AN IMPROVED DMAP CAPABILITY

David L. Herendeen

Universal Analytics, Inc.
Playa Del Rey, California

SUMMARY

A set of improvements has been designed and implemented into a test version of the NASTRAN DMAP (Direct Matrix Abstraction Program) compiler. These modifications simplify the use of the DMAP control language while enhancing its power and versatility. The implemented changes are described and examples are presented to illustrate their use.

INTRODUCTION

Traditionally, the NASTRAN engineer-user has not felt comfortable with the DMAP capability. Its style is oriented to the programmer. It uses complicated semantic constructs that are overly formalized. Also, its implementation retains many undesirable and confusing restrictions.

Modifications have been made to the DMAP compiler to alleviate many of these disadvantages. Removal of certain restrictions has unlocked powers that have been imprisoned in the system since its inception. Full downward compatibility with currently operational DMAP has been retained.

These changes fall into three overlapping categories:

Improved Syntax

Removal of Restrictions

Extension of Capability

Improvements falling into the first two categories allow for the inadvertent introduction of errors into DMAP programs that previously would not be allowed. To protect the user, the extended capabilities include an error-handling facility, and a new class of POTENTIALLY FATAL ERRORs is defined. Thus, the system protects users from costly errors arising from incorrect DMAP while allowing the user freedoms he could not otherwise enjoy.

37

IMPROVED SYNTAX

Several cosmetic changes have been made by removing many of the semantic inconveniences. These syntax improvements include:

1. Abbreviation of the parameter section to eliminate the redundant use of the C, V, and N specifications. For instance, the parameter section of a DMAP instruction

   /C,N,2/C,N,STATICS/V,N,NLØAD/V,N,PARM=1.0

   now may be written as

   /2/*STATICS*/NLØAD/PARM=1.0

2. Allowing the use of default parameters not only at the end of the parameter section, but also internal to it

   /2/PARM=1.0///4.5//NLØAD

3. Elimination of the need for trailing commas in both the input and output data block name lists when they are not required.

   TABPT EST,,,,// $

   now may be written

   TABPT EST// $

4. An automated checkpoint option controlled by a "predefined checkpoint" (PRECHK) declaration has been implemented. An analyst may generate all required checkpoint instructions by a single statement such as

   PRECHK A,B,C,D,F,X $

   Even more convenient forms have been implemented,

   PRECHK ALL $
   PRECHK ALL EXCEPT A,B,C $

   This extension still allows the user to place the conventional CHKPNT instruction throughout his program.

5. An automated SAVE specification to be included within the parameter section of the module instruction rather than a separate declaration.

   MØDX A,B/C/V,N,P1/V,N,P2 $
   SAVE P2 $

now may be written

MØDX A,B/C/P1/S,N,P2 $

The effects of (4) and (5) above are much more than superficial. When combined and used appropriately, they reduce the length of DMAP routines (i.e., Rigid Format 1) by as much as 30%.

This more concise code is superior in structure and is made more readable by eliminating many statements that are not essential to the solution flow. Such structure allows for more rapid understanding of the solution algorithm. The automatic SAVE prevents the accidental misplacement of the SAVE instruction that formerly caused a dump. Such a misplacement is now handled correctly and does not cause an abnormal termination.

Though most of the above changes are cosmetic in nature, the improvements described next offer significantly improved capabilities.

REMOVAL OF RESTRICTIONS

There are many capabilities intrinsically available via the DMAP language that cannot be exploited because of the arbitrary restrictions imposed by the current DMAP compiler. The following changes have been made to remove these restrictions and greatly enhance the power of the language:

1. Data blocks may now appear as input before they appear as output.

2. Data blocks may now appear more than once as output.

3. The REPT instruction has been modified to allow a variable number of loops to be determined during execution. This usage is exemplified by

   REPT LØØPTØP,NLØØP $

   where NLØØP is a parameter output by some module prior to the REPT instruction.

4. The severely restrictive positional requirements of the EQUIV instruction have been eliminated.

The removal of the first two restrictions and the addition of the variable REPT instruction opens new vistas for the DMAP user. In addition, change (2) allows locally used data block names to be reused later; thus the efficiency of file allocation within the NASTRAN system is increased. The positional requirements of the EQUIV module that have accounted for many confused hours of debugging have been eliminated. The single requirement is that the primary data block of the EQUIV instruction must appear as output prior to the EQUIV.

The DMAP user now has the tools with which to design and execute sophisticated iterative algorithms. Such facilities have been long desired within the NASTRAN environment for solving complex nonlinear problems which involve either single or nested looping.

EXTENSION OF CAPABILITY

New capabilities have been added that on the surface appear to be very simple, yet they have profound effects on the flexibility of file operations, on user understanding, and on program control. These are

1. The implementation of two modules, CØPY and SWITCH.

2. A new technique for user-specified compiler options, including a detailed cross reference of data block names, module names, and variable parameter names, and a complete and detailed ØSCAR dump.

3. Specification and control of the extended error-handling facility.

The CØPY and SWITCH modules are of great use in iterative DMAP routines. CØPY performs a complete physical copy from the specified input data block to the output data block. SWITCH operates directly on the FIAT (FIle Allocation Table) to interchange data block names. It does not copy data.

The combination of these modules greatly simplifies and shortens DMAP looping. For example, in Figure 1, the DMAP code for raising a matrix [Q] to the rth power is shown in the original DMAP and again in the improved and extended DMAP. The simplification that results is immediately apparent.

Special compiler options may be elected, not through the DIAG, but by a new instruction XDMAP (Execute DMAP). Convenient options include: GØ/NØGØ to allow a compilation only; LIST/NØLIST for DMAP compilation listing; DECK/NØDECK to request the punching of the DMAP program; REF/NØREF to provide a cross reference; and ØSCAR/NØØSCAR to print a detailed listing of all ØSCAR records generated by the compiler. The ØSCAR (Operation Sequence Control ARray) is the "object" code that DMAP generates and is used to direct the actual computational flow. Also activated by the XDMAP instruction is the extended error-handling facility described below. These features are easy to use and helpful to both programmers debugging DMAP programs and engineers/analysts having to write their own routines or make local modifications to existing DMAP.

The extended error-handling facility justifies the removal of restrictions and improvement of capabilities in that it assures the user that gross logic errors in DMAP programs will not be costly. This feature helps the user diagnose and solve any problems caused by the new semantics. A new class of POTENTIALLY FATAL ERRORs warns the user to review his operations carefully. The user can select the error level (WARNING, POTENTIALLY FATAL, or FATAL) at which termination of the job will occur. Examples of XDMAP instructions are

```
XDMAP  GØ,LIST,ERR=3 $
XDMAP  NØGØ,LIST,REF,ØSCAR,ERR=1 $
```

Figure 2 shows a composite illustration of the detailed cross reference, while Figure 3 is an excerpt from an ØSCAR listing. Both of these figures are derived from the Level 15.5 Rigid Format 1.

CONCLUSION

Changes have been designed and implemented into the DMAP compiler that have improved the syntax and simplified DMAP usage for the engineer/analyst. Also, convenient mechanisms have been added to extend the file manipulation features. Finally, the removal of arbitrary compiler restructions has unlocked implicitly existing capabilities which greatly enhance the flexibility of the language.

ACKNOWLEDGMENTS

The author wishes to express his appreciation to the NASTRAN Systems Management Office (NSMO) for sponsoring this effort. The work was performed by Universal Analytics, Inc., and monitored by McDonnell Douglas Astronautics Company - West. The author also wishes to thank Messrs. D. Herting and R. Hoesly for their valuable insight and diligent assistance in this effort.

OLD SYNTAX

```
 1    BEGIN      $
 2    MATPRN     Q,,,, // $
 3    PARAM      // C,N,NØP / V,N,TRUE=-1 $
 4    PARAM      // C,N,SUB / V,N,RR / V,Y,R=-1 / C,N,2 $
 5    PARAM      // C,N,NØP / V,N,FALSE=+1 $
 6    CØND       ERRØR1,RR $
 7    ADD        Q, / QQ $
 8    LABEL      DØIT $
 9    EQUIV      QQ,P / FALSE $
10    MPYAD      Q,QQ, / P / C,N,0 $
11    EQUIV      P,QQ / TRUE $
12    PARAM      // C,N,SUB / V,N,RR / V,N,RR / C,N,1 $
13    CØND       STØP,RR $
14    REPT       DØIT,1000000 $
15    JUMP       ERRØR2 $
16    LABEL      STØP $
17    MATPRN     P,,,, // $
18    EXIT       $
19    LABEL      ERRØR1 $
20    PRTPARM    // C,N,-1 / C,N,DMAP $
21    EXIT       $
22    LABEL      ERRØR2 $
23    PRTPARM    // C,N,-2 / C,N,DMAP $
24    EXIT       $
25    END        $
```

NEW SYNTAX

```
 1    BEGIN      $
 2    MATPRN     Q // $
 3    PARAM      // *SUB* / RR / V,Y,R=-1/2 $
 4    CØND       ERRØR1, RR $
 5    CØPY       Q / P $
 6    LABEL      TØP $
 7    SWITCH     P,QQ // $
 8    MPYAD      Q,QQ / P / 0 $
 9    REPT       TØP,RR $
10    MATPRN     P $
11    EXIT       $
12    LABEL      ERRØR1 $
13    PRTPARM    // -1 / *DMAP* $
14    END        $
```

Figure 1.- Raising a matrix Q to the power R.

LEVEL 2.0 DMAP COMPILER - DMAP CROSS REFERENCE LISTING

DATA BLOCK	DMAP STATEMENT NUMBERS							
BGFDT	0004*	0006	0015	0025	0040	0095	0119	0123
CASECC	0015	0053	0095	0119	0123			
CSTM	0004*	0006	0025	0032	0035	0040	0095	0119
DIT	0032	0035	0095	0119				
DM	0057	0059	0092*	0093	0100			
ECPT	0025*	0030	0032	0035				
ECT	0007*	0008	0009	0025				
EDT	0095	0119						
ELSETS	0009*	0015	0019	0123				
EPT	0025							
EQEXIN	0004*	0006						
EST	0025*	0030						
GEI	0025*	0030						
GEOM1	0004							
GEOM2	0004	0007						
GEOM3	0020							
GEOM4	0053							
GM	0057	0059						
GO	0057	0059						
GPCT	0025*	0030						
GPCT	0004*	0006						
GPL	0004*	0006						
GPSETS	0009*	0015						
GPST	0029	0030						
GPTT	0020*	0024						
KAA	0077	0078						
KFF	0071	0072						
KFS	0057	0059						
KGGX	0032*	0033						
KGG	0044	0045						
KLL	0083	0084						
KLR	0057	0059						
KNN	0058	0059						
KOO	0057	0059						
KRR	0057	0059						
KSS	0057	0059						
LLL	0089*	0090						
LOO	0057	0059						
MGG	0023	0024						
MPT	0032	0035						
OEF1	0119*	0120						
OES1	0119*	0120						
OGPST	0029	0030						
OGPWG	0040*	0041						
OPG1	0119*	0120						
OQG1	0119*	0120						

PARAMETER	TYPE	DMAP STATEMENT NUMBERS					
CARDNO	I	0041	0042	0062	0063	0120	0121
COUPMASS	I	0035					
CPBAR	I	0035					
CPQDPLT	I	0035					
CPQUAD1	I	0035					
CPQUAD2	I	0035					
CPROD	I	0035					
CPTRBSC	I	0035					
CPTRIA1	I	0035					
CPTRIA2	I	0035					
CPTRPLT	I	0035					
CPTUBE	I	0035					
EPSI	I	0103					
GENEL	I	0025					
GRDCNT	I	0022					
IRES	I	0103					
JUMPPLOT	I	0009					
LUSET	I	0004					
MPCF1	I	0053					
MPCF2	I	0053					
NOA	I	0053					
NOBGG	I	0035					
NOELMT	I	0027					
NOGENL	I	0025					
NOGPDT	I	0004					
NOGRAV	I	0020					
NOK4GG	I	0032					
NOL	I	0053					
NOMGG	I	0035					
NOSET	I	0053					
NOSIMP	I	0025					
NOSR	I	0056					
NSIL	I	0009					
NSKIP	I	0050					

MODULE NAME	DMAP STATEMENT NUMBERS							
COND	0014	0028	0031	0034	0038	0039	0046	0055
	0099	0106	0112	0116	0122			
EXIT	0139							
GPSP	0061							
GPWG	0040							
GP1	0004							
GP2	0007							
GP3	0020							
GP4	0053							
JUMP	0051	0114	0127					
MATGPR	0107	0108						
MCE1	0066							
MCE2	0068							
OFP	0041	0062	0120					
PARAM	0022	0027	0050	0056	0115			
PLOT	0015	0123						
PLTSET	0009							
PRTMSG	0011	0017	0125					
PRTPARM	0129	0131	0133	0135	0137			
RBMG1	0086							

Figure 2.- Composite illustration of the DMAP cross reference.

LEVEL 2.0 NASTRAN DMAP COMPILER - OSCAR LISTING

```
**********************************************************************
OSCAR RECORD NUMBER   2     MODULE TYPE =  1
MODULE NAME -  GP1          DMAP INSTRUCTION NO.   4

        SUMMARY OF INPUT DATA BLOCKS( 3 )
                GEOM1      0/    2/0/    2
                GEOM2      0/   17/0/    5
                ********INPUT DATA BLOCK  3 IS NULL

        SUMMARY OF OUTPUT DATA BLOCKS( 6 )
                GPL        0/   99/0/   55
                EQEXIN     0/  108/0/    5
                GPDT       0/   99/0/   47
                CSTM       0/  104/0/   22
                BGPDT      0/  108/0/   13
                SIL        0/  108/0/   13

        SUMMARY OF PARAMETERS( 3 )
                LUSET   (I  )          0
                CONSTANT(I  )        123
                NOGPDT  (I  )          0

**********************************************************************
OSCAR RECORD NUMBER   3     MODULE TYPE =  4
MODULE NAME -  XSAVE        DMAP INSTRUCTION NO.   5

        PARAMETERS TO BE SAVED( 1 )
                LUSET          1

**********************************************************************
OSCAR RECORD NUMBER   4     MODULE TYPE =  4
MODULE NAME -  XCHK         DMAP INSTRUCTION NO.   6

        DATA BLOCKS TO BE CHECKPOINTED( 6 )
                GPL        EQEXIN     GPDT       CSTM       BGPDT      SIL

**********************************************************************
OSCAR RECORD NUMBER   5     MODULE TYPE =  1
MODULE NAME -  GP2          DMAP INSTRUCTION NO.   7

        SUMMARY OF INPUT DATA BLOCKS( 2 )
                GEOM2      0/   17/0/   17
                EQEXIN     0/  108/0/    7

        SUMMARY OF OUTPUT DATA BLOCKS( 1 )
                ECT        0/   22/0/    7

**********************************************************************
OSCAR RECORD NUMBER   6     MODULE TYPE =  4
MODULE NAME -  XCHK         DMAP INSTRUCTION NO.   8

        DATA BLOCKS TO BE CHECKPOINTED( 1 )
                ECT

**********************************************************************
OSCAR RECORD NUMBER   7     MODULE TYPE =  1
MODULE NAME -  PLTSET       DMAP INSTRUCTION NO.   9

        SUMMARY OF INPUT DATA BLOCKS( 3 )
                PCDB       0/    7/0/    7
                EQEXIN     0/  108/0/   13
                ECT        0/   22/0/   22

        SUMMARY OF OUTPUT DATA BLOCKS( 4 )
                PLTSETX    0/    9/0/    9
                PLTPAR     0/  108/0/   13
                GPSETS     0/  108/0/   13
                ELSETS     0/  108/0/   13

        SUMMARY OF PARAMETERS( 2 )
                NSIL    (I  )          0
                JUMPPLOT(I  )          0

**********************************************************************
OSCAR RECORD NUMBER   8     MODULE TYPE =  4
MODULE NAME -  XSAVE        DMAP INSTRUCTION NO.  10

        PARAMETERS TO BE SAVED( 2 )
                NSIL           1
                JUMPPLOT       2

**********************************************************************
OSCAR RECORD NUMBER   9     MODULE TYPE =  2
MODULE NAME -  PRTMSG       DMAP INSTRUCTION NO.  11

        SUMMARY OF INPUT DATA BLOCKS( 1 )
                PLTSETX    0/    9/0/    9

**********************************************************************
OSCAR RECORD NUMBER  10     MODULE TYPE =  2
MODULE NAME -  SETVAL       DMAP INSTRUCTION NO.  12

        SUMMARY OF INPUT DATA BLOCKS( 1 )
                ********INPUT DATA BLOCK  1 IS NULL
```

Figure 3.- Excerpt from an ØSCAR listing.