO, YOU SHOULD HAVE TYPED A, B OR C.  THE ANSWER WAS B.
```

Figure 2. Sample format requirements for a PLANIT M frame in PLANIT/TACFIRE lessons.

The problem in the Multiple-Choice frame which is created because of the answer input sequence is also very similar for the Conditional Q frame. If anything, it is even more difficult to devise one that is satisfactory.

Interactive use of CALC is impossible for the same reason. The input that PLANIT expects is the CALC expression, but the one received is the switch action mnemonics. The use of control characters is a related problem because their function is to switch the user to interactive CALC.

On the output side of the interaction, two considerations are of utmost importance:

- Spontaneous output
- Lengthy output

Spontaneous output, as was mentioned earlier, refer to those messages which PLANIT gives to the user during execution which are not contained within the lesson. For example, if the user responds with a blank line, and no other provision has been made, PLANIT will output, ENTER YOUR ANSWER, and then expect another answer. This would be enough to get the CONTROL mode read sequence out of step, since the next answer to be received would be the switch action mnemonic, not the line of text.

One of the changes made to PLANIT during this contract which normalized this part of the operation (as well as provided more general capability) was to permit the author to detect conditions within the lesson which had previously produced spontaneous messages. Now, the "*" tag appearing in G4 will put the null answer condition under author control. This is also true of the "-" (unanticipated answer) tag in the Multiple-Choice frame. This means that the tag must be present in G4 to avoid the possibility of generating a spontaneous message. Assuming that MC frames will not be used, the two pertinent conditions in the Q frame are the null answer possibility and the NUMERIC ANSWER PLEASE message. Thus, if text from the CED screen is being evaluated, it is important that a "*" tag be included in G4 of that frame in the event that the portion under consideration is blank. Also, if the portion being evaluated is expected to be a number, it is important that at least one letter tag appear in G3 of that frame even if it must be a dummy answer.

Finally, regarding the lengthy output, use of commands such as STATUS and PRINT CALC will probably not fit on the screen and the end of the printout will be lost.

PLANIT/TACFIRE LESSON OUTPUT CONSIDERATIONS

In the CONTROL mode, lesson output can be directed to the RD screen, CED screen or the ELP. In addition, with COPY-ON, the ELP makes a copy of everything going to either screen. The Enhanced Authoring scheme makes it very easy to direct the output to either of the devices. Thus, the only suitable basis for choosing where to send the output hinges on the convenience and utility for the user. If the subject matter concerns TACFIRE operations, or if the "students" are people who have become used to TACFIRE operations, then they will expect certain kinds of information to be directed to each of the devices. For example, in TACFIRE, the RD screen is generally used to display information upon which the user is to act; the CED screen usually receives formatted data within which the user is to "fill in the blanks" and input the results, and the ELP usually keeps the permanent transcript. Most lesson material can also be expressed in a way that it would be analogous to the above, and this would provide one suitable set of guidelines to determine which output goes where.

With rare exceptions, the converted Fire Mission lessons direct lesson text to the RD screen, and empty fire mission formats to the CED screen. The ELP is only used to demonstrate fire mission functions which send output to the ELP. Therefore, upon entering a new PLANIT frame in the CONTROL mode, the normal (default) course of action is assumed to be that any text to be displayed will go to the RD screen.

Although it is not absolutely necessary, it is very convenient from an authoring standpoint to adopt certain default guidelines so that, upon entering any new frame, certain assumptions will govern execution unless explicit Enhancement commands direct otherwise. In this way, the freedom of branching around within the lesson is preserved. The alternative situation would force the author to anticipate every possible avenue of entering each frame at the time it is written.

It will be seen that the practice of specifying Enhancement commands in the first line of the Q frame is not acceptable. If it were, then conditions of execution prior to entry into the current frame would not matter. This report will adopt the position that Enhancement commands are appropriate in the very first line of text sent out after an answer is processed or as the very last line before another answer is expected, and rarely anywhere else. This normally eliminates the possibility of including commands on the first line of a Q frame since some text will very likely have already been sent out in the form of feedback from the previous frame,

thereby making the commands fall in the middle of the text stream. An example might help. Suppose that some feedback message from Frame 16 is sent to the RD screen, and then execution passes on to Frame 17, a Q frame. Suppose the first line in G2 contains the command, $$$ RD-WRITE, and several lines of text follow it. The effect of this sequence would be that the user would only see the text from Frame 17. The feedback message would have displayed on the RD screen, but because of the new RD-WRITE command, the new display would be written over it. The feedback message would be little more than a flash, giving no opportunity for reading it.

One obvious solution is to be sure that nothing is on the screen when moving from one frame to the next. However, the implications of such a decision are not good. Either all feedback messages must be combined with the text in the next G2 or a separate Q frame must be inserted between each two instructional frames to provide the pause while the message is being read. The first case eliminates the possibility of branching around within the lesson and the second is awkward, inefficient and space-consuming.

A more satisfying solution is to adopt a set of default conditions that govern the execution of each of the frames. Exceptions to the conditions can be tolerated, but only by full cognizance of the possible routes into the frame. The default conditions chosen for this lesson conversion effort were as follows:

- Output is going to the RD screen.
- The ELP is in the COPY-OFF condition.
- Not more than one line of output has been sent.
- KEYWORD is SET ON, all other answer processors are OFF.
- The CED screen may or may not have information displayed on it. If not wanted, it should be cleared.
- The Priority Message switch may or may not be lit.
- The Message Address switches may or may not be lit.

Given the above assumptions, the frame currently under consideration can be written in such a way that it could be entered from anywhere within the lesson.

In order to implement the above scheme, the Enhancement Commands are written into the lesson only at the beginning of feedback messages or in the last line of text, or both. The only exception to that rule is where text is to go to

two or more displays (RD, CED and/or ELP) in the same text stream. In that case, the appropriate WRITE command is inserted at the beginning of the text to that device, and no more text can then be sent to the previous device within the same text stream. The text stream starts with the first output character after a response is received and continues until execution is halted for the next terminal input. When the switch in display occurs, the final display command will switch the display mode back to "RD-WRITE."

Figure 3 shows an example of text going to the RD and CED displays in the same text stream. Notice that the text stream begins with one of two feedback messages in Frame 164, and ends with the last G2 line of Frame 165. That last line contains a "RD-WRITE" command even though no text follows it within that text stream, setting up the desired condition for the next stream of text. This example happens to show a case where a two-line feedback message is anticipated prior to Frame 165. Those two lines go to the RD screen along with the five in Frame 165, after which come the CED lines of text.

Finally, to reiterate, the lesson structure has no bearing on the manner in which the text stream is defined. The text stream can be composed of text only from one frame, from several sequential frames, from randomly distributed frames that are sequential only because branching instructions make them execute sequentially, or from non-lesson text that some lesson command might cause to be output. Interspersed lesson and/or CALC commands have no bearing on the text stream if they do not produce text themselves. Also, note that the "*" prompt constitutes an output line in the text stream. For this reason, and others to be discussed later, the "*" prompt was universally suppressed from the converted lessons. This was done by terminating the final line of the G2 text with the "$" suppress character, nearly always occurring on the line of Enhancement commands.

Thus, the output text stream consisted of display selecting enhancement commands (if different from the RD), followed by text to that display. That was followed by the selection of a different display and text, if output was being sent to more than one. Finally, the output stream was normally terminated by a line of Enhancement commands which reset all of the expected terminal conditions, and that line was terminated by a $ character. This pattern occurs repeatedly throughout the converted lessons, being maintained despite many different variations of lesson structure and uses of a variety of authoring aids for displaying the text.

```
1FRAME 164.00 (Q)
2UNDER NORMAL OPERATING CONDITIONS AN FM;SUBS MESSAGE
2INDICATING ADJUST FIRE COMMANDS IS TRANSMITTED TO THE
2TACFIRE COMPUTER BY:
2     A. DIVARTY          C. AN FO
2     B. THE FSO          D. A REINFORCING BATTALION   (CHOOSE A LETTER)
2$$$ SA-CLEAR $
3A MA
3B MB
3D MD
3C+MC
4-F:CHOOSE THE LETTER 'A', 'B', 'C' OR 'D'. B:164
4ABD F:NO, AN FO WOULD TRANSMIT AN FM;SUBS MESSAGE
4     F:UNDER NORMAL OPERATING CONDITIONS.
4C F:RIGHT ON.

1FRAME 165.00 (C)
2THE CED SCREEN BELOW LISTS THE MNEMONICS FROM THE FM;SUBS MESSAGE
2(FIGURE 14, PART A). LIST THOSE MNEMONICS ON THE TOP LINE OF THE
2CED WHICH YOU WANT EXPLAINED. PRESS 'PRIORITY MESSAGE' TO STEP
2THROUGH THE EXPLANATIONS OR 'PAGE' TO PROCEED WITH THE LESSON.
2IF YOU WANT THEM ALL EXPLAINED, PRESS 'CYCLE MESSAGES.'
2$$$ CE-WRITE
2@@ALTER,   EOM,   RAT,   DIR,   SIT, SHIFT,  SPOT,   AUF,   ASF,
2  LOT,   CHG,    ME,  CONT,  TYPE, DISPO,   CAS, UFFES,    SH,    FZ
2$$$ SA-CLEAR RD-WRITE PM-ON $
3A PG
3B CM
4A F:$$$ SA-CLEAR CE-CLEAR B:186
4B C:SET MATRIX(X,20) C:X(I)=I FOR(I=1,20) C:SET NM=1 B:167
4- C:SET MATRIX(X,20)  C:X=20+X
```

Figure 3. Text being sent to two TACFIRE displays in a single text stream sequence.

PLANIT/TACFIRE LESSON INPUT CONSIDERATIONS

In the CONTROL mode, lesson input (terminal response actions) is processed in one or more steps. Each required step must correspond to a point in the lesson where a response is expected. Thus, if the intention is to discover what has been typed on each of the seven CED lines and which switch was pressed to transmit the contents of the screen, a minimum of eight lesson response points must be encountered. This can be done with eight frames or by recycling through a lesser number. Also, no output can be sent to any display device until the response has been fully analyzed.

One important implication of the above is that any textual reply which is typed on the keyboard normally requires at least two PLANIT frames to evaluate what was typed. The first of the pair of frames processes the SP switch action mnemonics. This frame must appear whether or not the author cares which switch was pressed to transmit the text on the CED because the response line contains switch action mnemonics, not text. In the case of the lesson conversion, this frame was used to check for the SPARE switch in the event the user wanted to logoff, otherwise execution progressed to the next frame where a GET Enhancement command transmitted the line of text for analysis.

The analysis of the text line follows normal PLANIT Q frame rules with some important exceptions.

First, since the variations of the GET Enhancement command makes it somewhat likely that null answers will be transmitted, the "*" must be included in G4 to provide a branch back to the prior frame with appropriate corrective messages. Without it, PLANIT would prompt, "ENTER YOUR ANSWER," but the answer that would be transmitted would be the switch mnemonics, not the expected line of text. The match of that answer would fail and then the lesson would continue. One suggested alternative is to enforce a one-second waiting period (0 WAIT 1) in that frame, where the time-out tag (') causes a branch back to the same frame. This would put the answer processor back in step but at the cost of recording a wrong (timed-out) answer along the way.

Next, the R: PLANIT command may not be used since it also would imply the processing of switch action mnemonics in the wrong frame. In place of the R: command, it will be usual to include a F: corrective message, followed by a B: branch back to the prior frame.

Finally, no feedback text can be given until the entire terminal response has been analyzed, regardless of the number of frames it takes to do it. In general, the pattern will be to say nothing for each response analysis line which meets the criteria until the final line has been processed. However, if an error is found, and no further analysis is desirable until a new terminal response is received, then feedback can be given at that point and a branch made to an appropriate frame to process the next set of switch action mnemonics.

This part is possibly the most confusing aspect of the Enhancement command scheme. It breaks down a complete terminal response into several components, the first of which is an input line of mnemonics that describe the switch action(s) used in the process of making the response. That analysis is performed by a PLANIT Q frame. If no text is anticipated on the CED as part of the response (i.e. if switch actions comprised the total response), then no further analysis of that response is necessary. However, if text was composed at the keyboard (on the CED) as part of the response, then any one line or any segment of one line can be analyzed in a PLANIT Q frame. The GET Enhancement command in G2 specifies which line (and the segment bounds if less than a full line), and the standard G3 matching information analyzes the line. If more than one line and/or segment is involved, the process is repeated in yet another PLANIT frame until the analysis of the full screen is complete. It is this sequence that cannot be interrupted by any kind of output text to any display.

When composing the G3 lines for analyzing the response, keep in mind that the text consists of any characters which might have been displayed on the CED as well as those which the user has typed in. If the user begins with a blank screen, then only that which is typed will be transmitted. However, it is common to display an empty format for the user to fill in. In that case, the G3 matching information must also take into consideration any of the format characters which fall in the match field. Figure 4 shows an example of a two-line format which the user fills in. Notice that all of the frames in the example are necessary to analyze that single terminal response. If the reader is familiar with using PLANIT's REPLY buffers to perform subsequent analyses of a given response, it is somewhat analogous to the above, except that, unlike the REPLY buffer scheme, care must be taken not to interrupt the analysis sequence with output.

Reflection on the above will show why answer analysis is most often desirable with KEYWORD ON. In the case of switch action mnemonics, several are transmitted, but KEYWORD permits the identification of only the one(s) of current interest. To perform an exact match, the author would need to know the

```
1FRAME 911.20 (Q)
2O.K.  SELECT AN EMPTY MESSAGE FORMAT AND ENTER THE CHANGED
2OBSERVER LOCATION AS FOLLOWS:Q
2  OBSERVER 3
2  GRID 3640028320, ALT 360
2$$$ SA-CLEAR CE-CLEAR $
3A+FC F-8
3B FC A-8
3C FC B-8
3D+FC E-8
3E+CA
4C:RELATED ON
4A B:FMOBCO R:$$$ SA-CLEAR $
4B B:FMREAF R:$$$ SA-CLEAR $
4C B:FMSUBS R:$$$ SA-CLEAR $
4D B:FMDIR B:912
4-F:NO, WRONG SWITCH.  TRY ONCE MORE. B:911.2
4-F:NO, REVIEW THE PROCESS AND TRY AGAIN. B:876

1FRAME 911.30 (Q)
2$$$ GET(2) $
3A+OB:3;CORD:36400/28320/360;
3A+OB:03;CORD:36400/28320/360;
3B REAF
3B SUBS
4A F:VERY GOOD.  CHECK THE ELP FOR THE REPORT. B:ELP B:912.4
4- F:NO, WRONG ENTRIES. FILL IN ONLY 'OB' AND 'CORD'.  TRY AGAIN.
4 B:911.2
4- F:NO, OB: 3;CORD:36400   /28320   /360  ;       TRY AGAIN. B:911.2
4* F:NO MESSAGE FORMAT.  DISPLAY THE FORMAT FIRST. B:911.2
4B F:WRONG FORMAT, USE FM;OBCO OR FM;DIR.  TRY AGAIN.

1FRAME 984.00 (P) LABEL=FMOBCO
2F:$$$ CE-WRITE
2PRINT'      ;P: ;SB: / / /  /   ;C:   ;SG:  ,  ;DT:  ,  /  /  ;'$
2PRINT'ID:    ;A: ;'
2PRINT'FM;OBCO;OB:  ;CORD:      /        /     ;GZ:   ;SPHERE'
2F:$$$ RD-WRITE
2RETURN
```

Figure 4. Sequence of PLANIT frames used to display and evaluate a fire mission format.

identity of each switch mnemonic and the order of its transmission in the response. Even then, the allowable combinations of Message Select switches would make such a match nearly impossible. The KEYWORD match is very practical, using it to identify any (one or more) particular switch action(s) of interest. KEYWORD match also assists in locating correctly filled in blanks within a format without matching the entire format. It turns out that the instances where KEYWORD ought to be OFF are relatively few, making it more efficient to leave KEYWORD in the SET ON condition, changing it only for those frames where a different type of match is needed.

There are a couple of other input considerations which are relevant to TACFIRE lessons. One has to do with answer analysis algorithms and the other with the amount of work space needed to complete the analysis. Both considerations pertain to PLANIT authoring in general but characteristics of TACFIRE lesson scenarios make them especially pertinent here.

The PLANIT answer analysis algorithm has to do with parsing rules which PLANIT applies to the lesson answer and the student answer when a match is attempted. The parsing rules follow common sense so that the author seldom worries about them. Words constitute separate match units as also do numbers, computation symbols and punctuation. Multiple blanks are treated as a single blank except for leading and/or trailing blanks which are disregarded. When the presence of blanks is not necessary to define separations between match units (as is often the case), then blanks are completely optional. Thus:

```
3 + 4 - 5
```

will match:

```
3+4-5
```

TACFIRE formats have the property that the match units do not need the blanks or, in other words, the presence or absence of blanks would make no difference in the matching algorithms. This assumes that the TEXT processor is in the OFF condition. If TEXT is ON, only the blanks define separations between match units. Thus, with TEXT OFF, the G3 match lines can just as well be written without the blanks which normally occur in the format. It will not affect the match.

In regard to the work space, PLANIT has a limited amount of space available in which to attempt the match, and if that space is exceeded, the match will fail. The space is determined by the WORKSPACE Generation parameter. The needed space can be calculated as four plus the number of match units in the lesson answer plus the number of match units in the student's answer.

In conventional PLANIT installations, the answer analyzer can match 50 to 60 units, which can be distributed between lesson and student in any manner (i.e. 25-30 in each or any other combination whose total does not exceed WORKSPACE-4). is is usually more than enough. Answers are seldom more than a few words in length.

However, in TACFIRE scenarios, the format lines often contain 25 to 30 units before data are filled in. The amount of available matching workspace can become critical, especially when using a commercial installation to check out TACFIRE lesson scenarios. Answers will fail to match for no other reason than the work space was exceeded. This is not a problem on the TACFIRE hardware because the WORKSPACE parameter was given a sufficiently large value to prevent it from happening. If a certain commercial installation is known to be intended for TACFIRE scenario preparation, then it, too, should have the value of WORKSPACE raised. A range of 100 to 110 should be satisfactory for any format. An alternative is to keep the author answers short by using KEYWORD to find the answer within the longer line. This, again, shows the advantage of leaving KEYWORD in the SET ON condition. Another possible way to shorten the match is to use the Enhancement GET command with values which select only a part of the line at a time to be matched. However, this increases the number of frames which would then be required to evaluate the full format. Recall that the response line contains the format characters themselves in addition to anything the user types.

Thus, there are several ways to circumvent the problem. If the PLANIT installation anticipated TACFIRE scenarios and allowed a WORKSPACE parameter value in excess of 100, then there should be no concern. However, if the value is around 60 (as it often will be), then the problem should be considered, especially if answers which seem like they should be right are counted wrong.

One final input consideration has to do with the PLANIT use of the "?" (i.e. the only typed character of the reply). This produces a non-lesson generated message, ENTER YOUR ANSWER. Any answer then entered for that frame will not match because it will be the switch mnemonics, not the typed answer, that will be processed. There is seldom, if ever, any reason for authors of TACFIRE lessons to request that input, but it is pointed out here with the suggestion that it should be avoided.

LESSON CONVERSION METHODS THAT HAVE AUTHORING IMPLICATIONS

In the process of converting 11,000 lines of lesson material to PLANIT/TACFIRE format, many experiences were encountered which have direct bearing on authoring. There were also a number of techniques developed which have no particular bearing on future authoring, such as skills in using our particular context editor (WYLBUR) to find and change lesson code as efficiently as possible. It is not likely that these editing techniques would be of particular interest since they were closely tied to the manner in which these lessons happened to have been written and to the pecularities of our editor utility. Therefore, the following discussion will be limited to those aspects which may provide useful information for future authoring.

Lesson expansion. PLANIT lessons on TACFIRE tend to be larger than on commercial equipment, even when the content is essentially the same. And TACFIRE lessons in the CONTROL mode will be larger yet than TACFIRE lessons in the NORMAL mode. In the context of the present conversion experience, the lessons being converted were already written for TACFIRE execution in the NORMAL mode, or else even greater expansion would have occurred. As it was, the 11,000 lines grew to more than 14,000.

The cause for the expansion is twofold. First, the seven-line organization of the screens cause a higher degree of segmentation of the material than on normal commercial equipment. In order to intelligently answer the posed question, sufficient prompting information must fit on seven 72-character lines (six in the NORMAL mode). This causes a lot of repeating of textual information in order to keep relevant items together on the screen at any given time. Also, messages designed to provide hints or corrective information after wrong answers also must make provision for re-asking the question if another chance at the answer is being given, since the original question will be gone from the screen. It will be necessary to insert extra Q frames for no other reason than to insure that text associated with two different contexts gets divided properly on the screen.

In the CONTROL mode, it requires the insertion of a frame to break text into screen-sized displays. The automatic stepping through output text with the "PRIORITY MESSAGE" switch which is furnished in the NORMAL mode does not apply to the CONTROL mode, so each of those instances must instead have an explicit Q frame for that function. In addition to this, any response which is typed on the keyboard must be analyzed in two passes, the first one for the switch action mnemonics and the second for the typed text. This means that, for simple typed answers, the author will normally use two PLANIT frames where only one

would otherwise have been needed. Then, for each additional line (or piece of a line) which is to receive separate answer evaluation, yet another Q frame will be needed. Thus, authors of PLANIT/TACFIRE lessons will be looking for techniques to reduce the rate at which frames accumulate.

The recommendations will be speaking directly to this problem, but the following discussion will relate to the system as it presently exists.

Conserving lesson frames. There will be a need to conserve frames as much as possible. The problem does not lie in the amount of space consumed by the lesson because the extra frames are generally small, and add little to the amount of space being consumed. Rather, the problem stems from the rate at which new lessons become necessary. A PLANIT lesson is usually only a manageably small segment of a course, the maximum length being defined by the total number of frames being allowed. This number is usually about 100 or so, depending on parameters used to generate that particular installation. When these frames are consumed rapidly, it requires more PLANIT lessons to make up the course. Additional PLANIT lessons make the monitoring task that much more difficult, since there are more lessons to keep track of an more student records to collect. Thus, it pays in the long run to be as frugal as is reasonable with the frames as the lesson is being written.

One obvious way to conserve frames is to divide the information in such a way that the screen is kept full when it can make sense to do so. There will certainly be instances when one- or two-line messages are all that the lesson will accomodate. But, in general, an attempt should be made to divide the content, and intersperse the questions, between full screens.

Another frame-saving technique is to re-use frames where possible. This can be done by providing review paths through previously seen sequences, using RELATED to vary the feedback on the next time through. It can be done in the case of erroneous answers by presenting a one-line clue and then branching back to the same frame again (or the beginning of the sequence in the case of a keyboard response). It can also be done by placing frequently used displays in P frames and then calling them in subroutine fashion from other frames. Empty TACFIRE formats are particularly appropriate for this technique. Figure 4 shows such a format. Notice that PLANIT PRINT statements were used instead of F: commands, primarily because the formats use a variety of mnemonics immediately preceding colons, and the PRINT statement avoids any possibility of confusing the formats. Also, in regard to the formats, some of the characters go on the far right of the line. It is better to avoid writing lesson lines beyond column 71 in the event that sequence numbers might be punched

into the last eight columns of lesson cards. (Note that the group number in the first column must also be counted). Thus, the P frames, using the PRINT statements with "$" terminators to break long lines can be re-used to conserve frames. The Figure 4 example of the P frame also shows the switch to the CED display (where empty formats normally go), and the switch back to the RD display again at the end, the last line being a RETURN statement to make the frame function as a subroutine. With this organization, it would be important that the frame be called first, before any RD text is displayed, or last, after it is displayed. The P frame must not be called in any sequence that would break up text going to the RD, or else the first part of the text would simply flash past and not be seen.

There are also a number of PLANIT authoring "tricks" that can be used to reduce the total number of frames. These apply to any PLANIT authoring but could be especially important to PLANIT/TACFIRE lessons where lesson expansion is more prevalent.

Single frames can be constructed in such a manner that they perform multiple tasks. Figure 5 is an example of one such frame. Frame 123 in Figure 5 is designed to analyze the SA mnemonics plus seven lines of input from the CED, or a total of eight responses to PLANIT. Each response will be recorded and provision is made for unanticipated text as well as null lines. This method would be inappropriate for input data which contained similar lines, but that would be very rare in TACFIRE. The frame could be easily modified for input of formats which contained less than seven lines.

Note that Figure 5 uses the R: lesson command in its algorithm, also using the SPECIAL call version of the GET Enhancement command. Although this document earlier recommended against the use of the R: lesson format, its use in Figure 5 is clearly not according to normal PLANIT usage. Thus, both Figures 4 and 5 show the R: command being used, but its use is for a carefully considered branch back to G3 of the same frame, not for the normal answer retry function. In this context and with careful planning, the R: command can be very useful.

Another lesson structure which can reduce the number of required frames capitalizes on the properties of PLANIT's RECYCLE command. Essentially, RECYCLE permits the re-analysis of an answer line within the same frame to consider the previously un-matched portion of the line. This is a compact method of doing what might otherwise require several frames. Figure 6 shows an example of a sequence in which the use of RECYCLE significantly reduced the total number of required frames. In this case, the lesson happened to have been constructed

```
1FRAME 123.00 (Q)
2INFORMATION AND/OR QUESTION.
2$$$ SA-CLEAR CE-CLEAR $
30 SET N=0
3S+ KEYWORD MATCH FOR SWITCH MNEMONIC
3A+ MATCH FOR FORMAT, LINE 1
3B+ MATCH FOR FORMAT, LINE 2
3C+ MATCH FOR FORMAT, LINE 3
3D+ MATCH FOR FORMAT, LINE 4
3E+ MATCH FOR FORMAT, LINE 5
3F+ MATCH FOR FORMAT, LINE 6
3G+ MATCH FOR FORMAT, LINE 7
4ABCDEFS C:N=N+1 C:SPECIAL(1,1,TERMINAL,1,7,N,1,72) R:$$$ SA-CLEAR $
4C F:YOU ARE RIGHT.
4- C:PRINT'YOU MISSED SOMETHING ON LINE ';N
4* C:PRINT'LINE ';N;' WAS BLANK.'
```

Figure 5. One-frame structure for an eight-frame authoring sequence.

```
1FRAME 165.00 (C)
2THE CRT SCREEN BELOW LISTS THE MNEMONICS FROM THE FM;SUBS MESSAGE
2(FIGURE 14, PART A). LIST THOSE MNEMONICS ON THE TOP LINE OF THE
2CRT WHICH YOU WANT EXPLAINED. PRESS 'PRIORITY MESSAGE' TO STEP
2THROUGH THE EXPLANATIONS OR 'PAGE' TO PROCEED WITH THE LESSON.
2IF YOU WANT THEM ALL EXPLAINED, PRESS 'CYCLE MESSAGES.'
2$$$ CE-WRITE
200ALTER,   EOM,   RAT,   DIR,   SIT, SHIFT,  SPOT,   AUF,   ASF,
2  LOT,   CHG,    ME,  CONT,  TYPE, DISPO,   CAS, UFFES,    SH,    FZ
2$$$ SA-CLEAR RD-WRITE PM-ON $
3A PG
3B CM
3S SP
4C:LINK(10)=FRAME
4A F:$$$ SA-CLEAR CE-CLEAR B:186
4B C:SET MATRIX(X,20) C:X(I)=I FOR(I=1,20) C:SET NM=1 B:167
4S C:XIT
4- C:SET MATRIX(X,20)  C:X=20+X

1FRAME 166.00 (C)
2$$$ GET $
30 RECYCLE ON
3Z ,
3A ALTER
3B EOM
3C RAT
3D DIR
3E SIT
3F SHIFT
3G SPOT
3H AUF
3I ASF
3J LOT
3K CHG
3L ME
3M CONT
3N TYPE
3O DISPO
3P CAS
3Q UFFES
3R SH
3S FZ
```

Figure 6. Example of using RECYCLE and a computed branch to conserve frames (first of three pages)

```
4Z
4A C:X(1)=1
4B C:X(2)=2
4C C:X(3)=3
4D C:X(4)=4
4E C:X(5)=5
4F C:X(6)=6
4G C:X(7)=7
4H C:X(8)=8
4I C:X(9)=9
4J C:X(10)=10
4K C:X(11)=11
4L C:X(12)=12
4M C:X(13)=13
4N C:X(14)=14
4O C:X(15)=15
4P C:X(16)=16
4Q C:X(17)=17
4R C:X(18)=18
4S C:X(19)=19
4* C:SORT(X) C:SET NM=1
4- F:TYPE ONLY THE MNEMONICS SEPARATED BY COMMAS, AND ONLY ON LINE 1.
4  B:165

1FRAME 167.00 (P)
2IF X(NM) GR 6 B:LST
2ELSE B:L1,L2,L3,L4,L5,L6;X(NM)
2L1:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'ALTER'. B:LST
2L2:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'EOM'. B:LST
2L3:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'RAT'. B:LST
2L4:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'DIR'. B:LST
2L5:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'SIT'. B:LST
2L6:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'SHIFT'.
2LST:
```

Figure 6. Example of using RECYCLE and a computed branch to conserve frames (second of three pages)

```
1FRAME 168.00 (P)
2IF X(NM) LS 7 OR X(NM) GR 13 B:LST
2ELSE B:L7,L8,L9,L10,L11,L12,L13;X(NM)-6
2L7:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'SPOT'. B:LST
2L8:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'AUF'. B:LST
2L9:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'ASF'. B:LST
2L10:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'LOG'. B:LST
2L11:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'CHG'. B:LST
2L12:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'ME'. B:LST
2L13:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'CONT'.
2LST:

1FRAME 169.00 (P)
2IF X(NM) LS 14 B:LST
2ELSE B:L14,L15,L16,L17,L18,L19,LST;X(NM)-13
2L14:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'TYPE'. B:LST
2L15:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'DISPO'. B:LST
2L16:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'CAS'. B:LST
2L17:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'OFFES'. B:LST
2L18:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'SH'. B:LST
2L19:
2F:EXPLANATORY TEXT FOR MNEMONIC, 'FZ'.
2LST:

1FRAME 170.00 (C)
2                                                          (MORE)
2$$$ PM-ON $
3^ PG
4- C:NM=NM+1
4  B:167,165,186;0**(20-X(NM))+0**(20-NM)+1
```

Figure 6. Example of using RECYCLE and a computed branch to conserve frames (last of three pages)

in another format which used a number of frames, and there were several such sequences. Converting to this method of processing the responses reduced the total number of frames and helped to offset the increase in frames in other areas. In the Figure 6 example, the text in the P frames has been replaced to shorten the sequence for this document. However, in the lesson, each unit of text in the P frames could be up to six lines long. P frames were used in this case because of line branching properties which combined convenience with execution efficiency to permit several explanations to occur within a single frame. Theoretically, only one P frame would ever be needed for this type of sequence, but in fact, a limit exists on the amount of lesson text that a given frame can hold. Therefore, the scheme also allows the splitting of the text among enough P frames to accomodate size requirements.

Note that the function of the sequence in Figure 6 is to allow the user to type any number of mnemonics (up to one full line) at a time on the keyboard, and then step through the explanations of these mnemonics, one at a time, by pressing the PRIORITY MESSAGE switch. Or the CYCLE MESSAGES could instead be pushed which would be the same as having typed all the mnemonics. Also, the user can terminate the sequence at any time by pressing the PAGE switch, or sign off by pressing the SPARE switch. Thus, one sequence which contains a modest number of frames handles a variety of options, and RECYCLE is a key component to the algorithm.

Notice that the example in Figure 6 could just as well have required a frame for each of the 19 mnemonics being explained, and in fact, did in the original lessons. Here, it has been reduced to six.

Another technique for conserving lesson frames is also illustrated in Figure 6. Notice the last line of the example (in Frame 170) where the computed index for the B: command designates either of three frames (167, 165 or 186) to be executed next in the sequence. The index is computed from the explanation numbers (stored in the "X" array in Frame 166) plus the total number of explanations given (the value of the variable, NM). To compute the index, the indicator function, mentioned earlier (i.e. zero raised to a power), is used. Since there are two terms where zero is raised to a power plus the "1" term, the sum of the three terms can only evaluate to the values, 1, 2 or 3. The index will be "1" if there are more mnemonics in the requested list which have not yet been seen. A "2" result indicates that all in the current list have now been seen, so a branch should be made back to the starting point to give opportunity for another list to be typed. The "3" index result indicates that the entire list was seen, so the lesson should proceed. Notice that this one computed branch statement which only occupies a line in a Q frame would more often be regarded as a multiple-line Decision frame operation. Thus, the extra D frame was avoided.

Arithmetic algorithms such as that shown in Figure 6 can be very useful in structuring an otherwise difficult sequence within the lesson. However, such algorithms must receive careful attention so that they don't fail unexpectedly. Had this algorithm been organized differently, an incorrectly typed input line could have produced array subscripting errors. This algorithm will not fail regardless what the user does. Once an algorithm is worked out and tests show it to be foolproof and efficient, it can often be applied to different contexts within the series of lessons. Algorithms such as this comprise many of the "tricks" that lesson authors have been said to use.

Authoring which considers user convenience. PLANIT has been designed with convenience for the user in mind. However, the conveniences one normally expects in operating typical computer terminals are not necessarily present in TACFIRE terminals. In addition, the number of optional ways to enter a response on a TACFIRE terminal exceed that for most commercial terminals, especially in the variety of switches on the SA that might be pressed. Thus, the number of keystroke actions required to submit a multiple-choice type of response (single letter response) can vary from two to five, depending on how the lesson was written. Also, the illumination possibility for certain of the switch caps can be used to advantage to provide extra prompting so that the user is more apt to remember what to do.

Even choosing the most appropriate display screen for output can have a bearing on user convenience. The user may be in the habit of reading certain kinds of text from one display more than the other. If text is displayed on the CED screen beginning with line 1, then the user will probably need to blank the entire screen in order to type the requested response, thereby losing the remainder of the instructions about what is to be typed.

This document earlier recommended avoiding multiple-choice frames, referring to the PLANIT MC frame. However, this is not to imply that the multiple-choice format is to be avoided. In fact, the Message Address switches on the SP provide an excellent means of responding to multiple-choice questions, providing that the multiple-choice format has been presented in a PLANIT Q frame. The choice of the Q frame is necessary because the response will not be the "A", "B", "C", "D" or "E", but rather the mnemonics which result from the pressing of those switches.

Frame 164 in Figure 3 shows an example of a multiple-choice format presented in a Q frame which accepts as a response the pressing of Message Address switches. There are many

frames in the converted lessons which follow this general format. In choosing this method for presenting Multiple-Choice questions, several things must be kept in mind.

First, the Message Address switches are not the "interrupting" type, which means that the total response must include the pressing of one other switch after the letter switch choices are pressed. The author will need to instruct the users early in the lesson about the conventions for answering multiple-choice questions, and that instruction must include the pressing of another switch. The PAGE switch was chosen for the MC frames in the converted lessons because it is conveniently located just below the Message Address switches.

Second, the Message Address switches light when they are pressed, and stay lit until the lesson or the user turns them off. It is good practice to include an Enhancement command, SA-CLEAR in that final G2 command line of a MC-format Q frame. This will pose the question with all Message Address lights off.

Third, the hardware permits more than one Message Address switch to be in the "on" state at the time the next switch is pressed, so the response could very easily consist of more than one letter choice. As was stated before, the author will certainly want to have KEYWORD in the ON state to evaluate the switch mnemonics. Thus, without planned protection, the user could simply press all five Message Address switches every time and always get a right answer. This can be prevented by checking the wrong answers first. Figure 3 shows that a check is made for answers "A", "B" and "D" before the check for "C". The "E" choice is not checked in this example because it is not listed, and therefore is neither right or wrong. If that choice is made, it would be treated as any other unanticipated answer and another response would be requested.

Fourth, this MC format does not preclude a question which has more than one correct letter choices. For example, the instructions might be to choose all of the correct answers. In that case, the correct answer line in G3 of the Q frame will simply list all of the correct mnemonics, for example:

```
A+MA MC ME
```

The KEYWORD processor will then determine this to be the correct three. Note that a "wrong answer" match should have already been attempted for the MB and MD mnemonics, for example:

```
B MB
B MD
A+MA MC ME
```

Finally, an unanticipated answer alternative similar to that shown in Figure 3 (Frame 164, G4) needs to be included in the event that the user forgets the proper answer process and, perhaps, types the letter choice on the keyboard. Of course, a different frame organization could allow the typing of the response on the keyboard if that is desired.

Another user convenience relates in general to writing the lessons in such a way that they expect the user to respond in the manner that the TACFIRE equipment is meant to be used. Those who attempt to use TACFIRE terminals for PLANIT work are inclined to think of the terminals as being awkward and poorly designed, when in fact they are designed very well for their intended tasks. The taking of PLANIT lessons on TACFIRE equipment will be much more appealing if the author structures the lessons in such a way that the terminal equipment is used in the way its design intended. For example, the lesson may wish the student to identify that switch which, when pressed, will display the COMM LINE. To do that, it might list some of the switch names on a display and set up a multiple-choice format for the response, it could ask the user to type the name of the switch on the keyboard, or it could simply ask that the appropriate switch be pressed. The information among the three methods is essentially the same, but the amount of effort required of the user is vastly different. The equipment has been designed to simplify the displaying of the COMM LINE by the pressing of a single switch, and the lesson author can similarly capitalize on that convenience. This principle extends to many of the TACFIRE hardware features. In fact, the CONTROL mode was added to the PLANIT/TACFIRE system in order to make that kind of authoring possible.

Finally, user convenience is also dramatically affected by clarity of operating instructions that are given at the beginning of the lesson. Of course this is not unique to TACFIRE lessons, but perhaps the authoring burden is greater because of the added complexity of the terminal equipment. Figure 1 showed one such attempt at achieving an acceptable level of clarity. Notice frames 996 and 997 in particular. Their purpose is to provide an appropriate amount of information should the next lesson segment in the course sequence not be available for immediate execution. Without these frames, a simple branch to the next lesson would produce a cryptic PLANIT message about an illegal branch command at a given point in the lesson. However, the author can build into the lesson more definitive information, knowing the environment in which the lessons are to be taken.

CHECKOUT OF CONTROL MODE LESSONS

The first "student" of any CAI lesson is nearly always its author. The surest way to be sure that the presentation strategy follows the intended logic is to "play student." PLANIT not only provides the author the opportunity to check out lessons in this fashion, but enhances this opportunity by giving to the author freedom to execute any portion at will, to start over as many times as desired, to disregard previous trials and start out fresh again, to interrupt execution at any point and probe the logic of that portion of the lesson scenario, to change any part of the lesson and immediately test the effect of the change, and several other features which attempt to streamline this necessary checkout phase of authoring. None of the things mentioned can characterize the execution pattern for an authentic student, yet during the time that the author is "playing student," the system performs in a manner identical to that for the real student, but with those additional options.

The above paragraph attempts to show how important it is for an author to have execution capability which supercedes normal student execution, yet is identical to it where correspondence is important. It is at this point that several problems show up in the PLANIT/TACFIRE CONTROL Mode lesson scenarios and their potential for online checkout. Some of the problems have reasonably adequate solutions, but unfortunately, that is not necessarily true for all of them.

Checkout of PLANIT/TACFIRE Lessons on TACFIRE. The Enhanced Authoring commands which are interpreted in the CONTROL mode of PLANIT/TACFIRE execution orchestrate the manipulating and monitoring of the many features of the TACFIRE terminals. The only way an author has of being certain that the finished lesson scenario performs as it was intended to is to see it execute on that hardware. Thus, if lesson errors were made, the terminal will probably show it. Perhaps lights will be left on that were not intended, or they were not turned on, or a display is on the wrong screen, or it is missing altogether, or a screen has a leftover display on it when the lesson indicated that it was blank, or a variety of other similar things which are wrong would readily show up.

There are two kinds of problems in this kind of checkout. The first is the difficulty encountered by the author in entering and exiting the lesson scenario at any arbitrary point. This problem has a reasonably adequate solution through the use of the "environment" frames which were shown in Figure 1. With the inclusion of these frames, the

author regains much of the freedom of arbitrary execution which is lost in the CONTROL mode due to lack of opportunity of typing priviledge authoring commands. The authoring burden is somewhat greater since not only do these five frames need to be added to each lesson, but there must be continual sensing provision for the SPARE switch as well as a continual updating of the value in LINK(10).

The second kind of problem is probably the more serious. The TACFIRE hardware is simply not as available as authors would like it to be. Machines are not yet plentiful, and even apart from that, they only run PLANIT in a dedicated mode so that PLANIT authors must schedule time when the machine is idle.

Checkout of PLANIT/TACFIRE Lessons on Commercial Machines. PLANIT lessons destined for TACFIRE execution are more likely to be both authored and checked out on commercial equipment. This is simply because commercial equipment is more available.

The authoring of PLANIT/TACFIRE scenarios is no different on commercial equipment than on TACFIRE equipment. However, execution for purposes of lesson checkout is dramatically different. Some of the differences which will immediately be noticed on commercial terminals as compared to TACFIRE terminals include:

- No CONTROL mode
- One display screen (or paper printout) versus two screens and a printout
- No SA switches
- No provision for indicating illumination of switch caps
- No opportunity to merge displayed data with typed data on a single screen
- No comparable interpretation for SPECIAL calls
- No provision for multiple-line input
- No provision for input of selected column fields within a line
- No opportunity to view the "held-over" status of prior displays (from either screen)
- Inverted sequence of actions for responding to questions
- Impaired opportunity to insure that the seven-line screen requirements are being met

- No impact of the frame sequencing problem for the processing of typed answers as is the case in TACFIRE (thus the problem might exist in the lesson and be overlooked)

- Distortion of output due to the printing of command lines which are not intended to be printed (and will not be printed on TACFIRE)

- No comparable interpretation for preamble characters which will need to appear in some TACFIRE lessons

Every one of the above differences is potentially serious for the author who attempts to check out lessons on commercial equipment which are meant for TACFIRE. Consequences will range from abnormal lesson termination (e.g. from non-implemented SPECIAL calls) to the overlooking of potential problems on TACFIRE (because they do not cause the similar problem on the commercial equipment -- e.g. screen overflow).

Inconveniences in lesson checkout include the need to type switch mnemonics when the lesson instructs that the switch be pressed, needing to type complete lines which include format characters when the lesson only asks for the blanks to be filled in, inverting the process of responding (e.g. taking the switch action after typing the response on TACFIRE vs. entering the switch action mnemonic before typing the response on commercial equipment), not seeing switch cap illumination cues, etc.

The above problems make it extremely difficult for the author to check out PLANIT/TACFIRE lesson material on commercial equipment and deliver the finished lessons with much certainty that they will execute for students as they were intended. Some recommendations will be made which are designed to alleviate many of the checkout problems and even reinstate some of the lost convenience. However, these recommendations are placed in a later section so that some other recommendations can be made first.

The intent of the next section is to recommend some changes that could be made to the Enhanced Authoring facility which would effectively eliminate many of the current problems so that unnecessary efforts to find solutions could be avoided. Thus, when recommendations are made for solutions to the problems, the reader can weigh them against earlier recommendations which might do away with that particular problem altogether, given that a decision is made to modify the Enhanced Authoring system. In fact, a comparison of the recommendations for alleviating each of the problems might provide a sound basis on which to decide the next course of action.

RECOMMENDATIONS FOR CHANGING THE ENHANCED AUTHORING FACILITY

The recommendations proposed herein are not to be construed as "finished" and fully compatible with the TACFIRE system. There will need to be some additional communication with TACFIRE experts to resolve minor problems. However, the general model which will be presented is known to be sound and can be implemented.

The most confusing single difference in the authoring of Enhanced PLANIT/TACFIRE lessons as opposed to conventional PLANIT lessons is the sequence problem in the need to account for the input of the SA mnemonics in the lesson prior to analyzing the textual input. The effects of this enforced order of events accounts for most of the items on the list presented earlier in this document (page 15) of standard PLANIT language features which should be avoided in TACFIRE authoring (e.g. MC frames, non-lesson generated messages, interactive CALC, etc.). The reason given was that, following a given response, PLANIT would expect another line of text but would receive a line of switch mnemonics instead. This would throw the input sequence off up until the next line of switch mnemonics was expected. Meanwhile, the student would be receiving incomprehensible messages from PLANIT and would probably become thoroughly confused.

The general solution to this problem is to change the Enhanced Authoring facility such that the response to an "unconditioned" question is the top line of the CED screen. An "unconditioned" question is meant to refer to the default situation where no specific directions were given by the author regarding the content of the reply. In that case, there would be no further opportunity to examine the switch action which caused the transmission of the textual data.

The second part of the solution is to cause PLANIT/TACFIRE's MIOP to continually monitor the output text stream for the Enhanced Command prefix characters ($$$), in the NORMAL and the CONTROL modes, causing an automatic switch to the CONTROL mode when those characters are found at the beginning of a line of legitimate Enhanced Commands.

With this change, most of the remaining Enhanced Authoring language design would remain the same with the following exceptions:

- An SA-READ command would be added for the option of reading the SA mnemonics as is presently done by default

- The GET command can be used (in its present form) prior to the initial terminal read in a read sequence, resulting in the interpretation now given to the GET command. (The difference is that the GET could specify conditions for the initial read as well as subsequent ones).

- A new command, CH-???, be added where the ??? characters refer to any character triplet that should subsequently replace the $$$ triplet. (This would then permit lesson authors to print Enhanced command forms as they might like to do in lessons designed to teach Enhanced Authoring).

- Two new commands, ROLL-ON and ROLL-OFF, be added to change the method of processing the excess lines beyond the seven on the screen, treating them like in the NORMAL mode (for ROLL-ON), and discarding them for ROLL-OFF.

In a sense, two different (but similar) design change options are included above. The continual monitoring of the output text stream for the $$$ characters to switch to the CONTROL mode precludes any need for the ROLL-ON/ROLL-OFF commands. On the other hand, the ROLL-ON/ROLL-OFF commands could permit the doing away with the NORMAL/CONTROL distinction altogether. With appropriate default conditions (ROLL-ON, CE-WRITE, etc), conventional PLANIT lessons which contain no Enhanced Authoring commands might execute appropriately. However, this part might contain too many problems, e.g. VFMED preambles, when to blank screens, etc. The more sure recommendation would retain the present NORMAL/CONTROL dichotomy and include all of the above new features except ROLL-ON and ROLL-OFF.

Several PLANIT features were discussed earlier in this document which were inappropriate to include in PLANIT/TACFIRE lessons, most of which could again be used with their normal meaning under a system modified according to the above recommendations. Specifically, some of the things which would be reinstated include:

- Use of M frames

- Use of Conditional Q frames

- Conventional use of R: command

- Normal execution pattern for non-lesson generated messages

- Interactive use of CALC (and all the many things which that implies--e.g. REVIEW, GOTO, FINISHED, etc.)

- Reinstatement of dotted frame types for easier reentry back into the lesson following an interruption

- Enforced numeric responses in Q frames

Most of the special environment which was suggested in Figure 1 for lesson execution control could be eliminated. At most, an author might want the SPARE switch in the response list to cause the execution of a PLANIT FINISHED command for those instances where no opportunity is afforded for entering CALC to type "FINISHED".

There may be some objection to the loss of the switch mnemonics if they had not been requested on the first step of the read sequence (i.e. with the SA-READ command). This could be remedied in a couple of different ways. One way would be for the author to include the SA-READ command in the frame where that line might be missed, and store the line in a REPLY buffer for later analysis. Another way might be found through the addition of another Enhancement command (e.g. SA-REFRESH) which would cause the previously read switch status to be reformulated into a response line.

Authoring Enhanced lessons for PLANIT/TACFIRE execution is certainly considerably more difficult than conventional PLANIT authoring. In fact, it was probably the perception of that increase in difficulty which led to the present effort. The above recommendations provide some suggestions which, if resolved to a single coherent design, could reduce much of that difficulty and eliminate much of the need for added tutoring of present PLANIT authors to equip them for Enhanced authoring.

RECOMMENDATIONS FOR TACFIRE TERMINAL SIMULATOR

It is fairly well acknowledged that PLANIT lesson scenarios which are intended to be executed on TACFIRE hardware will most often need to be prepared on commercial computing hardware. Primarily, this is because of the lack of sufficient time on TACFIRE systems to dedicate to PLANIT authoring, but the advantage of commercial terminals over TACFIRE terminals for lesson authoring would be a factor as well. However, using the commercial equipment for the lesson checkout phase can prove to be inadequate as well as frustrating.

This section undertakes to recommend a software package which could be added to the commercial PLANIT installation to simplify the checkout process of Enhanced lessons.

Since all of the Enhancement provisions for the PLANIT/TACFIRE system are contained in MIOP, it follows that some comparable specially coded MIOP on a commercial machine could be made to resemble it. Thus, the special MIOP would also monitor the text streams, looking for the $$$ character triplets and the commands which follow, then take some appropriate action instead of allowing the lines to print. It is obvious that such a MIOP could not take all the same actions as the TACFIRE MIOP does since the screens, lights and switches are not physically present. (However, an elegant simulation might build these things into a box).

Instead, the simulation being recommended here would document the actions as they occur and make other provisions for simulating the TACFIRE terminal.

Display Documentation. The special MIOP (SMIOP) would take two courses of action regarding output information. First, it would retain a core image of the contents of each of the screens and the SA switches and lights, and the second action would be to annotate the display as the lesson executes. For example, a certain unused character would be co-opted for use in the documentation and to communicate directly with SMIOP. Let's assume that the special character is "!". Then some examples of the annotation might be:

```
!!!!!! FOLLOWING TEXT TO THE RD
!!!!!! PM SW. LIT
!!!!!! GET EXECUTED.  CHARACTERS ON NEXT LINE
TYPE:PERS  /INF   ;
```

Since SMIOP would also retain a core image of the contents of the screens and status of the switches, these could be displayed

by the author on command at any time, for example:

!RD -would display the current status of the RD screen.

!CED -would display the current status of the CED screen.

!SA -would display the status of the lights on the SA.

Thus, the author could at any time see what the TACFIRE terminal "looks like."

In addition to the above, SMIOP could keep running counts of the lines going to each of the displays, and could detect and print warnings when information is transmitted from the lesson which would be unreadable on the screen, either because it exceeded the screen size or because the screen was overwritten without giving the user a chance to read it.

Switch Actions. The author could take switch actions by using some other special SMIOP commands, for example:

!PM -would "push" the Priority Message switch.

!A-8 -would "push" the two Format Select switches.

!A-8 FC -would "push" the "A", "8" and FORMAT COMMAND switches.

Even though mnemonics were being used to push the switches, they would react in the usual TACFIRE fashion. That is, switches which are non-interrupting (e.g. Message Address switches, Format Selection Grid switches, etc.) would do nothing more than to change the internal status of SMIOP, while interrupting switches (e.g. PRIORITY MESSAGE, CYCLE MESSAGES, PAGE, etc.) would allow lesson execution to proceed. The mnemonics would be the same as those used in lesson construction, and several could be grouped on a line to represent a series of switch actions.

Keyboard Input. Strings of typed characters which simulate keyboard responses would be merged with the core image of the CED display, much as occurs in TACFIRE. The following examples show how full lines might be typed or blank fields filled in:

!1 THIS IS TEXT THAT WOULD BE INPUT ON LINE 1 OF THE CED
!4 THIS TEXT WOULD GO ON LINE FOUR.
!2,31 THIS TEXT WOULD BEGIN IN COLUMN 31 OF LINE TWO.

Note that typed text is non-interrupting. That is, the lesson would not advance after the typing of one or more such lines until an interrupting switch was mnemonically "pushed." The author could also display the CED screen at any time to see

what the merged version looked like. This action would not advance the lesson either.

With the above capability, blanks in formats could be filled in much more easily than typing the entire line (including the format) as is now the case. Not only that, when the interrupting switch was finally "pushed", the entire CED screen would be sent, just as TACFIRE does, and notice also that the switch action would then follow the typing as it is supposed to (but doesn't now).

Finally, one additional convenience would make use of a default condition for typed lines. Lines which have no "!" prefix would be treated as a full line typed on line one of the CED, and the pressing of the physical return key on the terminal would count (in that case alone) as the "pushing" of an interrupting switch. Thus, simple one-line responses could be typed into the TACFIRE simulator with the same ease that they are now typed into commercial PLANIT lessons. In order to cover the necessary bases, there would be one additional command which would allow the user (author) to establish which interrupting switch is being simulated in that instance, and change it whenever necessary, for example:

!CR-PM

Then, the following line:

THIS IS MY ANSWER.

when ended with a normal carriage return, would be transmitted on the top line of the CED followed by the "pushing" of the PRIORITY MESSAGE switch, all without further action. Note that this one feature alone would permit the same SMIOP to also be used for NORMAL mode PLANIT/TACFIRE lessons, so long as the answer line did not begin with the "!" character.

Coding The Special MIOP (SMIOP). MIOP must be coded locally to install PLANIT in any case, so SMIOP could be coded instead. However, there is a better way. MIOP represents the last stage in coding the logic of PLANIT before the actual device with which data communication takes place comes into view. MIOP is a subroutine that PLANIT calls upon by name, but for which no code exists until it is coded locally. Therefore, it would be possible to develop SMIOP in two modules, one that performs the TACFIRE simulation, and the other that communicates with devices. In this case, the first module (the one simulating TACFIRE) would retain the name, MIOP, because PLANIT expects a subroutine by that name to exist. The second module we will call CMIOP, and that module will only be called from MIOP.

CMIOP will be coded in exactly the same fashion that MIOP currently is; only the name will be changed. It will therefore be the Current MIOP.

The new MIOP will now link PLANIT to CMIOP, but in doing so, will perform the character stream monitoring functions and process the special command forms (using the "!" prefix). Thus, for any present PLANIT installation, the procedure for changing the installation to a TACFIRE simulator would be to change the name of the current MIOP to CMIOP, and link in the new MIOP with it.

The installation could be changed back and forth in a few minutes simply by changing subroutine names and re-linking, or more simply by including a software switch in the new MIOP so that is does nothing but pass data untouched for conventional operations.

Given the above configuration for MIOP and CMIOP, it would then be feasible to code the new MIOP in the same machine independent format that is now used for the PLANIT source code. This would provide a machine transferrable version of a PLANIT/TACFIRE simulator. Installation procedures for this PLANIT version would be identical to the present except that there would be one extra module to Generate, and there would be a name change for the subroutines (MIOP and CMIOP). These would be trivial differences. Thus the present installation information would also be valid for that version as well.

The new simulation code package (which must be called MIOP, but is not to be confused with the present MIOP) would be completely independent of the internal operations of the PLANIT code since the interface between PLANIT and the new MIOP would of necessity be the same as the current interface between conventional PLANIT/MIOP installations, and this would in turn be the same as the interface between the new MIOP and CMIOP. This would be analogous to disconnecting an electric drill from a wall outlet, plugging it into a speed controller and plugging the speed controller into the wall outlet. It just adds a new module for control while the interface requirements remain the same.

With this kind of modularity, new versions of PLANIT could be mated with PLANIT/TACFIRE simulation installations just as is now done in conventional installations.

Finally, with TACFIRE simulation as an objective, the PLANIT system would be generated using parameters which match, as closely as possible, those used to generate the TACFIRE hardware version. Thus, the simulator would provide the author with an accurate picture of the manner in which the new lesson will perform on the TACFIRE hardware, and it just might be possible that lessons which have been checked out in this fashion would execute flawlessly on TACFIRE.

RECOMMENDATIONS FOR PROJECT REORIENTATION

This report is being submitted at the end of the eighth month of the twelve-month project. The remainder of the project was originally intended to be devoted primarily to the development of guidelines for Enhanced authoring.

It is easy to see from the foregoing that such a document will need to be extensive. That fact was probably never questioned by either party to the contract even from the outset. Note that the guidelines would also need to provide information for checking out PLANIT TACFIRE lessons on commercial installations, at least as well as they can presently be checked. This is also not a small task.

The size of the task that the guidelines might represent is not the present issue. The real question is whether to proceed with a significantly costly task which would be applicable to a system which is not nearly as good as it ought to be -- or could be.

If any or all of the recommended changes were made to the Enhanced authoring system, it would significantly change the information which should go into the authoring guidelines regarding how to construct scenarios. If TACFIRE terminal simulator efforts were to be undertaken, there would be little incentive to document the present "Mickey Mouse" routine for checking out Enhanced lessons on commercial equipment, since it would all change (and be simplified).

There seems to be at least three good reasons not to proceed with the writing of the Authoring Guidelines for the current system as originally intended:

- The recommendations for Enhanced authoring changes would provide so many authoring advantages that they ought to be considered, and if the changes were made, any Authoring Guidelines written now would be invalidated,

- The recommendations for a TACFIRE terminal simulator seems to be worth serious consideration if the intent is to use commercial computer equipment to prepare TACFIRE lesson scenarios, and if that is implemented, authoring guidelines prepared now concerning lesson checkout would be completely off, and

- A comprehensive Author's Guide for the current system has been nearly completed already by Litton Data Systems which could easily serve interim needs until the disposition of these recommendations is decided.

In place of the Authoring Guidelines, two new tasks would seem to be in order.

Task 1, New Lesson Conversion. In the confidence that the recommended changes to the Enhanced Authoring facility will be looked upon with some interest, it is likely that there might also be interest in having a sample lesson available which reflects the proposed authoring format. This would provide clear illustrations of authoring under that system. Further, it is recommended that one of the present sixteen lessons which have already been converted to the current format be chosen to be re-converted to the proposed format. Then the reader would not only have the new authoring illustrated, but would also have the old and new to lay side-by-side, making judgments easier about the desirability of making the change.

This task would then consist of designing a tentative set of cohesive changes to the present Enhanced Authoring language (several options were given earlier), documenting the format and functional description of that set, and then converting a lesson (among the current sixteen Fire Mission lessons) to the modified language standards.

Task 2, TACFIRE Terminal Simulator Design. This task would encompass two subtasks, both made necessary by the proposed simulator characteristics.

The first subtask would be to design language and execution procedures which would completely simulate the TACFIRE ACC terminal. Examples of this design were given in the recommendation, using the "!" prefix and mnemonics. However, they were incomplete and not completely thought out. The complete design would need to be checked with one who was expert in the functioning of that terminal. A similar design would eventually be needed for other TACFIRE terminals, but this would provide a manageable start, and even permit software requirements to be realized.

The second subtask would be to work out the necessary design for linking the proposed program modules, PLANIT, MIOP and CMIOP. I am confident that it can be done, yet it will be the first time that two independent programs, each coded in PLANIT's machine independent format, will need to co-exist and work together. There are some known design problems which will need to be worked out. Each will need its own data space while also interfacing with the data space of the other. Each will have to schedule terminal events, but with the same terminal in view. It is a puzzle for which a solution is known to exist because the operations are well-defined and logical, but the solution is yet to be found.

If the above two tasks appear to be a better course of action for the remainder of the project, then there is good reason to believe that they can be accomplished, even though the remaining four months would likely be insufficient. Actually, by the time the recommendations are read, considered and a decision is made, less than four months will be left. However, since the current effort has spent at a rate which is less than originally projected, sufficient funds remain to extend the project for an extra two or three months. This would provide enough time.

The intention of all of the above suggestions is not to obtain a change of signals mid-stream. Rather, we are all cognizant that the final objective is to provide a facility whereby authors can prepare training scenarios in as easy a manner as possible on readily available commercial hardware which have the highest probability that can be obtained of running correctly when put on the TACFIRE equipment. I believe that the recommendations provide for such significant improvements in those regards that a re-evaluation of our current course of action is essential.

REFERENCES

1. PLANIT Author's Guide, TM-(1)-4422/001/01, Northwest Regional Educational Laboratory, October, 1970

2. PLANIT Language Reference Manual, TM-(1)-4422/002/01, Northwest Regional Educational Laboratory, October, 1970

3. PLANIT Language Extensions Through Version 2.8, Northwest Regional Educational Laboratory, March, 1976

4. Design Description Document for PLANIT System Enhancements, 125200-901, Litton Data Systems, April, 1977

5. PLANIT Author Handbook for the TACFIRE Advanced Training Program, 148002-903, Litton Data Systems, In press

DISTRIBUTION

1 US ARMY WESTERN COMMAND ATTN: APPE
1 DEPARTMENT OF THE NAVY TRAINING ANALYSIS AND EVALUATION GROUP
1 HQDA ATTN: DAAG-ED
1 US PACIFIC FLEET HUMAN RESOURCE MANAGEMENT DETACHMENT
1 HQ, ICATA ATTN: ATCAT-OP-Q
2 HQDA RESEARCH AND STUDIES OFC
1 MILITARY OCCUPATIONAL DEVELOPMENT DIV DAPC-MSP-O, RM 852C HOFFMAN BLDG 1
1 HQDA OFFICE OF THE CHIEF OF CHAPLAINS
4 OASD (MRA AND L)
1 HQDA ATTN: DAPC-PMP-S, HOFFMAN BLDG 1
1 6970TH AB GP BD 9801 SOCIAL ACTIONS OFC
1 HQDA ODCSPER
1 USA AVIATION SYSTEMS COMD ATTN: DRSAV-ZDR
1 EQUAL OPPORTUNITY BRANCH ATTN: AFZT-PA-HR
1 USA ARRADCOM ATTN: ATFE-LO-AC
1 HEADQUARTERS, US MARINE CORPS ATTN: CODE MPI-20
2 US ARMY EUROPE AND SEVENTH ARMY
1 CHIEF, ATTITUDE & OPINION SURVEY DIVISION ATTN: ATZI-NCR-MA, HOFFMAN BLDG II
2 HQ TRADOC TECHNICAL LIBRARY
1 MILITARY OCCUPATIONAL DEVELOPMENT DIRECTORATE ATTN: ATZI-NCR-MS-M, RM 3N33 HOFFMAN BLDG II
1 DATA ANALYSIS DIVISION ATTN: ATZI-NCR-MD, HOFFMAN BLDG II
1 USA MILPERCEN ATTN: DAPC-POO-T
1 HQDA ARMY FORCE MODERNIZATION COORDINATION OFFICE
1 HQDA ATTN: DASG-PTB
1 123D USARCOM RESERVE CENTER
1 US ARMY SOLDIER SUPPORT CENTER /
1 DIRECTORATE OF TRAINING ATTN: ATZQ-T
1 DIRECTORATE OF COMBAT DEVELOPMENTS ATTN: ATZQ-D
1 HQDARCOM MARINE CORPS LIAISON OFC
1 DEPARTMENT OF THE ARMY US ARMY INTELLIGENCE & SECURITY COMMAND
1 USA MISSILE COMMAND ATTN: DRSMI-NTN
1 US ARMY CECOM ATTN: DRSEL-ATDD
1 USA FORCES COMMAND
1 US MILITARY DISTRICT OF WASHINGTON OFC OF EQUAL OPPORTUNITY
1 NAVAL CIVILIAN PERSONNEL COMD SOUTHERN FLD DIV
22 ARI LIAISON OFFICE
1 7TH ARMY TRAINING COMMAND
1 HQDA, OCS STUDY OFFICE
1 U.S. NAVY TRAINING ANALYSIS EVALUATION GROUP
1 USACDEC ATTN: ATEC-EX-E HUMAN FACTORS
1 ATTN: SM-ALC/DPCR
1 USAFAGOS/TAC SENIOR ARMY ADVISOR
1 INTER-UNIV SEMINAR ON ARMED FORCES & SOC
1 OASA (RDA) DEPUTY FOR SCIENCE AND TECHNOLOGY
1 OFC OF NAVAL RESEARCH /
1 AFHRL/LRT
1 AFHRL/LRLG
1 AIR FORCE HUMAN RESOURCES LAB ATTN: AFHRL/TSR
1 FEDERAL AVIATION ADMINISTRATION CENTRAL REGION LIBRARY, ACE-66
1 NAVY PERSONNEL R AND D CENTER /
1 NAVY PERSONNEL R AND D CENTER DIRECTOR OF PROGRAMS
2 OFC OF NAVAL RESEARCH PERSONNEL AND TRAINING RESEARCH PROGRAMS
1 NAVAL PERSONNEL R & D CENTER /
1 OFC OF NAVAL RSCH ORGANIZATIONAL EFFECTIVENESS PRO.
1 NAVAL AEROSPACE MEDICAL RSCH LAB AIRBORNE RANGER RESEARCH
1 DEPT. OF NATIONAL DEFENCE DEFENCE AND CIVIL INSTITUTE OF ENVIR MED
1 NAVAL AEROSPACE MEDICAL RSCH LAB AEROSPACE PSYCHOLOGY DEPARTMENT
1 USA TRADOC SYSTEMS ANALYSIS ACTIVITY ATTN: ATAA-TCA
1 HEADQUARTERS, COAST GUARD CHIEF, PSYCHOLOGICAL RSCH BR
1 6970 ABG (SL)
1 USA TRAINING BOARD ATTN: ATTG-ATB-TA

1 USA MATERIEL SYSTEMS ANALYSIS ACTIVITY ATTN: DRXSY-C
1 USA RESEARCH OFC /
1 NAFEC HUMAN ENGINEERING BRANCH
1 BATTELLE-COLUMBUS LABORATORIES TACTICAL TECHNICAL OFC
1 USA ARCTIC TEST CEN ATTN: AMSTE-PL-TS
1 USA COLD REGIONS TEST CEN ATTN: STECR-OP
1 USA CONCEPTS ANALYSIS AGCY ATTN: CSCA-RQP
1 DEFENSE LANGUAGE INSTITUTE FOREIGN LANGUAGE CEN
1 HQ WRAIR DIV OF NEUROPSYCHIATRY
1 USACAC ATTN: ATZL-CAC-IA
1 USACACDA ATTN: ATZL-CAC-A
1 USA ELECTRONIC WARFARE LAB CHIEF, INTELLIGENCE MATER DEVEL + SUPP OFF
1 USA RSCH DEVEL + STANDARDIZA GP, U.K.
1 NAVY PERSONNEL RSCH + DEVEL CENTER ATTN: (CODE 307)
1 USA RESEARCH AND DEVELOPMENT LABS CHIEF, BEHAV SCIENCES DIV, FOOD SCI LAB
1 USAARL LIBRARY
1 HUMAN RESOURCES RSCH ORG (HUMRRO) /
1 SEVILLE RESEARCH CORPORATION
1 USA TRADOC SYSTEMS ANALYSIS ACTIVITY ATTN: ATAA-SL (TECH LIBRARY)
1 UNIFORMED SERVICES UNIT OF THE HEALTH SCI DEPARTMENT OF PSYCHIATRY
1 BATTELLE REPORTS LIBRARY
1 FEDERAL AVIATION ADMINISTRATION ATTN: CAMI LIBRARY ACC-44D1
1 GRONINGER LIBRARY ATTN: ATZF-RS-L BLDG 1313
1 CENTER FOR NAVAL ANALYSIS
1 NAVAL HEALTH RSCH CEN LIBRARY
1 NAVAL PERSONNEL R AND D CEN LIBRARY ATTN: CODE P106
1 HQ, FT. HUACHUCA ATTN: TECH REF DIV
1 USA ACADEMY OF HEALTH SCIENCES STIMSON LIBRARY (DOCUMENTS)
1 SCHOOL OF SYSTEMS AND LOGISTICS /
1 ERIC PROCESSING AND REFERENCE FAC ACQUISITIONS LIBRARIAN
1 DEPARTMENT OF THE NAVY TRAINING ANALYSIS AND EVALUATION GP
1 NATIONAL CENTER FOR HEALTH STATISTICS /
1 USMA DEPT OF BEHAVIORAL SCI AND LEADERSHIP
1 OLD DOMINION UNIVERSITY PERFORMANCE ASSESSMENT LABORATORY
1 USA COMMAND AND GENERAL STAFF COLLEGE ATTN: LIBRARY
1 USA TRANSPORTATION SCHOOL USA TRANSP TECH INFO AND RSCH CEN
1 USA ADMINCEN TECHNICAL RESEARCH BRANCH LIBRARY
2 HQDA USA MED RSCH AND DEVEL COMMAND
1 USA FIELD ARTY BD /
1 NAT CLEARINGHOUSE FOR MENTAL HEALTH INFO PARKLAWN BLDG
1 U OF TEXAS CEN FOR COMMUNICATION RSCH
1 INSTITUTE FOR DEFENSE ANALYSES
1 USA TRAINING SUPPORT CENTER ATTN: ATIC-DST-PA
1 USA MOBILITY EQUIPMENT R AND D COMMAND ATTN: DRDME-ZG
1 HQ, USA MDW ATTN: ANPE-OE
1 DA US ARMY RETRAINING BDE RESEARCH + EVALUATION DIVISION
1 DANVILLE RESEARCH ASSOCIATES, INC.
1 USA AEROMEDICAL RESEARCH LAB SCIENTIFIC INFORMATION CENTER
1 HUMAN RESOURCE MANAGEMENT CEN, SAN DIEGO
1 USAFA DEPT OF BEH SCI + LEADERSHIP
1 US MILITARY ACADEMY DEPT. OF HISTORY, BLDG 601
1 USA INTELLIGENCE CEN AND SCH ATTN: SCHOOL LIBRARY
1 MARINE CORPS INSTITUTE
1 NAVAL SAFETY CENTER /
1 US COAST GUARD TNG CEN ATTN: EDUCATIONAL SVCS OFFICER
1 USAAVNC AND FT. RUCKER ATTN: ATZQ-ES
1 US MILITARY ACADEMY DIRECTOR OF INSTITUTIONAL RSCH
1 USAADS-LIBRARY-DOCUMENTS
1 USA SERGEANTS MAJOR ACADEMY ATTN: LEARNING RESOURCES CENTER
1 USA INTELLIGENCE CEN AND SCH ATTN: ATSI-DT-SFL
1 USA ARMOR SCHOOL ATTN: ATZK-TD
1 NAVAL POSTGRADUATE SCH ATTN: DUDLEY KNOX LIBRARY (CODE 1424)

```
  1 USA TRANSPORTATION SCHOOL  DEPUTY ASST. COMMANDANT EDUCA. TECHNOLOGY
  1 USA SIGNAL SCHOOL AND FT. GORDON  ATTN: ATZH-ET
  1 USA ARMOR CEN & FT. KNOX  OFFICE OF ARMOR FORCE MGT & STANDARDIZATION
  1 USA SIGNAL SCHOOL & FT. GORDON  EDUCATIONAL TECHNOLOGY DIVISION
  1 HQ AIC/XPTD  TRAINING SYSTEMS DEVELOPMENT
  5 USA INTELLIGENCE CEN AND SCH  ATTN: ATSI-ERM
  1 US ARMY ARMOR CENTER  ATTN: ATZK-TD-PMO
  1 USA QUARTERMASTER SCHOOL  DIRECTORATE OF TRAINING DEVELOPMENTS
  1 US COAST GUARD ACADEMY  /
  1 USA TRANSPORTATION SCHOOL  DIRECTORATE OF TRAINING & DOCTRINE
  1 USA INFANTRY SCHOOL LIBRARY  /
  1 USA MP & CHEM SCH/TNG CEN & FT. MCCLELLAN  ATTN: ATZN-PTS
  1 USA MP & CHEM SCH/TNG CEN & FT. MCCLELLAN  DIR: COMBAT DEVELOPMENT
  1 USA MP & CHEM SCH/TNG CEN & FT. MCCLELLAN  DIR: TRAINING DEVELOPMENT
  1 USA MP & CHEM SCH/TNG CEN & FT. MCCLELLAN  ATTN: ATZN-MP-ACE
  1 USA INSTITUTE OF ADMINISTRATION  ATTN: RESIDENT TRAINING MANAGEMENT
  1 USA FIELD ARTILLERY SCHOOL  MORRIS SWETT LIBRARY
  1 USA INSTITUTE OF ADMINISTRATION  ACADEMIC LIBRARY
  1 USA WAR COLLEGE  ATTN: LIBRARY
  1 USA ENGINEER SCHOOL  LIBRARY AND LEARNING RESOURCES CENTER
  1 USA ARMOR SCHOOL (USARMS)  ATTN: LIBRARY
  1 US ARMY INTELLIGENCE CENTER & SCHOOL  ATTN: ATSI-TP
  1 US ARMY INTELLIGENCE CENTER & SCHOOL  ATTN: ATSI-TD-PM
  1 US ARMY INTELLIGENCE CENTER & SCHOOL  ATTN: ATSI-ES
  1 DEPARTMENT OF THE AIR FORCE  AIR UNIVERSITY LIBRARY (ATC)
  1 USA CHAPLAIN CENTER & SCHOOL  ATTN: ATSC-TD-OD
  1 USA CHAPLAIN CENTER & SCHOOL  ATTN: ATSC-TD-ED
  1 USA CHAPLAIN CENTER & SCHOOL  ATTN: ATSC-TD-SF
  1 USA CHAPLAIN CENTER & SCHOOL  ATTN: ATSC-DOS-LLC
  1 HQ TRADOC  TRAINING DEVELOPMENT INSTITUTE
  2 BRITISH EMBASSY  BRITISH DEFENCE STAFF
  2 CANADIAN JOINT STAFF
  1 CDLS (W) LIBRARY
  1 FRENCH ARMY ATTACHE
  1 AUSTRIAN EMBASSY  DEFENSE, MILITARY AND AIR ATTACHE
  3 CANADIAN DEFENCE LIAISON STAFF  ATTN: COUNSELLOR, DEFENCE R AND D
  1 ROYAL NETHERLANDS EMBASSY  MILITARY ATTACHE
  1 CANADIAN FORCES BASE CORNWALLIS  ATTN: PERSONNEL SELECTION
  2 CANADIAN FORCES PERSONNEL APPL RSCH UNIT
  1 ARMY PERSONNEL RESEARCH ESTABLISHMENT
  6 LIBRARY OF CONGRESS  EXCHANGE AND GIFT DIV
  1 DEFENSE TECHNICAL INFORMATION CEN  ATTN: DTIC-DDA-2
140 LIBRARY OF CONGRESS  UNIT DOCUMENTS EXPEDITING PROJECT
  1 US GOVERNMENT PRINTING OFC  LIBRARY, PUBLIC DOCUMENTS DEPARTMENT
  1 US GOVERNMENT PRINTING OFC  LIBRARY AND STATUTORY, LIB DIV (SLL)
  1 THE ARMY LIBRARY  ATTN: ARMY STUDIES SEC
  3 /  /
  1 /  /


NUMBER OF ADDRESSEES  171

TOTAL NUMBER OF COPIES 355
```