

A General Finite Element Analysis Data Management Technology

Yu Zhou, Fen Yan

Institute of Nuclear Energy Technology, Tsinghua University, Beijing, 100084, China

ABSTRACT

It is important to manage amount of data from nuclear reactor component design and stress analysis. There are more and more requirements to support data exchange and data share among CAD, CAM and CAE. Finite Element Analysis (FEA) is one of CAE technologies that are widely applied during nuclear engineering component design process. Nowadays, data management technologies for CAD/CAM are concerned much more than those for CAE. This paper tries to find out a way to build a general FEA data management platform that can deal with the FEA data exchange and data share. A neutral FEA data set file format is suggested first, and then FEA database structure based on Database and MIS technology is discussed. Finally, data management programs are developed for some general FEA applications, such as MSC/Nastran.

1 INTRODUCTION

Finite Element Analysis (FEA) is one of the most important CAE technologies that used for the engineering component stress analysis. FEA technology was founded in 1950s and developed rapidly in 1970s and 1980s, but for a long time the data exchange and share among different FEA applications or between FEA and CAD applications are not considered on a uniform platform. Historically, most of FEA applications had not been paid enough attentions on the data exchanging and sharing. Nowadays, more and more FEA applications are facing the strong requirements from end users about the convenient and uniformed data exchange and share, and more, data management.

The current CAD/CAM applications show advantages on the data exchanging and sharing. The relative technologies have been developed in CIMS. Some products data exchange standards, such as IGES, PDES/STEP, have been published.

Hence, compared with CAD/CAM product data exchange and share technology, FEA data exchange and share should keep the compatible with the technologies used in CAD/CAM on principle and implementation method. Based on the reason and consideration, FEA data exchange and share can be constructed on PDES standard.

In order to deal with the FEA data not only to exchange or share the data, but also to manage the data, the developing data management technology, such as Database or MIS, can be considered.

There are two ways to manage FEA data. One is by neutral file; another is by database management technology. Here, the neutral file way will be discussed firstly, and then database way.

2 FEA DATA SET DESCRIPTION

Finite Element Analysis is widely applied in the engineering design and analysis. As a numerical analysis technology, FEA can be used to solve field problems. The field problems are listed in the following:

Solid Problems

Fluid Problems

Thermal Problems

Other Fields Problems.

Considering the common characteristics of FEA for different fields, FEA data set should include the following parts for FEA data exchange (see Table 1.)

Global Information

Discrete Information

Auxiliary Information

Solution Information
Results Information

Table 1. FEA Data Classification

Global Information	<i>File Description Information</i> <i>Units Information</i> <i>Basic Unit System</i> <i>Derived Unit System</i> <i>Coordination Information</i> <i>1D Coordination System</i> <i>2D Coordination System</i> <i>3D Coordination System</i>
Discrete Information	<i>Node Information</i> <i>Element Information</i>
Auxiliary Information	<i>Element Physical Parameters</i> <i>Element Material Parameters</i> <i>Beam Cross Section Parameter</i>
Solution Information	<i>Initial Information</i> <i>Boundary Set Information</i> <i>Load Set Information</i>
Results Information	<i>Data at Node</i> <i>Data on Element</i> <i>Data of Nodes on Element</i> <i>Others</i>

The above parts are considered as the basic structure of the FEA data set and a neutral file format or its database is constructed on it.

3 NEUTRAL FILE STRUCTURE OF FEA DATA

Based on the FEA data set structure, solid and stationary FEA is considered as the research focus.

The neutral file structure for FEA data exchange that based on PDES/STEP standard^{[1][2]} is suggested. The following is the description of neutral file structure definition that is by C Language.

(1) Global Information

The global information includes three parts, 'File Description', 'Unit System' and 'Coordination System'.

File Description Information	<pre> struct head { char FileName[256]; /* File name */ char Descrip[256]; /* File description */ TIME CreatedTime; /* File created time */ char Author; /* Author */; </pre>
Unit System	<pre> struct unitsys { int UnitNum; /* Unit system ID */ char UnitName[256]; /* Unit system name */ double LenSca; /* Length scale */ double MasSca; /* Mass scale */ Double TempSca; /* Temperature scale */ Double TempOff; /* Temperature offset */; </pre>

Coordination System	<pre> struct coordinate { int CoordNum; /* Coordinate system ID */ int CoordType; /* Coordinate type */ char CoordName[256]; /* Coordinate system name */ int CoordRefer; /* Referenced coordinate system */ POINT OriginP; /* Origin coordinates */ POINT X_point; /* The coordinate value of Point A on +X axis */ POINT Y_point; /* The coordinate value of Point A on +Y axis */}; struct point { double x_coor, y_coor, z_coor; /* The coordinates of a point */}; </pre>
---------------------	---

(2) Discrete Information

The discrete information includes 'Node Information', 'Element Information' with physical and material linkage information. For a solid component or part, it is usually divided into many elements that decided by nodes, such as triangular element type, rectangular solid element type, etc.

Node Information	<pre> struct node { int NodeNum; /* Node number */ int DeflCoord; /* Deflect coordinate system */ int DispCoord; /* Displacement coordinate system */ POINT Node_p; /* Node coordinate */}; </pre>
Element Information	<pre> struct element { int ElemNum; /* Element number */ char ElemType[6]; /* Element type name */ int PhysTab; /* Physics parameters table of element */ int MateTab; /* Material parameter table */ int ElemNode; /* Node numbers on a element */ int NodeTab[MAXNODEELEM]; /* Node table */ int ElemSpec[4]; /* Beam cross section specification parameters */}; </pre>

(3) Auxiliary Information

Some information to determine the solid component or part characteristics is defined in the paper. They are 'Element Physical Parameters', 'Element Material Parameters', and 'Beam Cross Section Parameters'.

Element Physical Parameters	<pre> struct physicaltab { int PhysNum; /* Physical parameter table number */ char PhysType[6]; /* Element type group */ char PhysName[256]; /* Physics table name */ float PhysTab[30]; /* Physics parameters */}; </pre>
Element Material Parameters	<pre> struct materialtab { int MateNum; /* Material number */ char MateType[6]; /* Material type */ char MateName[256]; /* Material name */ float MateTab[40]; /* Material characteristic parameter table */}; </pre>
Beam Cross Section Parameters	<pre> struct beamcrosssection { int CrosNum; /* Beam cross number */ </pre>

	<pre> char CrosType[6]; /* Beam cross type */ char CrosName[256]; /* Beam cross name */ float CrosFeat[10]; /* Beam cross feature */ float CrosData[30]; /* Beam cross characteristic data */; </pre>
--	---

(4) Solution Information

The solution information is used to determine the field problem's solution conditions, which includes 'Initial Information', 'Load Set Information' and 'Boundary Set Information'.

Initial Information	<pre> struct dofset { int DofSetNum; /* Degree of Freedom (DOF) set number */ char DofSetName[256]; /* DOF name */ int SetTotalNode; /* Total node number */ DOFDATA SetData[]; /* DOF set data of node */; struct dof { int NodeNum; /* Node number */ int Dofs; /* Node DOF */} DOFDATA; </pre>
Boundary Set Information	<pre> struct restset { int RestSetNum; /* Restrain set number */ char RestSetName[256]; /* Restrain set name */ int SetTotalNode; /* Total node number in the set */ RESTDATA SetData[]; /* Node data of the set */; struct restraint { int NodeNum; /* Node number */ int RestSwitch; /* Restrict switch */ float RestTab[6]; /* Restrict table */} RESTDATA; </pre>
Load Set Information	<pre> struct loadset { int LoadSetNum; /* Load set number */ char LoadSetType[6]; /* Load set type */ char LoadSetName[256]; /* Load set name */ int LoadTotal; /* Total load number */ LOADDATA SetData[]; /* Load data in the set */; struct loads { int LoadPos; /* Load position */ int LoadVect; /* Load vector */ int LoadPoint; /* Load point */ float LoadValue[6]; /* Load value */} LOADDATA; </pre>

(5) Result Information

The result information is used to restore the solution result that includes 'Data at Node', 'Data on Element', and others.

Data at Node	<pre> struct dataatnode { int NodeNum; /* Node number */ float Data[6]; /* Analysis result data at node */} DISPDATA; </pre>
Data on Element	<pre> struct dataonelem { int ElemNum; /* Element number */ float Data; /* Analysis result data on element */} ENERDATA; </pre>

Node Data on Element	<pre> struct datanodeelem { int ElemNum; /* Element number */ int NodeNum; /* Node number on element */ float Data[MAXNODEELEM][6]; /* Analysis data */ STREDATA; </pre>
Others	<pre> struct analysiset { char Desc[256],LoadName[256],LoadTitle[256]; /*Discription, Load name, title */ char Type[6]; /* Result data type */ int LoadNum; /* Load number */ int DataTotal; /* Total data number */ union { DISPDATA A1[]; /* Displace data at node */ ENERDATA A2[]; /* Energy data on element */ STREDATA A3[]; /* Stress data of node on element */ }; }; </pre>

4 ENTITY-RELATIONSHIP MODEL OF FINITE ELEMENT ANALYSIS DATA MANAGEMENT

Database technology can be considered to solve the FEA data management. Based on the neutral file structure of FEA, it is easy to describe the entity-relationship model of FEA data (see Fig. 2).

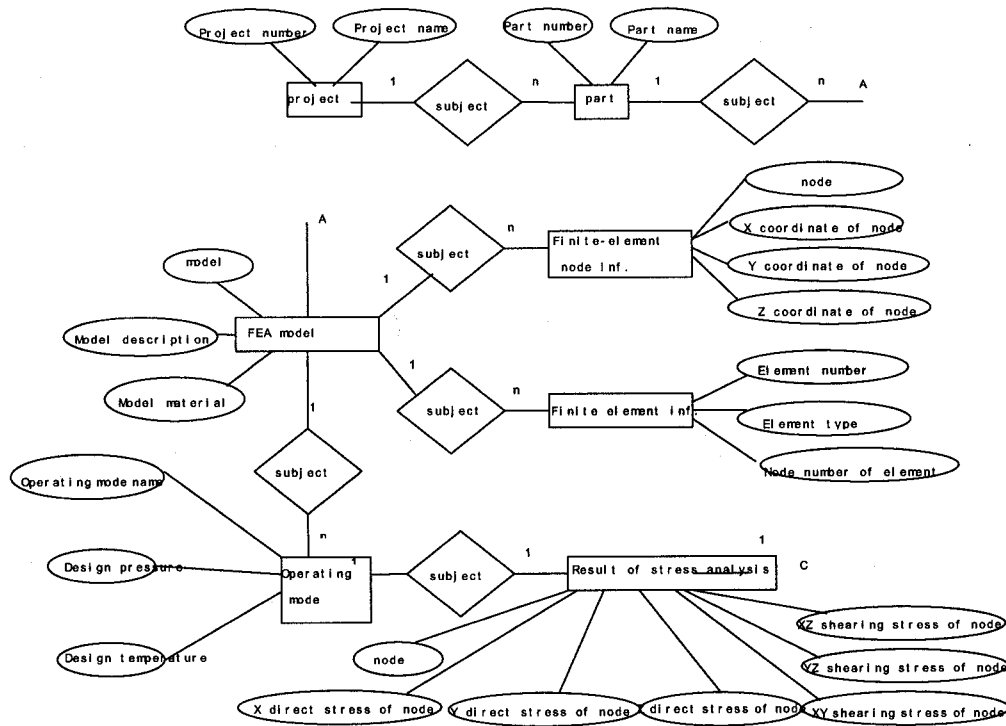


Fig.2 Entity-relationship Model of Component FEA

5 IMPLEMENTATION OF FEA DATA MANAGEMENT

5.1 Neutral File Way

The FEA neutral file format is built up. For the implementation, in our research, the most two popular FEA applications are selected as the testing objects. One is MSC/Nastran, another is IMAG/I-Deas.

An access interface package for FEA data exchanging and sharing is developed. It includes two main functions. The first one is to access data from a FEA application solution result, such as Nastran or I-Deas application, and the second one is to write the data in to a neutral file with the format and style suggested in this paper. For each FEA application, what need to do is only to create its own independent access interface program to PDES/STEP-based FEA neutral file.

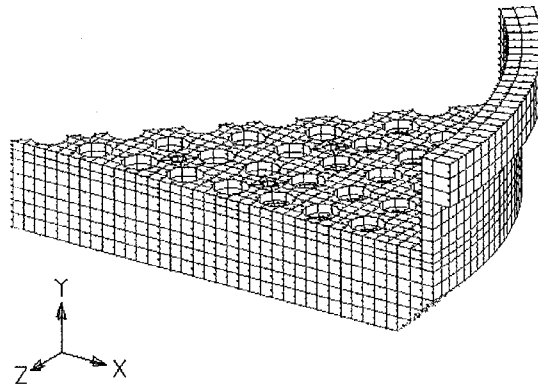
5.2 Database Way

For FEA data management, it should supply a uniform data exchange and share platform. A database application is developed on the entity-relationship model. The more detail of the database way will be in future papers.

5.3 Data Exchange

In our research, we developed the package that includes programs

<i>NAS2STEP</i> :	To access NASTRAN data file to FEA neutral file, or database.
<i>STEP2NAS</i> :	To access FEA neutral file to NASTRAN data file, or by database.
<i>IDE2STEP</i> :	To access I-DEAS data file to FEA neutral file, or by database.
<i>STEP2IDE</i> :	To access FEA neutral file to I-DEAS data file, or by database.



An implementation example, which is a discrete information of a cylindrical shell that included solid FEA elements and nodes information, is showed in figure 3.

6 CONCLUSION

A neutral file format is successfully constructed for FEA data exchange, data share. The method can bring benefits as showed in CAD/CAM. For FEA applications, if their own access interface was developed to suit for a standard styled neutral file, it would not need to develop so many interfaces for other FEA applications respectively but only the access to the middleware, that is FEA neutral file.

FEA database platform or system is more than FEA data exchange and data share. It shows a possibility to uniform data expression and management of different FEA applications.

REFERENCES

1. International Organization for Standardization, ISO 10303-1, "Industrial automation systems and integration - Part 1 Overview and fundamental principles", 1994.
2. International Organization for Standardization, ISO 10303-21, "Industrial automation systems and integration - Part 11 Description methods: The EXPRESS language reference manual ", First edition, 1994.
3. International Organization for Standardization, ISO 10303-21, "Industrial automation systems and integration - Part 21 Implementation methods: Clear text encoding of the exchange structure", First edition, 1994.
4. ZHOU Yu, YING Ming, The concept of FEA data base set and the development of the visualization software - DPP, Visualization Technology in Scientific Computation and Engineering Design, p.81 ~ p.87, The Press of Beijing Industrial University, 1995.4.