TECHNICAL AND SOCIAL IMPACT OF NASTRAN

Thomas G. Butler
NASA Goddard Space Flight Center

3

SUMMARY

Some estimates are made as to the direction in which a new generation of general purpose applications programs can be expected to migrate. NASA has released its new general purpose structures program, NASTRAN (NASA STRuctural ANalysis). Predictions are made as to the impacts that this newly available space technology is liable to have within the field of structural engineering and on the society in which it interacts.

INTRODUCTION

NASTRAN is the first of a new generation of general purpose application programs for individual disciplines. It was established for the discipline of structural analysis. There is a strong possibility that other disciplines can adapt the NASTRAN format for a general purpose approach to their problems, and it is therefore timely to discuss this topic in an open forum such as the *NASTRAN Colloquium.*

The primary feature of NASTRAN as a general purpose management framework that should make it attractive for other disciplines is its freedom from semantic implications. It can manage a host of problem types without having the individual natures of these disciplines influence the logic of the management of the associated problems during the solution process on the computer. The basis for this statement will become apparent in the following discussion on characteristics.

CHARACTERISTICS

Computer Independence

All interfacing between the problem solution (Functional) Modules and the computer operating system is confined to a small section of the program called the "Executive." This allows the program to operate with and under the computer operating system. The Executive is organized about a central driver that stays resident in core. The central driver calls Executive Modules into core, consults a tape stored scheduler (called by the acronym OSCAR, which stands for Operating Sequence Control ARray) for the initiation of each module, and consults a file status table for peripheral service needs. The central driver also consults core resident tables that do the bookkeeping for file and core assignments and that contain the values or names of parameters, depending on whether they are constant or variable. These parameters either exert problem control or act as communication links between modules. The Executive Modules intercede for a Functional Module in invoking support from the operating system, such as reading and writing, or peripheral storage assignments. Naturally, some of these Executive Modules must be machine dependent, but these are a small fraction (about 1 percent) of the total program code so that there is a minimum of machine dependence.

The other design feature that has contributed to machine independence is the almost exclusive writing of Functional Modules in FORTRAN and using FORTRAN wherever possible (even in writing the machine-dependent Executive Modules). The FORTRAN language used was derived from the intersection of IBSYS FORTRAN IV, OS 360 FORTRAN IV, UNIVAC's FORTRAN V, and CDC's FORTRAN. Certain exceptions were allowed, e.g., in the area of nonstandard returns. Hence an interpreter is needed to generate the source decks for the CDC 6600 from the basic library.

The gross link/overlay design is essentially the same on each machine, but here again there are certain inherent uniquenesses that have to be accommodated. Thus for almost each computer model (not only for each computer manufacturer) there will be an individual strategy to build the architecture of the NASTRAN executable. The dimensions of NASTRAN level 15.0 are roughly 14 links wide with a depth of 7 levels of overlay in each link. To achieve seven levels of overlay on the 6000 class of CDC machines, a separate linkage editor/loader had to be written to replace that provided by the SCOPE operating system.

Internal Storage Management

Initially, the operating system will make an assignment of core and an assignment of a combination of secondary storage devices with both serial and random access devices. A combined minimum of 30 serial and random access files is required for NASTRAN. From this point on, NASTRAN does its own internal dynamic management of files and core. File requirements are defined in terms of predetermined sets of data (called data blocks). Depending on the peculiarities of a given structural problem, a given physical file during one problem execution may have been occupied by a large succession of data blocks entirely under the control of a NASTRAN Executive Module. Each Functional Module is overlaid into core in a tightly packed fashion so that the maximum amount of remaining core is available as working space for the storage of matrices. This management of core and files, it must be reemphasized, is internal and is under the control of an Executive Module.

Modularity, Open-Endedness, and Maintainability

The working basis of NASTRAN is through the companion pairs, modules, and data blocks. It takes the execution of many modules and several link control transfers to achieve a total cycle of a problem type relating to a given discipline. It takes a number of subroutines to compose the many functional steps of a module through several levels of overlay. A predetermined set of input data blocks, output data blocks, and scratch files belongs to each module. Information sources for data blocks can be tables or matrices. In nearly every instance, these various working parts have been made open ended. For instance, the number of links can be extended, the number of levels of overlay can be increased, the number of problem types can be expanded, the number of modules can be enlarged, the number of entries in tables can be augmented, the quantity of tables can be opened up, and the number of subroutines can grow. In rare instances, operational considerations have circumscribed the entries in a table, but these are sufficiently rare that it is fair to say that NASTRAN is generally open ended.

The modular design allows for program maintenance. Each module communicates only with the Executive and is not allowed to communicate directly with another module. Any communication that is needed between modules is accomplished by passing information to output data blocks that are used as input data blocks to succeeding modules or by

passing parameter values to parameter tables that can be accessed by a succeeding module. In this way, the interface with the Executive is fixed module by module so that an entire module can be replaced or revamped, so long as it maintains the same interface as previously. Hence no internal module change of this type has a cascading effect through the program. This modularity also allows for an internal input/output routine to process internal quantities in a machine-independent fashion. As the technology advances, new subroutines and modules can be added or inserted in place of old ones and the program can be kept modern without any organizational modification.

The matrix operations performed by the modules do not require that the amount of core working space be of a size to hold an entire matrix or set of matrices at once. Solution logic allows only portions of each matrix in an algebraic step to be present at any one time while the remaining portions are fed from secondary storage as operations on preceding portions are completed. This type of core usage vis-a-vis matrix operation is given the name "spill logic" and allows the program to operate on matrices of virtually unlimited size.

Modules

Beside Executive Modules, the bulk of the program is composed of three kinds of modules: Utility, Mathematical, and Functional. Most of these are useful for any discipline. The Utility Modules are certainly discipline independent (having to do with printing matrices, performing coordinate transformations, etc.) and need no further discussion. The Mathematical Modules were written to perform operations on large matrices, because those routines provided by standard operating systems are not efficient enough to be functional for large matrices. "Large" is defined to be a combination of order and density* so that the matrix will not fit in core even in packed form. A number of Mathematical Modules were written to take matrix characteristics into account, such as sparsity, bandedness, type (real or complex), or positive definiteness. Mathematical Modules are immediately adaptable to other disciplines.

Functional Modules currently favor structural mechanics, but conceivably they would be useful to other disciplines under different names. They have names such as Geometry

*Density is a ratio of the nonzero elements to the maximum possible number of matrix elements.

Processors, Constraint Eliminators, Elastic Properties, Mass Properties, Damping Properties, Weight Generators, Load Generators, Simultaneous Algebraic Equation Solver, Eigenvalue Extraction, Stress Data Recovery, Piecewise Linear Analysis, and Differential Equation Integration. If there are other functions that other disciplines require, they can be generated and made part of a problem sequence, then their existence can be noted in the Module Property Table.

Internal Compiler and Scheduler

The heart of its ability to adapt this Executive management to other applications programs resides in the way it organizes the functions to be performed. Various problem types peculiar to a given discipline (and in this case structural mechanics) are classified into categories such as statics, eigenvalue analysis, dynamic analysis, random analysis, and others. The steps needed to accomplish the solution of a problem of a given type are presented in an orderly sequence of module executions and Utility Executive operations. The statements of these solution steps are written in an internal language called "DMAP" (for Direct Matrix Abstraction Process). An internal reference library of statements for each problem type has been established with appropriate cataloguing. In the spirit of open endedness there is no requirement as to problem types to be kept in the library, so a given discipline (or combination of disciplines) can have as many problem types catalogued in the library as is convenient to the user. No matter to what discipline a set of statements belongs, the Executive merely calls out the set of statements by catalogue number according to an analyst's command.

In response to a user's call in the control section for a particular problem type, the Executive selects the corresponding sequence of DMAP statements and directs this sequence to the DMAP compiler. The output from the compiler in effect becomes the scheduler of module and utility executions. All loops are unwound and all jumps are translated into a purely serial string; all associated data block names and their file storage needs are tabulated in the scheduler. This scheduler, the OSCAR, is written onto tape and does not occupy core or disc space. The Executive central driver consults with OSCAR at the end of each module before calling a succeeding module. It is now evident how the Executive can operate on any discipline without semantic implications.

There are two operations that the Executive performs in conjunction with the problem type and the control section, to be discussed later. These operations have semantic implication and are only quasi-Executive operations. They are called, respectively, the Input File Processor (IFP) and the Output File Processor (OFP). The IFP checks on the legality of input bulk data and the logic of control section statements and organizes these input data into data blocks for eventual processing by the Functional Modules. The OFP does sorting on the solution vectors and organizes these results into formats that are suitable for a particular discipline. The IFP and OFP would have to be replaced entirely for the Executive to manage another discipline.

Control Versus Sets

The system to be analyzed is treated as the basic entity for purposes of defining a problem. In the case of structures the basic system is the geometric arrangement among elastic members. Most other quantities affecting the problem are less basic and are liable to frequent change, such as loads on the system, or constraints on the system, or partitions on the system from which response information is desired. The scheme for managing these varying conditions is based on sets.

As many sets of varying conditions that one desires may be assembled in a pool of information. These sets will remain dormant until they are individually activated. The analyst consequently is in a position to exercise quite precise control in the Case Control Section of the program by creating subcases wherein any of a particular load, a particular constraint, a particular spectrum, a particular sequence of time steps, or a particular series of output quantities and output locations can be activated from the data pool merely by specifying the appropriate set identifiers. As many subcases can be prepared for a single computer submittal as is convenient for the analyst.

The quantities that fall under the surveillance of the Case Control Section are kept in a table. In keeping with the open-ended spirit of the program, the table can be augmented so that unique quantities pertaining to other disciplines can be accommodated.

Another hierarchy of control is exercised to distinguish among major options within the program. This does not operate on the basis of sets but by fixed names. This portion is called the Executive Control Section.

Plotter Independence

All interfacing between the solution results and the plotters that will be used to display them is confined to a section of the program called the Plot Module. The sorting by sets, the projection, the orientation, the color, labeling, and the general commands to produce these effects are accomplished without regard to the display hardware. A small translator subroutine will convert these plot commands from general internal expression to specific commands for execution on individual plotting machines. In this instance again, the general purpose character has been preserved with essential plotter independence.

ADAPTATION TO OTHER DISCIPLINES

The ease with which the current NASTRAN general purpose organization can serve other disciplines will be examined by major program units. The reason for adaptation is to provide a general purpose applications program as opposed to a special purpose program.

Executive

The general purpose character of the Executive Section can make it immediately adaptable to other disciplines. The link/overlay management amply provides for the needs of most applications. The lean organization of having only a central driver permanently resident in core along with its vital tables, while all other Executive functions are provided by modules that are brought in temporarily to perform their jobs and then vacate core, is a concept that warrants preserving. The interrelationships between the Executive operations and computer operating systems have already been made and should certainly not be duplicated unnecessarily. When other computers have been added to the NASTRAN set, these too will become publicly available.

Modules

The particulars of a given discipline with respect to Functional Modules are considered here. It appears that if a discipline is based on potential theory, just as structural mechanics is based on the linear theory of elasticity, the existing Functional Modules could nicely serve merely by a change of name. If this is so, the existing forms could be preserved in their applications to another discipline simply by providing a new mask for input card mnemonics, headings of the output listings, and terms in the diagnostic messages. Because electrical engineering is based upon the potential theory of electromagnetics, we can draw

on analogies in NASTRAN. The existing Mass Properties Module could apply to an Inductance Property Generator; the existing Elastic Property Module could apply to a Capacitance Property Generator; and the existing Damping Property Module could apply to a Resistance Property Generator. Geometry Processor Modules could apply to the description of the Circuit/Nodal Pattern, while the Load Generator Module could apply to the External Voltage Array (or the Potential Gradient) at Nodes. Similar reasoning can be used in optics, electronics, and fluid flow. Presuming the universality of some of the Functional Modules, it is germane to return to the Executive Modules. IFP's can be preserved with respect to data input card reading and data block generation. Adaptation to electrical engineering would require a new mask for the terms in diagnostic messages concerning legality of input.

Compiler

One would not expect too much similarity between the way problem types are formed in different disciplines. It is reasonable to assume then that the catalogue of statements referring to the module executions for various problem types would be completely replaced. Because the sequence of statements is written in DMAP language, the creation of a new library of problem types is relatively trivial. The compiler and the generation of the OSCAR would remain intact. Quite possibly some new Functional Modules may want to be added to the existing structural set to round out the field for a given discipline, such as the counterpart to the Stress Data Recovery Module. Each discipline will probably have unique collateral characteristics recoverable from the solution vector.

In summary, a new discipline may provide a new name for the solution vector, such as acidity, voltage, humidity, light intensity, or population instead of elastic deformation, but most of the working portion of the program can probably be preserved with only a replacement mask for the names assigned to data.

The important factor in the ability to convert the program to other disciplines is the general purpose management approach in the Executive. Even though a few new modules and all new problem types will need to be added, the open-ended design will allow them to be easily integrated. Of course, it is a grubby job to comb through the code to prepare a new language mask, but the point to be made is that it is a relatively minor task compared to the generation of an entirely new general purpose applications program.

One of the huge benefits of adapting an existing general purpose applications program like NASTRAN to another discipline is the freedom from a major debugging exercise. The corollary advantage is the reliance on future maintenance that NASA is currently practicing.

IMPACT ON THE STRUCTURES COMMUNITY

In narrowing the topic of discussion to general purpose programs in structural engineering only, it is interesting to indulge in some philosophical reflections.

Some sort of analysis has always been part of the design process in structural engineering, but until recently, analyses have been limited to the engineer's ability to solve only simpler approximations to the real structure and the usual attempt was to solve a pair of such approximations to bracket the real case. Analysis has advanced to the point where a very close representation of complicated three-dimensional structures is now possible. Because there was uneven advancement of structural analysis capability by problem type, for years there was a tendency to limit one's analytical skill to one or two specialties. A sufficiently broad range of structural problem types has advanced enough to warrant the investment in general purpose programs. Having access to general purpose programs, the structural analyst at long last is abandoning his fragmented approach to analysis and is applying a cohesive approach. As a result, analysis is now coming into its own as a vital part in the decision-making process of structural design.

The aerospace industry is a heavy user of analysis for structures. Here a large variety of tried-and-true special purpose programs does most of the analytical tasks. The full range of structural problems is generally being dealt with analytically in the aerospace industry. Engineers, however, are finding an increasing need to communicate analytical data. They would like to feed data that are being output from one special purpose program as the input to another special purpose program; but they find that they are frustrated by the lack of compatibility. A general purpose program on the other hand has the solution capability of the combined special purpose programs with no commensurability nor compatibility problems. In addition, the continual updating of some of the major general purpose programs is causing general purpose programs to enjoy increased popularity.

The civil engineering profession is rapidly moving toward analysis as an everyday tool. Static analyses for stress and deflection responses dominate the activity in civil

engineering problems; however, vibration analysis is on the increase. Only an occasional attempt is made at solving transients and most of these are from seismic excitation. Because their problems can become quite large, civil engineers are making use of general purpose programs.

Machinery manufacturers show the most diversity in their attitude toward analysis. The automotive and turbine industries head the list of those pledged to analysis; mining is the least active. As to the range of problem types, heavy industry is also the most varied. Boiler people are about the only ones doing random analyses. Rotating machinery designers are analyzing for vibrations. Statics is definitely the most popular. In several years, it is expected that heavy industry will become more involved with analysis.

Economics tends to prevent the use of analysis in consumer goods. Only on occasion when a sticky local problem is annoying him will a consumer goods manufacturer ask an engineering analyst to bail him out. Usually such problems are solved by consultants rather than by in-house engineers. It is not easy to say how big a part analysis will play in the design cycle of consumer goods in the years to come.

As part of the structures community, universities are showing an increased awareness of production analytical tools as opposed to those that concentrate on academic niceties. Probably general purpose programs will have the most far-reaching impact on engineering colleges and on the way that the engineering profession is practiced. The structural field has been fragmented into civil engineers, mechanical engineers, and aeronautical engineers. In industry the profession was further fragmented into dynamicists, thermal specialists, hydraulic engineers, propulsion specialists, weights engineers, and stochastics specialists. With the advent of general purpose programs, the tools are at the disposal of every engineer, enabling him to readily inquire into a host of problems. General purpose analytical capability will give rise to the education and training of engineers with broader backgrounds. Eventually they will become structural engineers spanning the whole range from statics through thermal to dynamics.

SOCIAL IMPACT

Increased emphasis on analysis is bound to have an influence on deemphasizing some activities and promoting still others. The extensive development of general purpose

programs is symbolic of the maturity of analysis. As such, a revolution of sorts is being set into motion. The social impact of this revolution will be the realignment of the work force.

Until recently, the structural engineer has depended upon testing to serve two masters: as a proof of design and as a proof of manufacture. Constant improvement in instrumentation and test equipment has enhanced the dependency on testing. As an engineer becomes more proficient at analysis, he will find that the cost of using analysis for design certification will be far less than that of testing and that the time lapse between the completion of a design and the performance results will be far shorter. Lastly, he will discover that he will be able to inquire into the behavior of locations that were too remote to be accessible by instrumentation. The triple impact of less cost, less time, and more complete information will wean designers away from testing for design certification. We predict that analysis will eventually take its proper place so that the roles of analysis and testing can be encapsulated as–

Analysis is for proof of design.

Testing is for proof of manufacture.

This does not mean, of course, that no testing will be used for proof of design. For instance, photoelastic testing of machine parts with severe stress gradients will certainly remain as a practical check of design. For the most part, however, it is safe to say that analysis will invade the province of testing as the primary tool for checking on designs.

What will be the social impact of this shift of emphasis away from testing? It means that less mechanical test equipment will be manufactured, fewer testing laboratories will survive, and fewer testing personnel will be employed. The demand for the manufacture of test articles will slacken, with its attendant reduction in the need for model builders.

As the dependency on analysis increases, the need for digital programmers will be felt in the areas of debugging, advanced module writing for updating programs, the writing of pre-processors and post-processors to take the drudgery out of data handling, and the intensification of the writing of support routines for on-line interactive graphics. The net effect will be increased business for software houses and their increased employment of programmers.

Probably the design cycle will change to the extent that some design tasks will be done by the computer instead of being strictly the province of the designer. Conceivably for highly complicated structures, the task of producing the initial design would devolve on a highly creative designer. Subsequent tasks would be performed by structural analysts on the computer, first to analyze, then to synthesize in response to the characteristics shown by analysis, and last to optimize for achieving the most acceptable design with respect to a parameter such as cost or weight. Probably a minimum of demand for detailing the parts of the conceptual design will be made during the computer phase of the design evolution. At the termination of the computer phase, designers and detail draftsmen will be needed to make final manufacturing drawings. The social impact of reemphasis on the computer during the design stage will be the increased demand for highly creative designers and decreased demand for routine designers and detail draftsmen. There will also be an increase in demand for engineers who are structural analysts.

The shift in emphasis toward analysis should not have a depressing effect on the small design office or small consulting businesses. Even though the small offices cannot afford to install their own computers, they can nevertheless operate all of the latest analytical tools in support of these tasks. Fortunately, hardware technology has kept pace with software technology and the little man did not get caught in the squeeze. Computer terminals have come to the rescue of the small design offices. They can communicate via the terminals to the central computers operated by such companies as McDonnell Automation, CDC's Cybernet, CSC's Infonet, and IBM's Service Bureau. A rather complete library of structural analysis programs is available at these central computer installations to support the range of problems that the remote offices need to solve.

The social impact of this remote terminal support will be to preserve small business and to stimulate competition. Small design and consulting firms will be maintained and will possibly grow. Business from these numerous smaller firms should contribute to the incentives for competition among terminal manufacturers and among central computer operators. The demand for repairmen should offer new employment.

Finally, the increase in structural analysis has produced a further need for plotting equipment. The computer printout is an orderly form of record-keeping but is too

awkward for digesting the quantity of data that gush from these new analytical programs. Plotters and cathode ray display tubes are starting to fill this need. More and more clever ways of presenting plotted data are abetting programming employment. This mode of data reduction is also helping the plotter manufacturer and his staff of repairmen as well.

In presenting the social side of the trends in structural analysis, we plead that we have painted only a qualitative picture. We have not braved the more risky path of reducing these social effects into quantitative values in manpower or in a time frame. Nevertheless, we firmly believe that a minor revolution is currently in motion and that this revolution will not reach its full development for another 5 years. NASTRAN is an important influence in provoking this ferment. It is well checked out and available to the general public at a nominal cost. Its general purpose character and its degree of machine and plotter independence make it particularly attractive to those who are looking toward longevity of methods.