158 KSME Journal, Vol. 10, No. 2, pp. 158~168, 1996

An Integrated CAE System for Dynamic Stress and Fatigue Life Prediction of Mechanical Systems

Hong Jae Yim* and Sang Beom Lee**

(*Received May 2, 1995*)

This paper presents computer simulation methodology for dynamic stress time history computation to predict the fatigue life of machine components using flexible multi-body dynamics. A hybrid method which employes stress superposition as a function of constraint loads and component accelerations that are predicted by flexible body dynamic simulation is utilized and implemented using established codes. A system integration methodology for dynamic stress computation of mechanical system components is described to provide a usable environment for an engineer. It uses a database management system such as the IAC and the established dynamics and finite element analysis codes.

Key Words : Dynamic Stress, Fatigue Life, Flexible Multi-body Dynamics, Dynamic Load, Finite Element Analysis

1. Introduction

When designing a mechanical system, stress and deformation in structural components must be predicted in the design process in order to satisfy performance requirements and guarantee that loads imposed on the component are safely supported. To record the dynamic stress or strain history in a component of a mechanical system in operation, experinental method have long been used. The experiments, however, always have required prototype components that are tested in a system dictated environment. To reduce the time required for a design change cycle, cost effective and time saving approaches are needed. Analytical and numerical dynamic stress analyses of mechanical components have traditionally been carried out with estimated critical loads and a finite element quasi-static analysis method that neglects vibration effects. Vehicle components such as the frame and B. I. W. (body-in-white) structures are elastic, however, so they deform when they are subjected to external loads and inertia loads. This elastic deformation can not be neglected in predicting loads on vehicle components for accurate stress analysis. Large scale general purpose dynamic analysis codes such as DADS(1986) and DISCOS (Bodley, et. al., 1978) have been developed for flexible dynamic analysis. Even though these general purpose codes give accurate joint reaction forces, displacements, velocities, and accelerations of flexible bodies, dynamic stress calculation is not supported.

The time histories of dynamic stresses can be calculated with the aid of existing analysis codes, without developing a new independent code that would require a huge amount of effort. In the implementation of these computational methods, a large amount of data must be communicated between analysis codes. The selected data sets that are generated from analysis codes must be manipulated to calculate dynamic stress time histories. Therefore, a command processor system must be developed to launch application and support programs. A database management sys-

*Dept. of Mechanics and Design, Kookmin University

**Dept. of Mechanics and Design, Graduate Scool, Kookmin University

tem that supports unified data structures is necessary for effective data communication between different analysis codes. With the command processor and database management systems, existing analysis codes can be integrated into a single system for complete dynamic stress analysis.

An integrated analysis system for dynamic stress and fatigue life prediction of mechanical system components is proposed in this paper, utilizing existing system integration technology.

2. Flexible Body Dynamic Simulation

The rigid body dynamics approach has been commonly used for mechanical systems, which are simulated by assuming that its components are modeled as rigid. This method may be very useful for very stiff mechanical systems, in which all the components behave essentially as rigid bodies. However, this method does not consider the flexibility on dynamically induced loads, which can be quite important for high speed and/or highly flexible mechanical systems. In the flexible body dynamic approach, a mechanical system is simulated by assuming that flexible component structures are considered as flexible bodies.

Flexibility information is obtained by finite element vibration analysis of the component structure, in terms of modal mass and stiffness matrix, lumped mass, and inertia. Using a well established finite element structural analysis code such as NASTRAN or ANSYS, deformation modes along with mass and stiffness matrices are computed and used for flexible dynamic analysis. Deformation modes may include vibration modes, static correction modes, and Ritz modes. An intermediate processor selects a set of modes and computes the corresponding modal mass and modal stiffness matrices. Using flexibility information, a general purpose flexible body dynamic system simulation code such as DADS is used to simulate dynamics of the flexible mechanical system.

For given motion, flexible dynamic simulation gives results necessary for calculating load histories on mechanical components, such as reaction forces at kinematic joints, finite element nodal displacements, velocities, and accelerations of nodes that were used in structural analysis to obtain deformation modes. Distributed inertia loads on the flexible mechanical system are computed as nodal loads, lumped at the finite element node points that were used for deformation mode analysis. Some node points must be located at kinematic joints to compute stress fields due to reaction forces that are supplied by a dynamic simulation code.

NASTRAN supports inertia relief approach, in which inertia loads are computed for the components that experience rigid body motion. In the inertia relief approach in NASTRAN, acceleration is computed using the rigid body motion only. However, structural components often experience nonlinear large gross rigid body motion that is coupled with linear elastic deformation, particularly when a mechanical system experiences non-linear motions, such as a rotational motion. Therefore, in computation of inertia loads for such components, acceleration must be computed with rigid body acceleration that is coupled with elastic deformation.

Figure 1 shows the finite element model of the

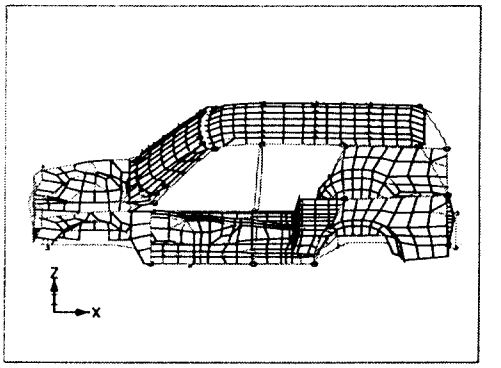

Fig. 1 Finite element model of the frame and B. I. W. structure

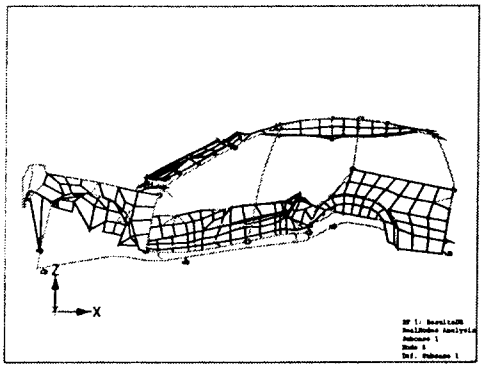

Fig. 2 A vibration mode of the frame and B. I. W. structure

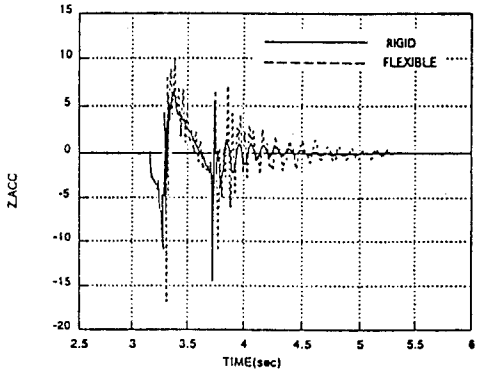

Fig. 3 Vertical acceleration of a point of the vehicle structure

vehicle structure used in this paper.

As an example, two dynamic simulations have been executed for a vehicle, which moves over a proving ground pothole. For flexible vehicle dynamic simulation, frame and B. I. W. structure is modeled as finite element model, as shown in Fig. 1. Figure 2 shows a vibration mode, which is used as one of deformation modes of the vehicle structure. For rigid body dynamic simulation, frame and B. I. W. structure are modeled as a rigid body, whose total mass is lumped at the center of gravity. Figure 3 shows the vertical accelerations of a node point of chassis frame, in which two simulation results are compared. As shown in the figure, larger magnitude of accelerations is obtained from flexible body simulations than from rigid body dynamic simulations. This

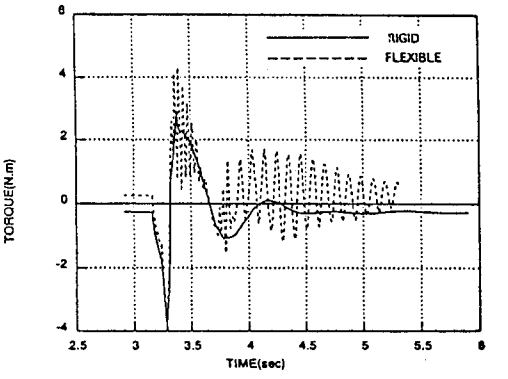

Fig. 4 Joint reaction torque of the bushing of the front lower control arm

is because coupled large gross motion and elastic deformation cause large magnitude acceleration of the chassis structure, as mentioned above. Figure 4 shows the comparison of joint reaction torque of the bushing at the front left lower control arm. Larger joint reaction forces are also generated from flexible simulation rather than from rigid dynamic simulation, since larger magnitude accelerations cause larger inertia loads and thus larger reaction forces.

3. Stress Computation Procedure

Finite element stress analysis is done for the given flexible component, for each unit load set at finite element node points for which an inertia load or reaction force is computed. The total stress field in each mechanical system component can be expressed as a combination of reaction forces at joints and inertia loads. Quasi-static finite element structural analysis utilizes D'Alembert's principle, in which inertia loads are applied to the mechanical component to satisfy static equilibrium conditions. A stress field due to a unit load associated with each component of each reaction force and each inertia load component can be computed.

Using the reaction force and inertia load histories obtained from the flexible body dynamic

simulation output and stress fields due to unit loads, dynamic stress histories in critical areas of flexible mechanical components can be computed by linear superposition. Life of mechanical components can be predicted using these stress histories. Figure 5 shows an outline of the methodology and conceptual data flow from structural finite element analysis to flexible body dynamics, stress history calculation, fatigue analysis, and life prediction.

When the mode superposition method is used, convergence of stresses is slower than convergence of displacements. In other words, many modes are required to obtain accurate stresses. For mechanical system components that experience large motion, which makes the system highly nonlinear, global system modes as well as component modes may influence the internal stress behavior. Considering inertia loads that are coupled with flexible body inertia, more accurate stresses can be computed as described in the previous section(see Fig. 5). Detailed theoretical procedures are described in this section.

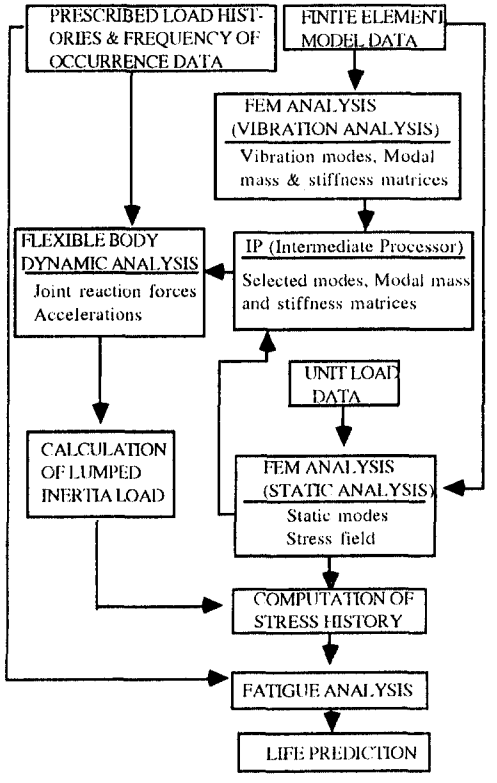

Fig. 5 Improved dynamic stress calculation and fatigue life prediction methodology

After motion and time histories of the flexible body dynamic system are obtained, inertia load distribution can be computed for each flexible component. The lumped inertia load at node point P of a flexible body can be computed as (Yim and Haug, 1990)

$$m\ddot{r}^{p}=m(\ddot{r}+A\tilde{\omega}'\tilde{\omega}'s'^{p}-A\tilde{s}'^{p}\dot{\omega}' +2A\tilde{\omega}'\dot{u}'^{p}+A\ddot{u}'^{p} \quad (1)$$

where m is the lumped mass at node point P, $\ddot{r}^{p}$ is the acceleration of point P, A is the transformation matrix from the x'-y'-z' body reference frame to the x-y-z inertial reference frame, $\tilde{\omega}'$ is the matrix form for the vector product of angular velocity, u'^{p} is the elastic deformation in the body reference frame, and s'^{p} is the displacement vector from the origin of the x'-y'-z' frame to point P in the deformed states, respectively.

For the convenient use of finite element analysis codes, all boundary conditions, joint reaction forces, and inertia load are computed using the local body reference frame. Then although flexible mechanical components are moving in the space, only one finite element model configuration needs to be used for the different configurations experienced by a component during motion. The inertia load vector can be rewritten in the body reference frame as

$$mA^{T}\ddot{r}^{p}=m(A^{T}\ddot{r}+\tilde{\omega}'\tilde{\omega}'s'^{p}-\tilde{s}'^{p}\dot{\omega}' +2\tilde{\omega}'\dot{u}'^{p}+\ddot{u}'^{p} \quad (2)$$

Then, the so called dynamic equilibrium equation can be obtained by applying the D'Alembert's principle,

$$M(A^{T}\ddot{r}+\tilde{\omega}'\tilde{\omega}'s^{p}-\tilde{s}'^{p}\dot{\omega}'+2\tilde{\omega}'\dot{u}'^{p} +\ddot{u}'^{p})+Ku=F$$

or

$$M\ddot{u}+2M\tilde{\omega}'\dot{u}+Ku=-M(A^T\ddot{r}+\tilde{\omega}'\tilde{\omega}'s'-\tilde{s}'\dot{\omega}')+F \quad (3)$$

where M and K are the mass and stiffness matrices for the finite element model of each flexible body, F is the joint reaction force, which is obtained from the previous flexible system dynamic analysis.

If complete nodal coordinates are used, the differential Eq. (3) for dynamic equilibrium would be very expensive to solve, because of the large number of nodal degrees of freedom. Instead, a quasi-static equilibrium equation may be solved by transferring the first two inertia load terms on the left of Eq. (3) to the right side of the equation. In this equation, existing information, which is generated from flexible body dynamic simulations, is used for the elastic acceleration, velocity, and the inertia load. Expressing the elastic acceleration and velocity vectors in terms of modal vectors, modal coordinate velocities and accelerations that are computed in the flexible body dynamic solution, Eq. (3) can be written as

$$Ku=-M\ddot{u}^*-2M\tilde{\omega}'\dot{u}^*-M(A^T\ddot{r}+\tilde{\omega}'\tilde{\omega}'s'-\tilde{s}'\dot{\omega}')+F \quad (4)$$

where

$$\dot{u}^*=\Psi\dot{a} \quad (5)$$

$$\ddot{u}^*=\Psi\ddot{a} \quad (6)$$

and Ψ, $\dot{a}$, $\ddot{a}$ are the modal matrix, modal coordinate velocity and acceleration vectors, respectively.

Substituting Eqs. (5) and (6) into Eq. (4) yields

$$Ku=-M(\Psi\ddot{a}+2\tilde{\omega}\Psi\dot{a})-M(A^T\ddot{r}+\tilde{\omega}'\tilde{\omega}'s'-\tilde{s}'\dot{\omega}')+F \quad (7)$$

The first term on the right side of Eq. (7) represents flexible inertia effects, the second term represents gross motion inertia effects, and the third term includes joint reaction forces and any spring or damper forces.

A new displacement field can now be computed by solving Eq. (7), using any finite element code as in Eq. (8),

$$u_{\text{new}}=K^{-1}[-M\Psi\ddot{a}-2M\tilde{\omega}'\Psi\dot{a}-M(A^T\ddot{r}+\tilde{\omega}'\tilde{\omega}'s'-\tilde{s}'\dot{\omega}')+F] \quad (8)$$

The new displacement field can be compared to the old displacement field that is represented by the original deformation modes used in the flexible body dynamic simulation. If the new displacement field is dissimilar to the displacement field that was calculated in the flexible body dynamic simulation, mode selection should be reconsidered and, if necessary, redefined and the flexible dynamic simulation was repeated with improved modal vectors. Once the displacement field is computed, stresses can be computed directly from this new displacement field by spatial differentiation.

In using finite element models for dynamic stress analysis, it is apparent that a large degree of freedom model is often inefficient. This motivates substructuring techniques in flexible dynamic and stress analysis, by which a large model is broken into a number of components. The advantage of substructuring is that it allows local stress analysis of only a reduced substructure with enhanced accuracy. If internal displacements and stresses in any substructure are not required, they need not be computed. Typically, substructures will be identifiable parts of the total structure. For example, in a chassis frame, the side rail, engine mount cross member, and rear suspension cross member may be substructures.

4. Fatigue Life Prediction

Mechanical components in real service environments experience irregular stress histories. "Cumulative damage," which is defined as fatigue effects of stress events other than uniform cycles, defines fatigue life in such cases (Miner, 1945). The nominal stress approach, which uses

the cumulative damage concept, can be utilized to predict fatigue life of machine components. To relate the effect of irregular stress histories to **S-N** or **e-N** curves, which are obtained with uniformly repeated simple stress cycles, a rule is necessary to count cycles and account for the range from the highest peak to the lowest valley. The so called "rain flow method" is currently the most popular for cycle counting (Fuchs and Stephens, 1980). Stress ranges counted by the rain flow method are converted to nominal stress ranges. The contribution to damage from each of the stress ranges is calculated from the S-N curve. Damage caused by one cycle is defined as 1/N, where N is the median number of repetitions of this same cycle that leads to failure. The Palmgren-Miner rule, which states that fatigue failure occurs when the sum of the incremental damage due to all blocks is equal to unity (Miner, 1945), is then utilized to compute the fatigue life of the component.

5. Integrated System

To compute dynamic stress time histories of mechanical components using the proposed method, many different analysis codes are required (Dopker and Yim, 1988). To manage the different application codes and data in this computational procedure, an integrated system must be utilized. An integrated software system for dynamic stress analysis is presented in this section. The object oriented database and application management system are based on a layered approach, where a network manager sits on top of a functional object manager, on top of an object oriented executive and data access, on top of a database and application management system. In this approach, information about data needs of different functional modules is imbedded in the various layers of the system. The general purpose CAE software environment developed in Ref. 6 is constructed from three components : (1) a central database and application management system, (2) an expert system, and (3) an advanced user interface system. In this way, the flow of engineering analysis data between major analysis and pre - and post-processing steps in the design process is managed by a single database management system. Codes that perform a specific analysis or pre/post-processing functions are referred to functional objects. Information that defines the processing capabilities of the different functional objects, their input and output requirements, and data objects in the database are managed by a single expert system. Communication between the engineer and the integrated system is managed through a unified user interface system.

5. 1 IAC (Integrated Analysis Capability)

Recently, a number of integrated systems have been developed. A flexible integrated database and command language that represents the current state-of-the art is the Integrated Analysis Capability (IAC) (Vos, et. al., 1983), developed by Boeing for NASA/Goddard. The IAC provides the capability of storing all kinds of data structures in the IAC database, concurrent data access, data cataloging and archiving tools. IAC facilitates the flow of data between different analysis softwares or between a module and the user, by providing a central database storage area, standard data structures and formats, and various data management tools. In order to serve the needs of diverse analysis codes and users for communicating with the database, IAC supports the interfaced data flow techniques. This technique can be used effectively where module source code is not available or the software is difficult to modify, and the module is capable of reading or writing the required data via a module defined file. The executive control module is an important part of the overall IAC system. In the usual mode of operation, it provides the complete user interface, and the access to all other analysis

codes in the system. It also uses the same system utilities and has the same operating privileges available to the other IAC to perform both interactive and batch tasks. The interface between the executive and the other modules is designed to support a relatively simple "plug-in" implementation for new modules. This integrated system capability is particularly useful for linking existing analysis codes, without source codes, since it allows a user to add new interface modules for implementing analysis programs in the integrated system.

The integrated system can generates generic NASTRAN and DADS input files based on the structural, kinematic, and dynamic data of the mechanical system. The IAC can launch DADS simulation run. Once a mechanical dynamic simulation is completed, time history of acceleration and joint reaction forces of the selected components are exported to the IAC database. During the process, engineer can check the simulation results to see whether they are acceptable or not. It can then launch modules to compute dynamic stress time history and predict fatigue life of the mechanical system, sequentially. Analysis tools, data structures, and command processor system for the integrated system proposed in this paper are summarized here.

5. 2 Analysis tools

To calculate dynamic stress time histories in mechanical components using the hybrid method, two different analysis softwares are needed. One is software for finite element structural dynamic analysis. This software analyzes deformation modes that are required for flexible multibody system dynamic analysis and computes stress influence coefficients. For this purpose, an existing finite element analysis code such as NASTRAN or ANSYS can be used. The other software that is needed for flexible multibody system dynamic analysis deals with large gross motion nonlinearity and small elastic deformation. This software computes time histories of modal coordinates, constraint reaction forces, and accelerations. For this purpose, DADS or DISCOS can be used. In this paper, the NASTRAN and DADS codes are used as basic software in development of an integrated analysis system.

For complete dynamic stress analysis, it is necessary to communicate some data sets between analysis codes. For example, vibration modes, modal mass matrix, modal stiffness matrix, and static deformation modes are read from NASTRAN vibration and static analysis output files and manipulated to obtain orthonormalized deformation modes. They are then used as input for DADS flexible dynamic analysis. An intermediate pre-processor (IP) is needed for this purpose in addition to the NASTRAN code. The IP is supported as a pre-processor in the DADS code. With these component flexibility data, a mechanical system is simulated using the DADS code. Constraint reaction forces and accelerations are obtained from DADS binary output files. Stress influence coefficients are read from NASTRAN static analysis output files. All these data sets must be stored in the central database, in unified structured data formats.

5. 3 Data structures

Most well established analysis software systems provide their own database management system; i. e., data for input and output are stored in their unique format. The NASTRAN code requires its own unique format for input data and stores analysis results in binary and/or ASCII output files. Likewise, the DADS code supports its own input and output data structure format, storing simulation results at every time step in ASCII and/or binary files. However, only a small fraction of data sets is required for dynamic stress analysis using the hybrid method.

In this section, data sets that are required for

Table 1 Data structures for the hybrid method

Data Structure	Calss	Contents
GRID	RELATION	Node ID, node location
CONNEC	RELATION	Element number, connectivity
MMASS	ARRAY	Modal mass matrix
MSTIF	ARRAY	Modal stiffness matrix
SDOF	RELATION	System degree of freedom, node ID, nodal degree of freedom,
NOD_IND	RELATION	All node identifications and references
NOD_MASS	RELATION	Node ID, nodal mass properties
MDIS_NODi	RELATION	Modal displacement vector for mode i, node ID,
SDIS_NODi	RELATION	Load displacement vector for mode i
CNSTR_FORk	ARRAY	Constraint reaction force vector for joint k, node ID, time ID
ACCL_NODi	ARRAY	Acceleration vector for node i, node ID, time ID
STR_ULOADi	ARRAY	Stress coefficients for unit loads at node i, node ID
STR_UACCLi	ARRAY	Stress coefficients for unit accelerations at node i, node ID
STRESS_TIME	ARRAY	Stress time histories, node ID, time ID

dynamic stress computation using the hybrid method are defined in unified standard formats. In the standard format all data are either in a relational or an array structure and the name of each data structure consists of four parts: (1) a basic name that associates the data structure with a structural component, (2) a 'number' construct to support parameter variation studies, (3) a type that identifies the data type of the relation/array (e.g. modal mass matrix), and (4) a version number. If data such as component geometry, spring coefficient, damping coefficient, and etc. are changed, and a new version number is given, then the design history of the mechanical system can be traced. Table 1 shows the data structures that are required for the hybrid method. Those data structures must be constructed in the central database, according to parameters that will be given when running interface programs.

6. Command Processor System

6. 1 Task flow

To establish a command processor system using the IAC, task flow must be defined first. As discussed in the previous section the hybrid method can be implemented by carrying out finite element structural analysis and flexible multibody dynamic simulation. The task flow for computation of dynamic stress time histories for mechanical system components is shown in Fig. 5.

For each component that is modeled as a flexible body in the mechanical system, either vibration modes or vibration modes combined with static correction modes are computed to represent the flexibility in the component, using the finite element method. Flexible data pre-processing generates input data that will be used in flexible multibody dynamic simulation. A combination of rigid and flexible bodies may be used for system components, depending on the flexibility of each component. The dynamic simulation computes constraint reaction forces and accelerations on each component. Stress influence coefficients are computed for each unit loads at constraint locations and for unit accelerations at each node or at each subset of nodes. Dynamic stress time histories are then computed, by linear superposition of stress influence coefficients multiplied by corresponding magnitude of constraint reaction forces and accelerations. The data input and output for each component are defined in Fig. 5.

In the previous procedure, since some data sets from one analysis are used as input data in a

subsequent analysis, considerable man-hours are expended for the preparation of computer input, both in construction of the mathematical model and transfer of data from one analysis to another. If errors are involved in a data set; e.g., use of inconsistent unit systems or reading data in the wrong format, it is very difficult to detect such errors in complicated data flows with large amounts of data, particularly for a large mechanical system such as a vehicle. Therefore, it is necessary to have a command processor that controls execution of software systems and a well organized database management system that supports unified data structures for engineering data. In the following sections, an integrated software system for dynamic stress analysis that employs the hybrid method is designed.

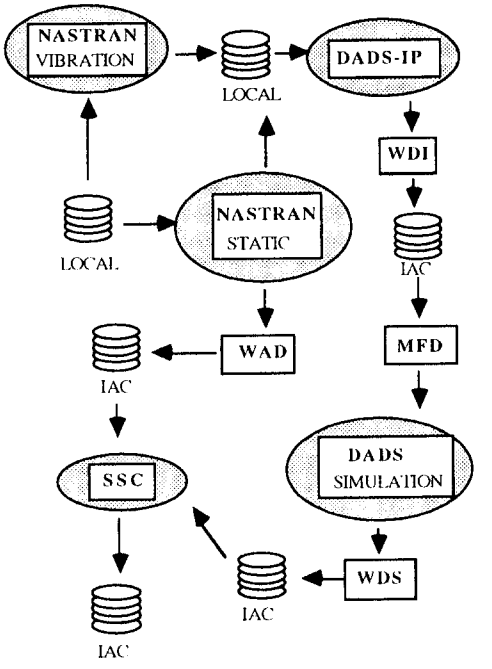

Fig. 6 Program organization for the intrgrated system

6. 2 Organization of programs

Based on the task flow described in the previous section, a run stream of analyses, organization of analysis codes, interface programs, and databases are defined in Fig. 6. Vibration and static correction modes are computed by carrying out vibration and static analyses, respectively, using the NASTRAN code. Data that are needed for the intermediate processor are read directly from the NASTRAN output files. The output that is generated from the intermediate processor is written to the central database, i. e., the IAC database, by interface program WDI. Interface program WDI reads data sets from the DADS-IP output file, reformats them into standard relations and arrays, and stores them in the IAC database. For flexible multibody dynamic simulation, the DADS code is utilized. Since DADS source code is not available, all data to/from the database from/to the code must go through interface programs MFD and WDS. MFD is an interface program that reads flexibility data in the IAC database, transforms them into the DADS input format, and stores them in the local database. WDS is an interface program that reads data sets from DADS binary output files, reformats them into standard relations or arrays, and stores them in the IAC database. Stress influence coefficient data are computed by carrying out NASTRAN static analysis. The relevant data from the NASTRAN output file are written to the IAC by interface program WAD. A small module SSC is needed for computing stress time histories, using stress influence coefficients and time histories of constraint reaction forces and accelerations that are stored in the IAC database. This module can be written as an interface program, so that communication between the codes and the IAC database may be direct.

6. 3 Module implementation in IAC

This section describes the approach for implementing interface programs within IAC. To implement module in IAC, the module software, RUN table, and one additional row to the MODULES. MOD table must be provided. The RUN

table is specific to a particular module and contains one row for each parameter available, via the executive RUN command for that module. The MODULES. MOD table pertains to all IAC modules and contains one row for each module. The parameter file is an IAC standard interface for passing parameters between different software procedures and programs; i.e., it connects the executive program in IAC with the module. It contains keyword parameter definitions and is usually generated by the executive program in IAC.

The executive program in IAC performs the following sequence of tasks for executing a module:

① Process the user's RUN command for the module, including any keyword supplied for parameter definitions.

② Create a RUNID consisting of an 8-digit RUN identifier unique to this invocation.

③ Create the parameter file and store this file in the user directory.

④ Spawn a subprocess to execute the IAC command procedure. The parameter file-spec is passed to the IAC procedure, in order to make the parameter file accessible to the module associated software. The specification is of the form lrunid. PPP, where runid is the value of the RUN identifier RUNID.

⑤ Execute the module software, via the host command "modjcl" where modjcl is the value of the MODJCL attribute in the MODULE. MOD table. This command initiates execution of the module software; e.g., module command procedures, executable programs, etc. The module software typically consists of a command procedure which runs an executable module

The IAC interface module WDS can be executed within an IAC user defined path. The general RUN schematic for executing an IAC interface module WDS, within IAC is

RUN WDS(F=XCAR, D=STUDY1, B=CHASSIS, O=JOINT)

The F parameter in the RUN statement is a basic file root name, used to identify input files. It will open the DADS output binary file that will be read by WDS. The D parameter gives the name (or optionally the name:number) for all the relation and array files that are generated by WDS to the IAC database. The B parameter selects the body of interest. The O parameter selects options that determine the specific data sets to be processed; i. e., this parameter selects what kinds of IAC relation or array files are going to be put in the database. For example, JOINT will select reaction force table for all joints in the CHASSIS component. WDS then reads the XCAR. BIN file and creates and inserts the array files into the IAC database corresponding to option O, and defaults values for the TYPE and AREA parameters.

7. Conclusions

In this paper, an integrated system for dynamic stress and fatigue life prediction of mechanical components has been proposed by developing a data structure which defines a mechanical system and interface programs that support multidisciplinary computer-aided simulation and design activities. To facilitate and manage data sharing among multidisciplinary engineers, IAC oriented database has been utilized. Whenever multidisciplinary engineers exchange mechanical system data through the global database and local database, version control in the design process can be accomplished. As an example, an integrated system for a vehicle system has been presented.

References

Bodley, C. Devers, A., Park, A. and Frish, H.,

1978, "A Digital Computer Program for Dynamic Interaction Simulation of Controls and Structures (DISCO)," NASA Technical Paper 1219.

DADS User's Manual, 1986, Rev. 4. 0, *Computer Aided Design Software,* Inc., Oakdale, IA.

Dopker, B. and Yim, H. J., 1988, "A Command Processor System with an Integrated Data Base for Flexible Mechanical System Dynamics," *Advances in Engineering Software,* Vol. 10, pp. 15~21.

Fuchs, H. O. and Stephens, R. I., 1980, *Metal Fatigue in Engineering,* John Wiley Sons.

Miner, M. A., 1945, "Cumulative Damage in Fatigue," *Journal of Applied Mechanics,* Vol. 12, pp. 159~164.

Vos, R. G., Walker, W. J., Beste, D. L., Price, G. A., Young, J. P. and Frisch, H. P., 1983, "Development and Use of an Integrated Analysis Capability," AIAA/ASME/ASCE/AHS Structures, Structural Dynamics and Materials Conference, Lake Tahoe, NV, *AIAA Paper 83-1017.*

Yim, H. J. and Haug, E. J., 1990, "Computational Methods of Dynamic Stress Analysis of Mechanical Systems," Technical Report R-84, Center of Simulation and Design Optimization of Mechanical Systems, The University of Iowa.