DESIGN OF A MODULAR PROGRAM FOR TRANSIENT NONLINEAR ANALYSIS OF LARGE 3-D BUILDING STRUCTURES

by

Sanjeev Srivastav[1] and John F. Abel[2]

December 30, 1987

Technical Report NCEER-87-0027

NCEER Contract Number 86-4011

NSF Master Contract Number ECE 86-07591

1 Graduate Research Assistant, Dept. of Structural Engineering and Program of Computer Graphics, Cornell University

2 Professor, Dept. of Structural Engineering, School of Civil and Environmental Engineering, Cornell University

NATIONAL CENTER FOR EARTHQUAKE ENGINEERING RESEARCH
State University of New York at Buffalo
Red Jacket Quadrangle, Buffalo, NY 14261

ABSTRACT

This report describes the plans for the development of a computer modelling and analysis system for transient fully nonlinear seismic analysis of large buildings. The emphasis is on modelling the buildings at a "macro" level, i.e., modelling in terms of elements which correspond to structural members such as beams, shear walls, connections, and the like. It is recognized that the research needs are vast with advances required in geometric modelling, finite-element modelling, material modelling, nonlinear analysis, and interactive-adaptive user interface schemes. Further, the magnitude of the computational problem in terms of computer memory and solution time requires access to supercomputing or some parallel processing option, preferably with a graphical "window" for monitoring the computations. It is felt that coordination and cooperation among different research groups is essential to avoid duplication of effort and to accelerate research. With this objective in mind, a shareable platform or testbed computer program is proposed and is currently under initial development at Cornell University. A high degree of modularity is a central objective in the development of this program to permit expansion, porting, and modification over its useful life. One of the keys to modularity is the availability of an efficient data management system, which can manage data associated with the relationships among the different components of the building, their attributes, and the methods of manipulating the attributes.

ACKNOWLEDGEMENTS

The preparation of this report and the associated research were made possible by funding from the National Center for Earthquake Engineering Research, Grant Number 86-4011. The work has been carried out at the Program of Computer Graphics and the Department of Structural Engineering, both at Cornell University. Other faculty investigators for this project are Professor William McGuire, Professor Donald P. Greenberg, and Dr. Christopher H. Conley. Any opinions expressed herein are those of the authors only and do not reflect the views of sponsors or colleagues.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SECTION 1

INTRODUCTION

High speed digital computing has revolutionized structural analysis techniques. Nevertheless, major gaps remain in the modeling of the response of buildings and other structures subjected to static and dynamic loading. Much research has been conducted in the areas of finite elements, material modeling, and numerical methods. However, normally only highly simplified two- and three-dimensional models are used for the analysis of most framed building structures. Current analysis techniques may be adequate for much routine design but cannot be considered accurate enough to provide a response history to match the real full-range behavior of structures. Understanding of such nonlinear behavior as limit states and energy dissipation is necessary for development and evaluation of improved design tools and for the design of unusual or critical buildings.

1.1 Background

The reasons for the shortcomings in complete building modelling for earthquake analysis are multifarious. Firstly, one must recognize the

limitations of a deterministic model for predicting the response of a system especially when the critical loading due to earthquakes cannot be predicted accurately. That, however, does not undermine the value of carefully selected deterministic analyses as useful tools available to the engineer for the understanding of building behavior. Secondly, the accuracy with which a structure can be modeled depends on two factors: (1) An understanding of the structural behavior of the elements of the structure and their assemblage, and (2) Availability of the tools needed for simulation of the problem, including a suitable numerical modeling method (such as the finite-element method), the solution algorithms, and the computational resources required.

An accurate model for the full range of dynamic behavior of a three-dimensional building requires an enormous amount of computational power and model sophistication. Such a model needs to reflect correctly the effects of all significant aspects of the behavior including structural elements (beams, columns, infill elements, floors, connections, etc.), material and geometric nonlinearities, loading conditions, and soil-structure interaction on a "macro" level. On a "micro" level, effects such as cross-sectional distortions, spreading plasticity, cracking, crushing, creep, residual stresses, bond slip, tension stiffening, and confinement need to be considered.

Design engineers have typically relied on the results of linear elastic analyses modified by some empirical rules to account for nonlinear

effects. This philosophy is also reflected in most design codes. However, with the analysis techniques and computing resources available now, nonlinear analysis of structures can be employed for a more detailed response simulation. A fully nonlinear analysis of the structure, i.e., one employing geometric and material nonlinearity provides the closest modelling of the type of behavior that is known to occur in buildings. Several computer programs have been developed for the transient time-history analysis of buildings using either a linear-elastic model [WILS75][DOVE80] or accounting for geometric and material nonlinearities [KANN73][GUEN77], and have been widely distributed.

Further research is required in all of the areas mentioned above, especially in the areas of nonlinear modelling. In view of this fact, there exists the need for a modular system of analysis which can draw upon the current wealth of information and be flexible enough to include not only a wide gamut of simulation and solution strategies but also evolving innovations. A shareable platform or testbed is required for conducting research on analytical modelling of buildings under both dynamic and static loadings.

Associated with the growth in sophistication of the building model and solution algorithms, is the need for efficient software development. The use of data management and result interpretation tools along with structured programming techniques lead to the development of modular and

efficient computer programs. Such programs provide synergy to the development of better modeling techniques and algorithms by easing the burden of coding and modifying the program for different solution strategies. Such software resources can be shared among research environments and help avoid duplication of programming effort.

Further, one has to move away from stand-alone programs for analysis, design, pre- and post-processing and aim for integrated systems. The development of such a system requires modularity for the programs to be efficient. The graphical pre- and post-processors along with the interactive analysis programs currently known as CU-PREPF, CU-STAND, and CU-QUAND [PESQ83][PESQ84][CAST85][SUTH86] and a few other in-house programs developed at Cornell can be considered precursors to the integrated analysis and design systems of the future. Such user-friendly computer programs along with the vast and yet expanding computing power available today are a significant aid to designers and researchers.

Current research in the Department of Structural Engineering and the Program of Computer Graphics has been focused on both the "micro" and "macro" aspects of modeling structural behavior. The continuity of this parallel effort is being maintained under the sponsorship of the NCEER. The development of this new program is another step primarily to advance the macromodeling effort. However, many aspects of this program can also support future research on micromodeling.

1.2 Objectives of this Program

The initial scope of this program encompasses static and dynamic fully nonlinear analysis of structures. Further additions in the area of design capability would, however, be natural. The proposed program is envisioned to be primarily a tool for use in research environments. The major areas in which research can be aided by such a program are:

1. Modeling of various building elements and components
2. Material modeling: nonlinear behavior and damage modeling
3. Algorithms: parallel or vectorized
4. Interaction modeling: interaction among components or systems of buildings
5. Structural behavior, damage, and survivability
6. Design studies and evaluation
7. Correlation with, and design of static, pseudo-static, and dynamic experiments

One of the major features of this program is the independence of its various functional routines. It is intended to develop clearly identifiable modules for performing specific operations. This provides a test-bed environment in which it is easy to make modifications to the program by "plugging-in" alternative algorithms or theoretical formulations. The objective here is to avoid writing a program for, say, any one specific type of algorithm or using one specific element type. The modules in the program represent different algorithms, element libraries, material models, or numerical solution techniques.

System specific features are isolated in a few sections, thereby enhancing portability to other computing environments.

Different explicit, implicit, and, possibly, mixed time-integration schemes for the transient response are implemented. In terms of element models, inclusion of elements to model the "non-structural" and other continuous components to represent shear walls, floors, etc. are given a high priority. Numerical analysis techniques for solving linear and nonlinear equations are modularized so that it becomes possible to compare, say, direct versus iterative algorithms or Newton versus secant methods for nonlinear solutions.

To ensure ease of expansion and modularity, a well-organized database is established for the main analysis program as well as for the graphics code. Efficient data retrieval, flexibility, and minimum duplication are prime considerations in designing the database.

Computer graphics tools are used for developing an interactive environment and to facilitate data input and output. The availability of "state-of-the-art" graphics hardware and new software tools, such as menu managers, aid in the software development. Though most of the current frame analysis programs at Cornell use black and white vector displays, it is apparent that high-resolution color raster devices will soon be available for use in this application.

Research in the area of parallel processing for nonlinear frame dynamics is already under way at Cornell. Parallel processing provides a relatively inexpensive alternative to computing on large scale vectorized machines such as the Cray supercomputers. It may, in fact, be the only feasible alternative to overcome the limitations of the processing speeds of sequential processors. Algorithms for concurrent solution techniques are, however, still in primitive stages of development and influenced by various factors such as the configuration of processors, the operating systems of the computers, whether memory is shared or not, and the geometry of the structure. The option for using multiple processors on some types of hardware configuration is available to the user of this proposed program.

In brief, the objective is to produce a system for simulating the fully nonlinear dynamic response of structures in the most realistic manner possible. In doing so, initially the best of existing modeling techniques, algorithms, and computer hardware are to be adopted. In the future, new algorithms and models are to be developed and substituted into the modular program as the needs arise and as research progresses in these areas. Also, as new hardware and computing environments evolve, the program should be able to adapt to, and take advantage of, the improved capabilities.

1.3 Objectives of this Report

The purpose of this report is to record the plans for and objectives of the research-oriented computer program now under development at Cornell. As stated above, it is intended that this program be a shareable platform or testbed for ongoing and future research on the simulation of building behavior and on building design. Therefore, the dissemination of this report has the goals of, first, informing colleagues in research and practice of both the plans and the developments in progress and, second, eliciting comments, suggestions, and inquiries from colleagues. The research needs are so great that they transcend the capacity of a single institution or project. Nevertheless, unnecessary or undesirable duplication of effort is to be avoided. It is therefore hoped that the sharing of the information and ideas in this report will foster coordination and cooperation in future research.

SECTION 2

SCOPE AND CAPABILITIES OF THE PROGRAM

The scope and capabilities of the planned modular program encompass several aspects: element library, material modelling, loading schemes, analysis types, solution procedures, computing environments, and visualization. Each of these aspects is discussed in the following subsections.

2.1 Element Library

Beam-column type finite elements have traditionally been used in modelling buildings. The use of these elements has been based on the premise that the main structural components of a building superstructure are its beams and columns. Other types of structural configurations which utilize shear walls and continua cannot be accurately analyzed using such a frame analysis program. The contribution of floors, infill panels, etc. to the structural behavior has often been only approximately simulated through the use of analogous braced frames [STAF81], equivalent struts [STAF69], the equivalent frame analogy [MACL76], the equivalent wide-column analogy [JENK66] and other methods.

Other techniques include modelling the frames and the shear walls as independent substructures, and having them connected through diaphragm links rigid in their own plane.[WILS75][GUEN77] In general, the behavior of a building can be better modeled by using an assemblage of assorted elements such as the beam-column ("line" type) elements and other "surface" type elements. Therefore, finite elements with two or more nodes are needed to model the various structural components of the building.

The coupling of diverse elements such as the beam-column elements, plane-stress elements, plate or shell type elements is not a straightforward issue either. Realistic modelling of the imperfect continuity or separation between the walls and the framework and other elements involves appropriate modelling for the interface between the finite elements representing these structural elements.

Since these elements have different numbers of nodes and degrees of freedom at each node, problems of compatibility and consistency of the FEM formulation are encountered. An additional issue relates to the methods of organizing data storage associated with each of these element types and must be addressed in writing the proposed computer program.

Infill elements with translational as well as rotational degrees of freedom can be implemented for use in a general frame analysis program. Beam-column elements are usually formulated with six degrees of freedom at each node - three translations and three rotations. However, if a

conventional 8-noded isoparametric plane stress element shared nodes and a side with the beam-column element, inter-element compatibility is not satisfied. It is preferable to have all the elements used in the program to be formulated with, say, six degrees of freedom at each node (disregarding initially other possible degrees of freedom like warping of the cross-section). This alleviates the problem of determining which degrees of freedom associated with a particular element are non-existent at any one node.

Using one or a few finite elements for every beam, column, shear panel or wall within a bay and a storey provides a macromodel of the building in that it is good as an overall behavior model. Finer details of the response such as the yielding patterns within the elements require a micromodelling effort where each of the macromodel elements is replaced by a large number of finite elements. To keep the computational requirements within bounds while attempting a refined analysis, micromodelling is performed on subassemblages of the original structure. In fact, initial experience indicates that a fully detailed micromodelling of a subassemblage can require more computational resources than the macromodelling of the entire building.[WHIT88]

2.2 Material Modelling

A nonlinear analysis program for buildings should have the capability of accepting material models for steel and reinforced concrete, the two most commonly used building materials. The nonlinear material model

represents, in steel structures, effects such as the spread of plasticity, residual stresses, cyclic hardening, and the Bauschinger effect, and, in concrete, cracking and stiffness degradation due to cyclic loading. Interactive programs developed at Cornell to date have typically been geared to handle steel frame buildings. It is, however, desirable to encapsulate the hysteresis model in a generic mould. This eliminates dependencies of other parts of the computer code on a specific model. A modular procedure for the material model with a clearly defined interface eases the comparative evaluation of the modelling schemes which abound in the literature. The capability of accepting different material types is vital to the realistic modelling of mixed construction type buildings that might have a concrete core surrounded by steel frames, or buildings made of the two different materials at different levels.

Material nonlinear analysis using the stiffness method involves the computation of the matrix, K_p, which is the contribution of material nonlinearity to the stiffness matrix. The beam-column elements currently in use in the Cornell programs utilize a concentrated plasticity model [HILM84] which extends the formulation in reference [PORT71] to include strain hardening. Nested surface and bounding surface models in 3-dimensional force space are currently employed for the strain-hardening behavior of structural steel. Again, this is a limited model of the true behavior of structural steel members in that plastification is considered mainly a function of the axial force and

the bending moment about the two principal axes of the member. More realistically, one might require an inelastic behavior model governed by a larger number of forces, or displacements, stresses or strains. Constitutive relationships for the micromodelling of concrete are also being proposed using the bounding surface type of concepts [YANG85]. Current research at the SUNY Buffalo as well as at Cornell University is aimed at developing macromodels for the behavior of concrete members. Hence, it is reasonable to expect that the inclusion of the capability to model concrete could follow lines similar to the one adopted for steel structures.

In addition to the modelling of the main structural members, the modelling of other components such as masonary or reinforced infill walls is required. Modelling of the soil-structure interaction is done at an elementary level by using simple linear spring models in the initial stages. Composite floors can also be modelled using special material models.

The micromodelling research can also be used for devising and calibrating material models for use in macromodelling. This route provides the opportunity of obtaining a large amount of data from numerical experiments, thereby aiding the development of macro material-models. This can alleviate the difficulty of performing physical experiments for all conceivable modelling situations.

It is desirable to identify clearly a set of input and output variables which can be expected to be used in a constitutive model. This is helpful in making modular procedures for modelling the inelastic behavior by identifying a general interface between the material model and the rest of the computer program.

2.3 Loading Schemes

Loading on the structure consists of static or quasi-static loading and dynamic loading. Static loads represent the dead load and live load on the structure and are applied at nodal points or on the members themselves. It is also possible to associate time histories (or loading path, in case of a quasi-static loading) with any of these loads, thereby making it possible to simulate sinusoidal or other special types of time-varying loading. Load histories are especially useful in simulating experimental results.

Thus, loading on the structure can in general be assigned through concentrated nodal loads, line-loads, and loads on surfaces. These loads then have to be converted into equivalent nodal loads for use in the stiffness method of analysis using some consistent method. This may require either a tributary-area type or some other method of distributing loads from the nonstructural surfaces and members to the structural members and/or to the nodes.

A library of earthquake ground motion records is available for input.

These accelerograms can be edited or scaled in magnitude or in time. Three components of ground motion are allowed to simulate earthquake excitation. The structure can also be aligned at an angle to the recorded directions of seismic motion. Generation of response spectra from the input ground motion records is possible, and these spectra can be easily plotted on the graphics device associated with the computer program.

2.4 Analysis Types

It is possible to perform analyses with different levels of nonlinear behavior. A first-cut analysis is usually a linear elastic one, while subsequent ones can include the effects of geometric as well as material nonlinearities. The analyst must have the option of considering the one or more forms of nonlinearity at will. (Similarly, choices must be available among static, quasi-static, and dynamic analysis capabilities. See subsection 2.5 below)

Computation of the geometric and material nonlinearities involves the generation of the corresponding contributions to the stiffness matrix. Geometric nonlinearities are accounted for by using the updated Lagrangian approach. The equilibrium equations are formed on the deformed structure using a closed form of the linear geometric stiffness matrix and closed form expressions for the element end forces which are nonlinear in the displacements [ORBI82][WHIT86]. These procedures are

still topics of research and need to be modularized in the computer program to allow alternative strategies.

2.5 Solution Procedures

Though primarily meant to be a dynamic analysis tool, static analyses can also be performed using this program. Further, because of the focus on fully nonlinear behavior, greater emphasis is placed on transient time-history analysis through direct integration techniques than by modal time history analyses. The solution procedures to be used for these analyses are outlined below.

2.5.1 Static Analysis

A global nonlinear solution scheme is utilized for the static analyses. This involves applying the loads incrementally and accounting for the nonlinear behavior. Either a pure-Newton or a quasi-Newton incremental solution procedure can be used for tracking the nonlinear response curve. Algorithms for capturing limit-point behavior are also to be considered. Arc-length methods are also used for problems where severe nonlinear behavior has to be tracked. A linear-elastic analysis can obviously be performed in one step.

2.5.2 Modal Analysis

An eigenvalue analysis is required for computing the modal periods of vibration and the mode shapes. Such information is useful in estimating

the time steps to be used in the transient analysis (especially for explicit analyses where the time-step size is crucial for accuracy and stability). It can also be used for studying the period elongation effects due to the inelastic behavior of the structure. For large structures, a complete eigensolution is prohibitively expensive, hence the first few and the highest frequency modes are the only ones to be computed. Modes can also be used as a basis for linear or nonlinear transient dynamics and for response spectrum analysis.

2.5.3 Transient Dynamic Analysis

The main thrust of this program is towards performing transient dynamic analyses for buildings. Direct integration techniques are used for solving the differential equations governing the response of buildings to complicated loadings such as those due to seismic excitation. These techniques provide a more realistic and complete solution in comparison with a response spectrum type of analysis, especially with buildings of irregular shape and when significant nonlinearities are involved.

Several different algorithms have been proposed for direct integration of the equations of motion. The most commonly known algorithms can be categorized into explicit and implicit algorithms. The widely used Newmark-beta or the Wilson-theta algorithms are examples of the implicit type while central difference techniques are typically explicit algorithms. Algorithms can either be single-step or multi-step methods. Again, with geometric and material nonlinearities, single-step methods

are favored particularly because they facilitate adaption of the time step size. Features typical of the various algorithms make them attractive in different situations. Though these algorithms differ widely, several computational steps are common among them. It is therefore possible to have a pool of utility routines (such as those for computing stiffness, member forces, etc.) with interfaces which allow access from several different algorithms.

A separate driver module is required for each of the analysis algorithms though they all access the same database and use the routines from a pool of structural analysis utility routines. Since data requirements of the different algorithms are slightly different, flexibility in the data storage has to be ensured. For example, memory has to be allocated for storing the global stiffness matrix in the implicit integration scheme, while this is not required in the explicit algorithm. The availability of dynamic dimensioning of arrays in the C language can facilitate the handling of data in different algorithms.

The use of multiple processors leads to a further proliferation of algorithms available for solution of the problem. Parallelism is exploited in the conventional explicit and implicit algorithms at various levels using direct as well as iterative methods. Current research in parallel processing for frame dynamics at Cornell includes the implementation of explicit as well as implicit techniques using coarse-grained parallelism. Much research needs to be done to refine

parallel algorithms and methods of subdividing workloads on processors in different processing environments.

2.6 Interactive-Adaptive / Batch Computing Environment

Nonlinear analysis of large arbitrarily shaped buildings presents various other challenges apart from those of complete modelling. Since a large amount of data has to be handled, and the analyses are affected by numerous parameters, it is extremely advantageous for the analyst to have a good interactive feedback from the analysis program as it progresses. Graphical menu-driven programs provide a user-friendly environment wherein it is possible to perform selectively various operations and obtain visually a large amount of information related to the analysis while it is being executed. The adaptive nature of these programs enables the modification of various parameters controlling the analysis algorithms as the analysis progresses through the different stages of loading or in time.

There is, however, a price that has to be paid for interactive-adaptive programs. Larger memory requirements and processing speed limitations make very large problems unsuitable for such programs unless very powerful processors and graphics display devices are available. In the absence of sophisticated computer equipment, large problems are better run in the "batch" mode. A powerful processor such as an FPS-X64 attached processor or a supercomputer can be used sequentially, or several processors utilized in parallel. The examination and

interpretation of the results of these batch runs can however, still be done in a graphical interactive environment. Hence, it is important to have a program which allows the analyst the choice of computing mode. Such considerations are important in determining the versatility of a program.

2.7 Visualization

The analysis of large buildings requires considerable pre- and post-processing capabilities. The concept of interactive-adaptive analysis, however, integrates preprocessing, processing, and postprocessing. The structural geometry, the analysis parameters, and the results can all be visualized and changed at any stage of the analysis. Such an environment provides a constant monitoring of the analysis and allows the analyst a better "feel" of the structural behavior, especially when nonlinearities are involved. The screen of the video device is usually divided into several viewports, in each of which different information can be monitored simultaneously. Alternatively, a scheme of pop-up windows may be adopted for the various displays.

2.7.1 Geometry of the Structure

The program CU-PREPF (formerly FRAME3D) has been used at Cornell for preprocessing of steel framed structures.[PESQ83] This program creates the geometry, and assigns boundary conditions, loading patterns, and

member properties. It has been limited to handling structures with beam-column type of elements only but is currently being expanded to include surface type elements too. The program is also being ported from a minicomputer with a vector display to a raster graphics workstation environment.

To model buildings more realistically, one requires a model which can accept both line and surface type elements in a fairly arbitrary configuration. This allows the representation of walls, floors, and infills in addition to the beams and columns. Further, special techniques are required for the representation of finite joint sizes, eccentricities of members, and imperfect joints (including hinges and releases). Extrusion can be used as a method of assembling large structures -- here a given configuration of members may be "extruded" through a given number of storeys to speedily generate a model with ease.

A geometric modelling system that is capable of representing objects built from wireframes and surfaces is required for the representation of the topology of a typical structure within the computer. This provides the flexibility required to represent a wide variety of elements in the model. Since the finite elements used in the program fall in the topological category of lines and surfaces, all the geometric information and attributes associated with the elements are efficiently stored in the data structures associated with the modelling system.

Schemes for representing such building frames on video devices are greatly aided by the use of color graphics. If the model of the building becomes too complicated, it may be necessary to simplify images either by depicting subassemblages or by using hidden-line or hidden-surface algorithms.

2.7.2 Analysis Parameters

Each analysis algorithm is controlled by a set of many parameters such as the time-step size, various tolerances, flags to toggle analysis options, etc. Default values and initial values are assigned to most of these parameters. Yet, it should be possible to modify interactively one or more of these parameters before or during an analysis. It is also possible to halt a running analysis and restart it with a modified set of parameters. An interactive menu-driven program is ideal for such capabilities.

2.7.3 Response of the Structure

Response quantities commonly monitored include displacements, member forces and information pertaining to the nonlinear deformations and/or damage. A variety of representation schemes like deflected-shape diagrams, graphical plots, color codes, and special symbols can be adopted for displaying this information. For example, one can associate colors with force levels in the members of the structure, special symbols to indicate the formation of hinges or audible signals with the

occurrence of any special events. Specific response quantities and force interaction diagrams selected by the analyst can be graphically displayed in one or more viewports on the video-terminal. A graphical representation reveals interesting features and trends in the responses which would otherwise be hidden in the maze of computer printouts.

SECTION 3

DATA REPRESENTATION SCHEMES

For the discussion of representation of data, the following subsections cover topics of data management in general, and specifically, topological and geometric modelling, data structures for structural analysis, and the data manipulation language to be used in a computer program.

3.1 Data Management

Engineering analysis and design programs handle vast amounts of data. This data might be generated by several different programs and must be available for use in various formats to the different components of a user-friendly, interactive analysis or design environment. Earlier programs were typically stand-alone programs and hence little consideration was made of information sharing. These programs were also difficult to transport due to dependencies on system-specific data handling techniques.

One key to writing a modular and portable computer program lies in an

efficient storage and retrieval system for data. A database should ideally be thought of as, first, allowing all information to be defined and stored and, second, providing users access to it for all processes. As stated in an earlier chapter, it is feasible to envision a completely integrated computer system for analysis, design, and construction. Such a system inevitably requires a well organized database. This is also helpful in moving towards a system where the overall process from the conceptualization stage to the final completion of a project can be interlinked. In fact, there exists a plethora of publications that recommend that all major software development efforts in engineering should incorporate an efficient and reliable data management system.

A data management system mainly differs from a typical flat file system in that it provides a data definition -- independent of any programs using the data -- which is used to manage the data. In a flat file storage, there is no separate data definition and so every application program must have its own. Any time a file's format or definition is changed, corresponding changes must be made in all the application programs that use this data. In addition to the data-definition, a data management system employs a high-level query language to allow the user access to specific data. The user need not have knowledge of the details of how that data is stored internally. It is thus possible to easily obtain information, such as "the list of all members adjacent to a given node," using queries. The applications program is therefore well isolated from the underlying data representation or schema.

It is also sometimes necessary to archive a large amount of data generated by an analysis program -- for example for playback of the results on a graphics device at a later time. A data management system can be used to support archiving.

3.2 Topological And Geometric Modelling

Another area in which improvements in data schemes are needed is the topological and geometric representation of structures. The need to construct and modify complicated models of structures calls for a program which accepts the requests of a designer and interprets them as numerical information to be stored in the computer. In other words, there exists the need for an efficient modelling system to represent the buildings to be analyzed by the program.

Many structural analysis programs for buildings have essentially relied on a wireframe representation of the structure. Fairly simple models have been represented so far, and the data associated with them has usually been stored in a format that is convenient just for the analysis program. For example, the only topological information that has traditionally been stored is the element connectivity. This is sufficient as well as efficient for operations like generating the stiffness matrices. However, an interactive analysis program might require information about, say, the number of members framing into a given node. To extract this information from the element connectivity data involves an expensive search. An interactive analysis system with

a good graphical user-interface requires considerably more topological information such as node-member adjacencies or edge(beam)-surface(plate) adjacencies, etc. Traversals of the data-structure should be efficient, i.e., involving constant or linear time searches. A database should also be flexible and easy to modify so that storage space is optimally utilized. A topological structure known as the Winged Edge structure has recently been used [WAWR87] for modelling of two-dimensional finite element meshes, and excellent program modularity has been obtained as a result.

A detailed representation of the structural form of a building requires a wire-frame model for the beams and columns and surface type elements to represent, say, the shear walls, floors, or cladding. Future extensions to this program could include detailed study of soil-structure interaction, where it may be necessary to include solid elements in the model. One of the popular methods of solid modelling used by mechanical engineers is constructive solid geometry where complicated objects are built using boolean operations on a set of primitive objects. This is however not suitable for structural modelling since it does not support non-manifold geometries (such as zero-thickness objects or an edge common to more than two surfaces).

Kevin Weiler [WEIL86] has recently proposed a new system for modelling of non-manifold geometries. This representation scheme supports the

unified and simultaneous representation of wireframe, surface, and solid modelling forms. A data structure known as the Radial Edge structure is presented to provide access to all topological adjacencies. This or a similar data structure can be used for representing the topological information in such a form that makes it efficient to make queries regarding the adjacencies of elements. Also, the graphics interface which requires quick traversal of the structure and retrieval of information benefits from it. Apart from the topological information, all other structural information such as the stiffness matrices or displacements associated with the elements and nodes of the building structure, can be stored as attributes of the elements of this topological data structure.

The use of a detailed data structure does, however, have the added overhead of storage required for the representation of topology. Extra effort in coding the database routines is also significant. In fact a completely general topological structure for non-manifold geometry would be fully utilized in a general finite-element program consisting of several finite element types and where the mesh needs frequent updating through the addition, deletion or modification of elements. However, even in a structural analysis program for buildings, it provides excellent support to program development, modularity, portability, and extendibility.

3.3 Data Structures For Structural Analysis

A structural analysis program mainly deals with data that is associated with either the finite elements in the structure, the nodal points, or with the degrees of freedom. The topological data structure (TDS) can form the basis of the database. This TDS, in addition to providing adjacency information, is extendible to store almost all other structural information. It should be noted that though there may not be a one-to-one correspondence between the finite elements and the elements of the TDS, all finite elements are represented at some level in this topological database. (A topological element can be a vertex, an edge, a loop, a surface, a region, etc.) The structural analysis data are mostly stored as attributes of the elements of the TDS. The adjacency data are stored in records or tuples and so the additional analysis data are stored in additional fields of the same records. Newer languages such as C offer much more advanced structures which can be manipulated using pointers. These C structures allow greater flexibility in grouping different data types together.

Given below is a listing of the major categories of data associated with a structural analysis program for buildings. This list is meant to serve as an example and is expansible.

A. Problem Parameters

1. Title and description
2. Number of nodes and elements

3. Current state of structure in time

B. Element Data (For each element):

1. Element type and geometry and material groups.

2. Orientation vectors, initial and current. (example: web vector of beam-column)

3. Element stiffness matrices -- elastic, and geometric and material nonlinear components. (If needed permanently)

4. Rotation transformation matrices for undeformed, current and updated states.

5. Member forces or stresses: total and incremental.

6. Damage information, ductility etc. (Example: Ductility at the two ends of a beam-column or at the Gauss points of the element)

7. Nonlinear model data: location of hinges or damage, memory parameters of inelastic material model, offsets of yield surfaces, plastic deformation matrix, and other state variables.

8. Connection types at ends or along the edges of members.

9. Graphical information: viewing transformations, display status, color code, other graphical attributes.

C. Nodal Data (For each node) :

1. Coordinates: current and initial (Geometry)

2. Associated degrees of freedom

3. Floor level identification

4. Masses

5. Displacements, velocities, and accelerations at the current previous or next step.

6. Incremental displacements

7. Graphical information: viewing transformations, display status, color code, other attributes.

D. Finite element Information (For each element type) :

1. Number of nodes

2. Number of degrees of freedom

E. Geometric Properties (For each element group)

1. Cross sectional area

2. Moments of inertia

3. Mass density

4. Depth, thickness, etc.

F. Material Properties (For each material group) :

1. Elastic properties

2. Nonlinear properties and calibration parameters for the material model

3. State variables of the nonlinear model

G. Loading Information

(i) Nodal and distributed loads:

1. Locations and magnitudes of loads

2. Category of loading -- static or dynamic

3. Load histories associated with the static/dynamic loads

(ii) Earthquake loads:

1. Accelerograms in X, Y, and Z directions

2. Scaling factors

3. Components of acceleration currently included in analysis

H. Matrices associated with solution algorithms

1. Vectors for displacements, velocities, and accelerations

2. Load vectors at a time step, basic and effective
3. Global stiffness matrix
4. Other temporary working space

It must be noted that all of this data may not be permanently stored. For example, in one case, it might be of interest to store all the computed response parameters, while in another, just enough information should be stored to restart the analysis.

3.4 Data Manipulation Language

The topological data structure which is used for modelling the building is created using a set of high-level operators relevant to that data structure. This set of manipulation operators has a hierarchy within itself. The lowest level operators manipulate the primitives of the data-structure while higher level operators invoke the lower level operators and manipulate more complex entities. Modifications to the topology also involve similar operators.

Once the topology has been defined, access operators are used for extracting information from the database. Access operators consist of queries and traversals. In the topological context, a query is a single access to adjacency relationship information to determine a single element of the adjacent group of a specified element. Traversals are repeated queries to determine all members in an adjacent group. [WEIL86] Since almost all data relating to the structural analysis are

stored in the form of attributes of topological elements, they can also be accessed through queries and traversals.

The physical storage of the data must be transparent to the program. If all the data access is through a set of intermediate query routines, should the storage scheme change, the only routines which have to be modified are the query routines. Thus the data can either be in the main memory of the computer or on a disk. Depending on the problem at hand, one or the other scheme can be chosen without affecting the analysis code. The set of manipulation and query operators constitutes the data manipulation language. Query routines are used for depositing as well as extracting data from the database. The initially empty database is first filled with the problem description or primary data. The secondary data (stiffness matrices and such) are generated by the program and again stored in the database. The results of the analysis program are finally stored when the program is run.

SECTION 4

SOFTWARE DEVELOPMENT

Development of large programs such as this one calls for adherence to the principles of software engineering. This is necessary to ensure that the program has a useful life of at least several years. One major aspect of the program design, the data management, has already been discussed in Chapter 3. Some of the other factors to be considered include the choice of the programming language, the program organization, and the user interface.

4.1 Programming Language

Most of the structural analysis code written to date uses the FORTRAN language. The familiarity of most engineers with the language and the efficient executable images that it produces have primarily been responsible for its popularity in the engineering world. There are, however, various other considerations which must now be made in selecting a programming language.

Though many other programming languages are available, the language C

seems to be gaining popularity in various computing environments. Major programming problems can arise when FORTRAN programs have to interface with programs written in other languages (such as an expert system written in LISP, a graphics package written in C, or a menu-manager written in C++). Interfacing is relatively an easier task if the C language, which is a lower level language, is used. (The term lower level does not imply a belittling connotation. It only means that the language allows operations even at the level of bits and bytes in addition to the usual high level constructs.)

Transportability is another major concern with FORTRAN programs. Many hardware manufacturers provide their own set of extensions to make FORTRAN more attractive. Programmers are often lured into using these special features while developing large and complicated programs as they provide significant conveniences. Transporting such code to a new computing environment is usually a difficult task. System-specific utilities such as those for dynamic memory allocation must also be isolated so that similar utilities can be substituted for them in another computing environment.

The efficiency of the executable files has been a major strength of FORTRAN. A recent publication [FAZI87] has compared the performance of programs written in C and FORTRAN for structural analysis purposes. It has been pointed out that C programs though larger in size (source code) produce smaller and faster executable images on the UNIX and MS-DOS

systems. (FORTRAN still performs better on systems such as the VAX/VMS which are well equipped for the language.)

The C language offers almost all the features of FORTRAN and in addition many more. It offers better control constructs, structured data types, and allows greater flexibility to the programmer. It is also possible to interface programs written in C with FORTRAN or PASCAL code, a fact which can be helpful in making the transition to C. Knowing FORTRAN, one may fairly easily learn at least the subset of C which is equivalent to FORTRAN. The knowledge of the rest of the features of the language would follow naturally. Though not a new language, C is gaining acceptance within the engineering and scientific computing community and must be considered a viable alternative to FORTRAN.

4.2 Layered Software For Interactive Programs

The use of computer graphics and menu-driven programs allows the kind of user-interface necessary for interactive-adaptive analysis. Writing such programs, however, involves considerably greater effort than that involved in the usual "card-input-printed-output" style of programs. Many programs developed at Cornell using computer graphics provide such interaction. They, however, have often been designed to run on the unique and state-of-the-art systems in use at Cornell. This is due to their use of specific software packages, especially for graphics, which in turn are written for specific hardware configurations. Since calls to these packages are interspersed through out the code, portability has

been restricted. These programs have established the style to be emulated, although the programming approach must be modified to avoid their pitfalls.

The proposed program is developed in a layered fashion. Layering involves writing routines at different levels. At the core of the program are usually the routines handling the database. Higher level routines invoke lower level routines for accessing data, performing analyses, for controlling a graphics device, or for user-input. The insulation of high level routines from the specifics of operation of the lower level routines promotes modularity.

4.2.1 Menu interaction

Menu driven programs involve an event-driven style of programming. This means that the flow of control within the computer program is not completely pre-determined. The sequence of events is partially controlled by the input provided by the user, and hence, there is relatively greater flexibility in performing various segments of the analysis. The user is allowed to select actions by pointing to available options on a video terminal using a pointing-device such as a pen and tablet or a mouse. The program then executes the specific routines which are required to comply with the request of the user. Coding such a program can be facilitated by the use of software packages known as menu-managers [DYKE87]. These packages take away the burden of planning and coding menus and allow a variety of functionalities using

buttons, and other input schemes on a graphics device. These packages are themselves written in a layered fashion and hence can be adapted for different hardware configurations.

4.2.2 Graphics Software

A graphics software package is used for the display of the three dimensional structure, various graphical plots and diagrams and menus on a graphics screen. Though it is not yet clear what package is to be used, it is assumed that several high level operations such as viewing transformations, clipping, hidden-line and hidden-surface algorithms, etc. are available for use. In fact, some of these capabilities are now provided by workstation manufacturers in hardware. Modern workstations provide a high quality color image using raster displays and often provide their own graphics packages too. It is therefore possible to use either the workstation graphics package or another commercial package.

4.3 Programming Procedures

It is good practice to adhere to certain rules and guidelines in writing computer programs. This not only enhances the readability and facilitates maintenance of the programs but is also helpful during the development stages of the program. The major programming procedures followed in program development are outlined below.

1. Providing standard header blocks in all subroutines giving detailed information about input, output, and local variables; the purpose of

the routine; and the procedures followed in the code.

2. Establishing conventions for naming variables and routines logically.

3. Using named variables in place of numerical constants.

4. Flexibility in the choice of measurement units.

5. Using defensive programming with error checking mechanisms built into the program.

6. Extensive self-documentation within the code.

The program under development at the Program of Computer Graphics at Cornell reflects a set of consistent programming standards which are currently being established. These include standardized header blocks, conventions for modular structure, naming procedures, and an emphasis on portability.

SECTION 5

HARDWARE REQUIREMENTS AND GRAPHICS TOOLS

The extremely dynamic nature of the computer hardware industry today has made planning for the future a difficult task. Tremendous advances have been made in the last few years and the processing power commonly available now in even small computers is significant. A brief discussion of current and future hardware components follows.

5.1 Current Configuration at Cornell

Most of the software developed at Cornell for structural analysis and design currently runs on the VAX/8300 and VAX/8700 mini-computers. The analysis of large problems which require a large amount of computing is done using programs written to run on the FPS 164-MAX attached processor with a VAX/750 as the host. The graphics device used is the Evans and Sutherland PS2/MPS vector refresh display. Mass storage is available on several Digital RA-81 (456 Mb) disks and a Digital TA-81 (1600/6250 bpi) tape drive. An experimental parallel processing program for structural dynamics operates on micro-VAX workstations connected via Ethernet and using the ELN operating system.

A discussion of the performance of this equipment and that of a desirable system follows.

5.2 Graphics Devices

The first step in the analysis of buildings is the preparation of the input data for the main program. A graphical frame analysis pre-processor (FRAME3D) is currently in use at Cornell and it employs the E & S picture systems. A new version of the same program, CU-PREPF, is now being developed in a portable form primarily for use on raster type devices such as the VAX-GPX workstations.

The maximum number of points (and lines) which can be drawn on a vector picture system places a limit on the size of buildings which can be assembled with this program. Further, if the analyst wishes to see the displaced shape of the structure along with the original shape while the analysis is in progress, the number of lines that have to be drawn on the screen increases considerably. With the overhead of the lines which must be drawn on the screen for menus and other information, the E & S picture system currently allows buildings with approximately 1500-2000 members to be displayed. This corresponds to a rectangular frame building approximately fifty storeys high with two bays in each direction. A more realistic building with, say, five bays in each direction could only be represented to a height of 12 to 13 storeys.

It is obvious that for representation of large realistic buildings, more

powerful devices are necessary. New high-resolution raster devices coupled with workstations offer almost an order of magnitude larger number of lines which can be dynamically moved on the screen. These devices also have the advantage of having color displays. In addition, they allow polygons to be drawn along with lines. It is certain that such devices will be available soon. In view of the fact that new graphics devices allow more information to be displayed on the screen than can be discerned by the human eye, it also becomes important to develop software which allows selective viewing.

5.3 Memory Requirements

The computer program, DNA3D [HILM84], currently in use at Cornell has been used to estimate the core memory requirements for nonlinear dynamic frame analysis purposes. The estimates in Table 5-I have been made of the storage required on a per element and per node basis and on the whole for an implicit dynamic analysis of a simple rectangular framed structure.

The figures in Table 5-I indicate that the nonlinear dynamic analysis of large buildings requires more core memory than is generally available on current workstations and must be performed either on a mini-computer with virtual memory or on a large main-frame computer. New workstations, however, will have both greater core memory and virtual memory capabilities and are therefore likely to be able to handle medium sized or large buildings on their own.

TABLE 5-I CORE MEMORY REQUIREMENTS

Description of the Building	# of members	# of nodes	Core Memory (M bytes)			
			Nodal info	Element info	Other	Total
6 bays x 6 bays x 20 storeys	1920	756	0.53	4.86	15.61	21.0
6 bays x 10 bays x 8 storeys	1448	594	0.42	3.66	16.93	21.0

A proposed new program can be expected to have higher memory requirements than those of the current program. The coding of its extra features such as multiple element types and material types requires greater memory. Also, it stores considerably more topological information in the database which adds to the memory requirement. The analysis of large buildings consequently must run on larger machines or on several processors using parallel processing. It is, in fact, feasible that the core-memory requirements would lie in the range of 100 Mbytes for a realistic simulation of tall building behavior.

In addition, it may be desirable to store all the results of a time-history analysis. Then it becomes possible to obtain the complete record of the analysis for detailed scrutiny after completion of the analysis. This requires a considerable amount of disk storage space. The program DNA3D can generate, for a medium sized building and for about 150 stored response steps, close to a hundred thousand blocks (approximately 50 Mbytes) of data. Hence, a sizeable amount of disk storage, e.g., several hundred Mbytes, is also necessary to keep a record of these nonlinear analyses.

5.4 Processing Speeds

Efficiency is another major consideration in selecting the hardware for programs such as this one. The rather limited speed of processing (of the order of 0.1 to 1 MFlops) available on workstations and mini-computers severely limits the size of structures which can be

analyzed in a reasonable amount of time. In fact, a nonlinear time-history analysis for a steel-frame structure involving approximately 3000 unknowns takes about 3-1/2 hours of processing time on an FPS-164 attached processor (with, say, an effective rate of 10 M Flops) to generate 150 steps of response with the program DNA3D. Higher processing speeds can be achieved using a supercomputer with sequential and/or parallel processing. Both sequential and parallel architectures are discussed in a later section.

Interactive-adaptive analysis requires that enough processing power be available to generate results even for a multi-storey structure in a reasonably short time to allow the user to control the program with ease. Such a capability is only available currently at Cornell for small (of the order of a few hundred DOF) structures. Workstations on the market now are rated in the range of 1-10 Mips. Much higher processing speeds will be available on the supercomputers.

5.5 Sequential Architecture

Most conventional computers can be classified as sequential machines. The processing speeds on these machines has however, dramatically increased over the past few years and is currently approaching the level where further increases are limited by the effective speed of signal transmission through a chip. These sequential supercomputers are rated at speeds as high as 200 MFlops per processor (compared with, say, 1 MFlop at best on some of the new workstations). A fairly large core

memory as well as disk storage is usually supplied by manufacturers. As these machines generate large amounts of data which must be constantly interpreted, a high-bandwidth link with a graphical workstation can allow results of the simulations to be monitored either as they are being computed or with a small time delay.

Applications without much potential for parallel processing must rely on such powerful computers. Vectorization is commonly used for increasing efficiency of computations on these machines. Conventional computer programs have to be evaluated to determine the most computationally expensive segments and these are executed using special utilities provided in the software libraries. Such computers are, however, expensive and scarce at this time. Vectorized accelerator boards plugged into workstations are another possibility for the future.

5.6 Parallel Architecture

Current research in parallel processing for nonlinear structural dynamics at Cornell focuses primarily on coarse-grained parallelism. The parallelism is essentially obtained by substructuring and having a processor assigned to each substructure. The particular system temporarily in use at Cornell consists of several micro-VAX workstations connected through a network (Ethernet). This is a bus-type configuration where all the processors can communicate with each other. ELN, the operating system in use for this configuration permits communication between, and techniques for synchronizing the various

processors. A distributed memory system is used in that no global shared memory is available to the processors.

The restrictions of such a system stem from the efficiency of the communication system. The particular configuration in use in the afforementioned research utilizes Ethernet, a network which also serves other machines. Hence the communications for the parallel hook-up are adversely affected at times of high traffic on the net. Since the computations must proceed in lock-step for a substructuring type of nonlinear analysis, poor communications adversely affect the speed-up ratios in the parallel environment. It is obvious that this system does not measure up to the standards for a practical parallel configuration. It, however, allows useful research on algorithms for future hardware environments with more efficient bus-type communication.

The memory of each processor is again a critical factor to be considered. If speed-up factors are a prime consideration, and parallelism is to be obtained by assigning a portion of the structure to each processor, it is desirable to have considerably large substructures assigned to each processor. This maximizes the computations within a processor and minimizes the inter-processor communication [HAJJ87], a situation ideal for parallelism.

Several architectures are available for coarse grained parallelism [Appendix A of [HAJJ87]]. The bus architecture seems well suited to the

problem of structural dynamics for buildings with arbitrary geometry if the capacity of the bus is not a limiting factor for the efficiency of the algorithms. Another versatile architecture is the hypercube. A large number of architectures can be mapped onto this one and it provides efficient communication paths between its nodes (processors). The choice of an ideal parallel architecture for transient nonlinear dynamics cannot be unanimously made given the evolving state-of-the-art. Hence, an attempt is made in creating a database that it is flexible enough to allow implementation on diverse architectures.

SECTION 6

MODULAR ORGANIZATION OF THE PROGRAM

This chapter describes the major modules to be developed in the program and discusses the overall flow of control. As mentioned earlier, the program is developed in a layered fashion. The main components of the program are:

(i) Database

(ii) Data definition and manipulation language

(iii) Structural analysis utility modules

(iv) Algorithm drivers

(v) User interface

The database is described in Chapter 3. The data definition language consists of routines which create the topological elements and store all adjacency information and attributes of the topological as well as finite elements. The data manipulation language consists of routines which are the means of accessing the database. The structural analysis utility modules are functional units which perform actions that may be common to several analysis algorithms, while the algorithm drivers contain the step-wise flow of control for the various algorithms.

The User-interface consists of the menu drivers and, at a lower level, graphics display routines.

Components (iii) and (iv) above encompass the areas in which much of the structural engineering research is required, whereas the other three components can be considered parts of the underlying support structure for this research.

Figure 6.1 shows the relationship between the different modules of the program. The kernel of the program consists of the database which can be accessed through routines written for that purpose. The algorithm drivers call upon the structural analysis utility modules, though both can access the database through the query/modification routines. The user interface allows the analyst to select algorithms of choice and to obtain information visually.

Figure 6.2 shows a high-level representation of the flow of control in this program. This sort of flow can be expected for interactive analysis using any of the algorithms to be implemented. Specific details of the flow within each algorithm vary.

6.1 Data Definition and Manipulation Language

Two kinds of routines are available in this category. The definition routines are used for storing or modifying data in the database, while the query routines are used for retrieving data from the database.

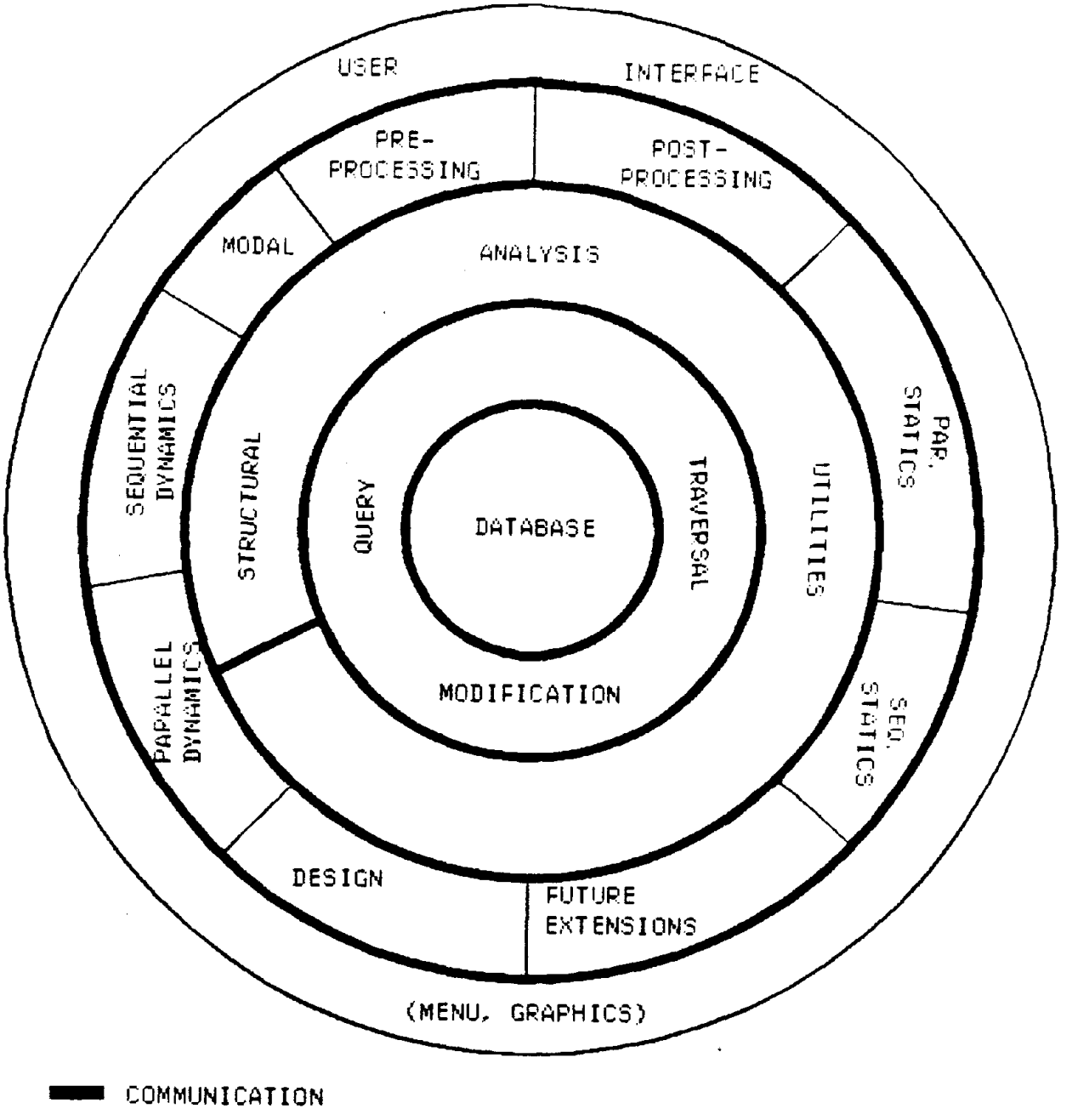

FIGURE 6-1 RELATIONSHIPS AND COMMUNICATION BETWEEN THE COMPONENTS OF THE COMPUTER PROGRAMS

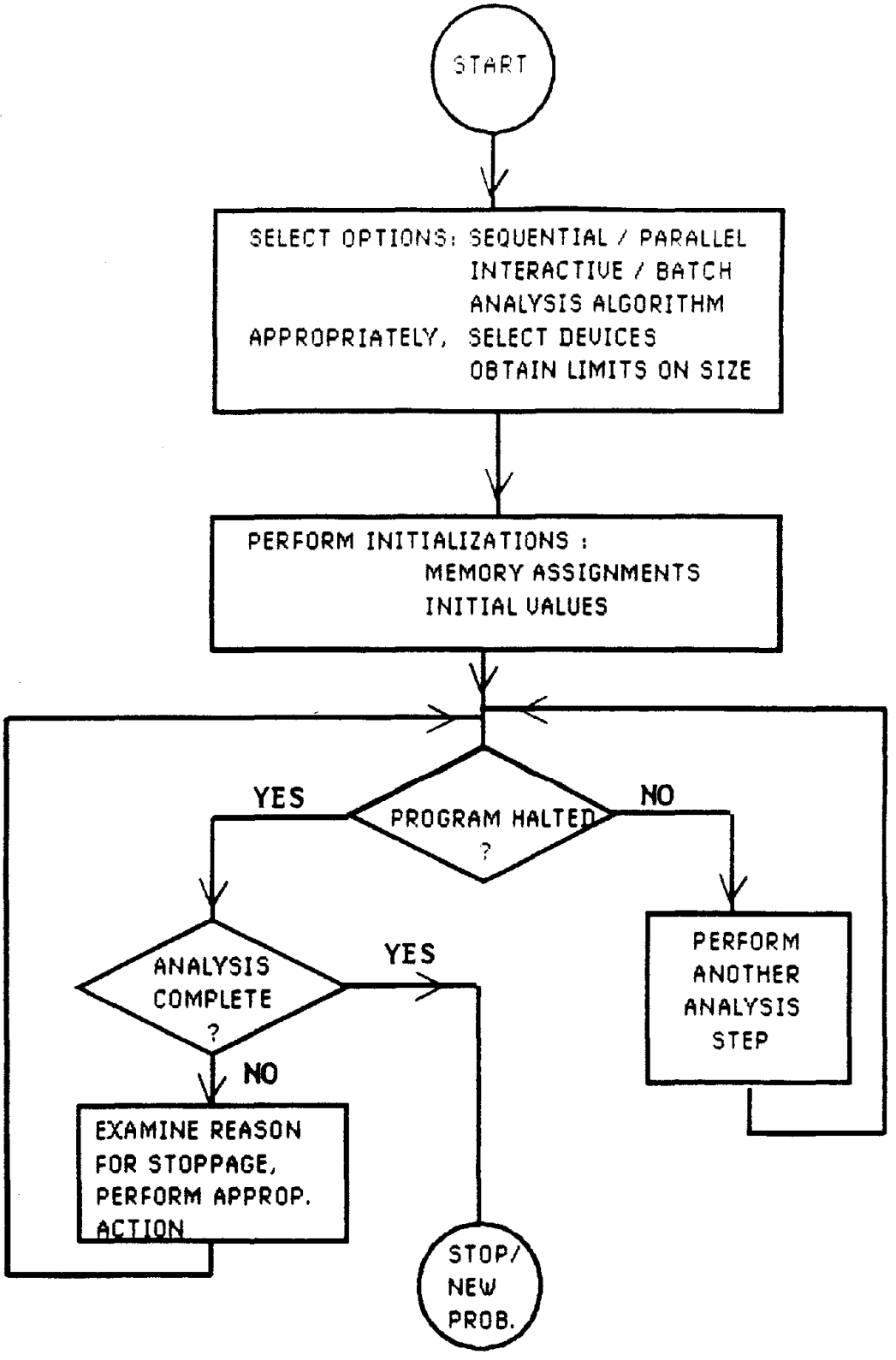

FIGURE 6-2 HIGH LEVEL FLOW OF CONTROL

Typical major routines in these two categories are:

Modification Routines:
- Create (or Delete) Element and Associated Record
- Initialize or Modify Attribute Fields

Query Routines:
- Retrieve Topologically Adjacent Element
- Retrieve an Attribute of an Element
- Traverse a Structure

(Note: Elements referred to here are topological elements.)

6.2 Structural Analysis Utility Modules

Routines or procedures commonly used in the nonlinear dynamic analysis of structures are grouped in this category. Such routines are for the generation of the nonlinear stiffness matrices, computation of loads, equation solvers, force recovery procedures, etc. The following is a tentative, partial list of routines in this category (More detailed outlines of a few typical routines appears in the Appendix) :

A. Rotational transformation routines

B. Stiffness computation routines:

- Global stiffness
- Local stiffness
  - Stiffness-elastic, Stiffness-geometric, Stiffness-material
    - Stiffness-elastic(for)element-type, etc.
- Global stiffness assembly

C. Loading routines:

- Load definition
- Load vector generation

D. Force recovery routines

E. Dynamic analysis utilities:

- Eigenvalue solvers
- Transient analysis
  - Explicit integration : Solution of equilibrium and central difference equations
  - Implicit analysis : Computation of effective stiffness, loads

F. Nonlinear analysis utilities:

- Evaluate unbalanced forces
- Update of geometry
- Finding step-size
- Drift control
- Ductility computation
- Damage computation

G. Equation solvers:

- Linear equations
  - Direct methods (e.g., Cholesky)
  - Iterative methods (e.g., Preconditioned Conjugate Gradient)
- Nonlinear equations
  - Newton methods
  - Quasi-Newton methods
  - Secant methods
  - Arc-length methods

H. Information routines:

- Structure geometry
- Parameters
- Response

6.3 Algorithm Drivers

These routines contain the step-wise procedure to be followed for the analysis algorithm chosen. Each of these typically have a segment for initializations and then a loop over the number of time or load steps to compute the response by some incremental scheme. These drivers also keep track of timing information and other statistics. Parallel algorithms also require routines for dividing the work load among the

processors and for other pre-processing. Further, the drivers for the different processors in a parallel environment must be identical, thereby permitting a general program independent of the number of processors involved. Thus, the following drivers are needed in the program :

Sequential analysis :
- Static analysis
- Modal analysis
- Transient dynamic analysis
  - Explicit analysis
  - Implicit analysis
  - Other mixed method

Parallel analysis :
- Division of structure
- Static analysis
- Modal analysis
- Dynamic analysis
  - Central difference
  - Domain decomposition
  - Others

6.4 User Interface

User interaction has often been ignored in developing structural analysis software. This has led to major problems due to incorrect data input, difficult control mechanisms, and almost incomprehensible result outputs. Large programs for the nonlinear analysis of structures require a user-friendly front-end and monitoring program for efficient operation.

Managing user interaction in large programs requires a significant amount of coding effort, especially when a graphical interface is

required. The portion of the program written for the interaction can, in fact, be much larger than that for the actual application. To ease this situation, packages are being developed which provide a "toolkit" for managing menu driven programs. These packages provide a menu description language, allow the design of the layout and display of menus, and handle the communication between the application program and the graphics device [DYKE87]. It is then possible to develop the menu with a high level description using the concepts of windows, keywords, toggle switches, options, prompts, numeric keypads, and messages. The application program, thus, may consist of specific modules which are called on request, leaving the high-level control of flow to the menu-manager. This is also referred to as "event-driven programming". The setting of variables is done by parsing the message sent from the menu-manager to the client program.

Several menu pages are required for handling the large number of functions in the program. Some menu systems are designed to follow a strict hierarchy of pages making it difficult to go from one page to another without climbing up the hierarchical tree. It is, however, more desirable to have a menu structure which allows free movement between pages. A set of menu buttons is permanently kept on the screen to facilitate easy access not only to other major menus but also some frequently used functions such as view specification, exit from the program, or printing a hard-copy.

SECTION 7

SUMMARY AND IMPLEMENTATION PATH

Finally, modelling realistic seismic response of buildings requires research in several areas that are summarized below. A tool to facilitate this research endeavor is being proposed, and it is hoped that this will permit greater cooperation among research groups.

7.1 Summary

The use of 3-D fully nonlinear dynamic analysis techniques for large buildings is currently not widely prevalent. This can be attributed to the need for further research to obtain reasonably accurate and efficient tools for the macro analysis of such large systems. Research is needed to develop and re-evaluate strategies of performing such analyses, especially in light of the computing environments which are becoming available now and will be in the future. These include workstations for, say, design office use or supercomputers for analyzing large systems. Advances are required in the following major areas:

1. Realistic modeling of the building with finite elements.
2. Developing refined models representing the behavior of materials in a macromodel.

3. Improving solution algorithms and implementing them for use in today's computing environments.

4. Devising efficient software tools to aid research in the areas mentioned above.

The first topic deals with the consideration of structural elements that are often neglected or improperly modeled in the analysis of buildings. The selection of appropriate finite elements to model these components and a study of the interaction between these is required. Further, it must be kept in mind that to keep the problem size within bounds, it is necessary to model buildings with relatively few finite elements, say, one or a few per beam, column, or infill panel -- in other words, a macro-analysis is required.

A form of the material model which is convenient to use in a macro-analysis is one where the material state is a function of relatively few force or stress parameters. For example, plastification and hardening in steel members is commonly modeled as a function of the axial force and bending moments about the two principal axes of sections. Better models would include the effects of the interaction with torsion and shear in nonlinear analyses. Measures of material damage are another area of interest in both steel and concrete modeling. Developments in these areas can be aided by research in micro-modeling which provides means of studying a large variety of "experimental" results.

Algorithmic research is required for both static and dynamic analyses in sequential and parallel computing environments. Given the dynamic nature of the computer industry, maximal utilization of the computing power that is likely to be available in the future is only possible with improved algorithms. Schemes for marching through the transient time-history using explicit as well as implicit techniques are of interest. For static analyses, stable and robust limit point algorithms are required.

A computer program embodying the principles of efficient data management and software engineering is being proposed. It is meant to be a modular program with a well organized data management scheme, structured programming techniques, and a friendly user interface. Graphical menu-driven programs are especially useful for performing interactive analyses. The modularity of the program permits expansion without modifying the code extensively and makes it easy to transport to a new computing environment. This permits sharing of software resources among different research groups.

Finally, the overall objective must be restated. The main objective of developing the proposed program is to obtain a tool for realistic studies of fairly large buildings subjected to dynamic excitation. This tool would be used for evaluating the various techniques of modeling structures and materials and the solution algorithms for static or dynamic analysis in a user-friendly environment. It allows for

expansion and modification as per the needs of the researchers using it, that is, it is sufficiently flexible to include various interchangeable element types, material models, analysis types, and solution algorithms.

7.2 Conclusions

A study of the state-of-the-art in dynamic nonlinear analysis of buildings and computer hardware indicates that conditions now exist where it is feasible to plan for such analyses for large systems. The availability of computing power continues to rise and it is important now to develop tools for harnessing the evolving technologies. In fact, computing speeds, core memory, and graphics performance soon to be available in stand-alone workstations makes nonlinear analysis affordable. As advances are made in modeling techniques and analysis algorithms, the need for a numerical testbed is enhanced. This testbed provides a research tool which allows the implementation and study of several analysis techniques. An interactive computing scenario is envisioned which makes easier studying the "art of nonlinear analysis" for buildings.

7.3 Stepwise Task Enumeration

The proposed computer program provides a testbed for research in the various areas that have been mentioned and is developed in a stepwise fashion. The initial focus is on the development of a data management scheme that is deemed appropriate for the problem. The development of the user interface, which is a menu driven and graphical one receives

attention next. This requires significant time and effort in the design stage. The subsequent implementation is likely to be easier due to the availability of software tools such as the menu-manager.

Existing analysis techniques are implemented next in a modular fashion. A modular program allows a "thin-slice" development approach. The components such as graphical input/output functions required for preprocessing and postprocessing, the element library, material models, and the analysis algorithms are implemented in several stages through a collaborative effort.

Concurrently with the development of the program, new ideas can be explored and tested using the program. The interaction of finite elements used for modeling the building requires significant investigation. The associated micromodelling research should also provide valuable input towards refining or creating material models for use in the macromodelling research. Corroboration of these models with experimental research is also sought. Algorithmic research for sequential as well as parallel environments is desirable, though greater challenges are posed in the area of parallel algorithms and so greater emphasis is likely to be placed in that area.

SECTION 8

REFERENCES

[CAST85] Castaner, J. L., "An Instructional System for Nonlinear Analysis and Design of Steel Frames with Interactive Computer Graphics," M.S. Thesis, Department of Structural Engineering, Cornell University, Jan., 1985.

[DOVE80] Dovey, H. H., and Wilson, E. L., and Habibullah, A., "TABS-80: Three Dimensional Analysis of Building Systems," A Report to the U. S. Army Engineer Waterways Experiment Station, Vicksburg, MS, Computers/Structures International, Berkeley, California, June, 1980.

[DYKE87] Dykes, G. W., "A Menu Manager," Internal Report, Program of Computer Graphics, Cornell University, June, 1987.

[FAZI87] Fazio, P., AND Gowri, K., "Structural Analysis Software and the C Programming Language," Computers and Structures, Vol. 25, No. 3, 1987, pp. 463-465.

[GUEN77] Guendelman, Rafael-Israel, and Powell, G. H., "DRAIN-TABS: A Computer Program for Inelastic Earthquake Response of Three-dimensional Buildings," Report NO. UCB/EERC-77/08 College of Engineering, University of California, Berkeley, California, Mar., 1977.

[KANN73] Kanaan, A. E., and Powell, G. H., "DRAIN-2D: Inelastic Dynamic Response of Plane Structures," Department of Civil Engineering, University of California, Berkeley, California, Sept., 1973.

[HAJJ87] Hajjar, J. F., "Parallel Processing of Nonlinear Structural Dynamics for Earthquake Analysis of Three-dimensional Steel Framed Structures," Ph.D. Thesis, Department of Structural Engineering and Program of Computer Graphics, Cornell University, Jan., 1988.

[HILM84] Hilmy, S. I., "Adaptive Nonlinear Dynamic Analysis of Three-Dimensional Framed Structures with Interactive Computer Graphics,"

Department of Structural Engineering Report Number 84-8, Cornell University, June, 1984.

[JENK66] Jenkins, W. M., and Harrison, T., "Analysis of Tall Buildings with Shear Walls under Bending and Torsion," Symposium on Tall Buildings with Particular Reference to Shear Wall Structures, University of Southampton, Apr., 1966, Oxford, Permagon Press, 1967, pp. 413-444.

[MACL76] MacLeod, I. A., "General Frame Element for Shear Wall Analysis," Proceedings of the Institution of Civil Engineers, 61, Part 2, Dec., 1976, pp. 785-790.

[ORBI82] Orbison, J. G., "Nonlinear Steel Frames," Department of Structural Engineering Report Number 82-6, Cornell University, March, 1982.

[PESQ83] Pesquera, C. I., McGuire, W., and Abel, J. F., "Interactive Graphical Preprocessing of Three-Dimensional Framed Structures," Computers and Structures, Vol. 17, No. 1, 1983, pp. 1-12.

[PESQ84] Pesquera, C. I., "Integrated Analysis and Design of Steel Frames with Interactive Computer Graphics," Ph. D. Thesis, Department of Structural Engineering, Cornell University, Jan., 1984.

[PORT71] Porter, Frank L., and Powell, Graham H.,"Static and Dynamic Analysis of Inelastic Frame Structures," Earthquake Engineering Research Center, College of Engineering, University of California, Berkeley, June 1971.

[STAF69] Stafford Smith, B., and Carter, C., "A Method of Analysis for Infilled Frames," Proceedings of the Institution of Civil Engineers, 44, 1969, pp. 31-48.

[STAF81] Stafford Smith, B., Girgis, A. M., and Abate, A., "Analogous Frames for the Analysis of Tall Shear Wall Structures," Canadian Journal of Civil Engineering, Vol. 8, 1981, pp. 395-406.

[SUTH86] Sutharshana, S., "Earthquake Analysis and Design of Steel Frames with Interactive Computer Graphics," Ph. D. Thesis, Department of Structural Engineering, Cornell University, Jan., 1986.

[WAWR87] Wawrzynek, P. A., "Interactive Finite Element Analysis of Fracture Processes: An Integrated Approach," Department of Structural Engineering Report Number 87-1, Cornell University, Feb., 1987.

[WEIL86] Weiler, K., "Topological Structures for Geometric Modeling," Ph.D. Thesis, Computer Systems and Engineering, Rensselaer Polytechnic Institute, Troy, NY, August, 1986.

[WHIT86] White, D. W., "Material and Geometric Nonlinear Analysis of Local Planar Behavior in Steel Frames Using Interactive Computer Graphics," Department of Structural Engineering Report Number 86-4, Cornell University, June, 1986.

[WHIT88] White, D. W., "Analysis of Monotonic and Cyclic Stability of Steel Frame Subassemblages, Vol. 1 and 2," Ph. D. Dissertation, Cornell University, Ithaca, New York, Jan., 1988.

[WILS75] Wilson, E. L., Hollings, J. P., and Dovey, H. H., "ETABS - Three Dimensional Analysis of Building Systems (Extended Version)," Report No. EERC 75-13, College of Engineering, University of California, Berkeley, California, April, 1975.

[YANG85] Yang, B-L., Dafalias, Y. F., and Herrmann, L. R., "A Bounding Surface Plasticity Model For Concrete," Journal of Engineering Mechanics, ASCE, Vol. 111, No. 3, 1985.

APPENDIX A

EXAMPLES OF MODULE OUTLINES

This appendix presents a few examples of the structure of some modules in the categories Algorithm drivers and Structural analysis utilities, referred to in Section 6. The intent here is to demonstrate a structured approach towards developing the routines, clearly identifying the input-output parameters and promoting isolation from the rest of the program. A high level of modularity is sought by keeping high level routines fairly general, i.e., independent of material models or element types. Evidently these descriptions are bare skeletons of a few routines in the applications program. Other packages of routines are needed for data management and the user interface.

MODULE NAME : EXPLICIT ANALYSIS DRIVER

CLASSIFICATION : ANALYSIS DRIVER

DESCRIPTION : This module controls the flow of the explicit analysis algorithm.

CALLED FROM : CONTROL routines

CALLS : Several Utility routines

INPUT :

PROCEDURE :

1. Initialize the variables and arrays used in the analysis.

2. While the time steps are not exhausted :

- Check for user input. (To halt the program etc.)

- If the analysis is interrupted, execute the request of the user; then continue the analysis.
  Possible user requests: Examine results
  Change analysis parameters
  Change graphics monitors
  Change structure
  (Call module INFO)

- Loop over all elements to :

  - Compute stiffness matrices, K_e, K_g, K_i.
    (This operation may not be performed at every step and the stiffness matrices may only be updated at certain steps)
  - Compute the load vector.

- Explicit integration. (Computing the acceleration using the equilibrium equation, and the displacement and velocity, using the central difference equations) This could be done on a dof basis.

- Update the geometry, i.e., nodal coordinates, orientation vectors etc.

- Nonlinear force recovery : for all elements

- Using the material model, for all elements :
  -Check for plastic deformations (yielding, unloading, etc.), find time-step length.

-Update plasticity information

- Recompute the element forces, and perform drift control algorithm. (If the time step has to be shortened due to the formation of a plastic hinge or such.)

- Compute the new member forces and reactions.

- Update the graphics display. (Only after every n steps)

OUTPUT : Time history of the response of the structure.

MODULE NAME : EXPLICIT INTEGRATION

CLASSIFICATION : ALGORITHM SPECIFIC

DESCRIPTION : Given the acceleration, velocity, and displacement vectors; and the mass, damping, and stiffness matrices, the new acceleration, velocity, and displacement vectors are computed. It should be noted that the assembled global stiffness matrix is not required here since all computations can be made on a dof basis. Alternatively, computations can be attempted on a nodal basis.

CALLED FROM : EXPLICIT DRIVER

CALLS :

INPUT : - Acceleration, velocity, and displacement vectors
- Element Stiffness, mass, and damping matrices

PROCEDURE : - Compute the new acceleration using the equilibrium equation.
- Compute the velocity and displacement vectors using the central difference equations.

OUTPUT : - New acceleration, velocity, and displacement vectors.

MODULE NAME : EFFECTIVE STIFFNESS AND LOAD

CLASSIFICATION : ALGORITHM SPECIFIC

DESCRIPTION : Using the Newmark-beta or the Wilson-theta equations the effective stiffness and load to be used in solving the system of linear equations at every time-step can be formulated.

CALLED FROM : IMPLICIT DRIVER

CALLS :

INPUT : Mass, damping and stiffness matrices;
Displacement, velocity, and acceleration vectors

PROCEDURE :-Loop over the number of non zero elements in the global stiffness matrix and perform the computations required to obtain the effective stiffness matrix.

-Loop over the total number of degrees of freedom and compute the terms of the effective load vector.

OUTPUT : Effective stiffness matrix and load vector.

MODULE NAME : INFO_MAIN

CLASSIFICATION : STRUCTURAL ANALYSIS UTILITY

DESCRIPTION : This module is used by the analyst to obtain information pertaining to the geometry of the structure, its response, and the analysis parameters. It can also be used to modify the corresponding information which involves writing new values into the database.

CALLED FROM : Several places.

CALLS : INFO_STRUCTURE
INFO_PARAMETERS
INFO_RESPONSE

INPUT : Coordinates of the nodes, topology, loading information, nodal masses,
element section properties, material properties,
transformation matrices,
plasticity (yield surfaces etc.) parameters,
nodal displacements, element forces,

analysis algorithm parameters,

flags to indicate physical device on which the information is requested, the type of graphical plots required.

PROCEDURE : Determine the type of information requested, and call the appropriate module.

Each module accesses the database for the specific information requested, and display it on the graphics terminal or the text monitor.

If the user wishes to change some information, an appropriate input medium (keypad, mouse, etc.) to communicate with the program is provided. This information is then stored in the database using the appropriate query routines.

OUTPUT : Required information, graphically or through text and an updated database.

MODULE NAME : MATERIAL STIFFNESS

CLASSIFICATION : STRUCTURAL ANALYSIS UTILITY

DESCRIPTION : The component of the stiffness matrix due to the material nonlinearity (K_i) is computed here.
Recall : $[K] = [K_e] + [K_g] + [K_i]$

CALLED FROM : Element Stiffness module

CALLS : Routines specific to the element type

INPUT :

- Element Type.
- Number of nodes and degrees of freedom at each node.
- Nodal coordinates, Orientation of the member
- Topology (Nodal connectivity)
- Member properties (A, J, I, d, etc.)
- Material properties (Yield stresses, Calibration parameters)
- Inelastic state information
    - Hinges formed, hinges hardening in steel members
    - State of cracking in concrete
    - State variables associated with the analytical material model.
- Damage or ductility information
- Flags for
    - Analysis type, i.e., Explicit or Implicit
    - Nonlinearities to be included

PROCEDURE : For an element of the structure :

- Call the routine which computes the material stiffness terms for the type of element being considered. e.g. : For beam-column elements, call module MATERIAL STIFFNESS BEAM-COLUMN.
- Store the material stiffness in the stiffness database.

OUTPUT : Material stiffness component, K_i.