Transient Dynamics Capability at Sandia National Laboratories

S. W. Attaway, J. H. Biffle, G. D. Sjaardema,
M. W. Heinstein and L. A. Schoof
Sandia National Laboratories
Albuquerque, New Mexico

INTRODUCTION

This report will present a brief overview of the transient dynamics capabilities at Sandia National Laboratories, with an emphasis on recent new developments and current research. In addition, the Sandia National Laboratories (SNL) Engineering Analysis Code Access System (SEACAS), which is a collection of structural and thermal codes and utilities used by analysts at SNL, will be described. The SEACAS system includes pre- and post-processing codes, analysis codes, database translation codes, support libraries, Unix shell scripts for execution, and an installation system.

SEACAS is used at SNL on a daily basis as a production, research and development system for the engineering analysts and code developers. Over the past year, approximately 190 days of CPU time have been used by SEACAS codes on jobs running from a few seconds up to two and one-half days of CPU time. SEACAS is running on several different systems at SNL including Cray Unicos, Hewlett Packard PH-UX, Digital Equipment Ultrix, and Sun SunOS.

An overview of SEACAS, including a short description of the codes in the system, will be presented. Abstracts and references for the codes are listed at the end of the report.

Additional information about obtaining SEACAS can be obtained by contacting:

Marilyn K. Smith
Division 1425
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-5800
(505) 844-3082; Fax (505) 844-9297

PHILOSOPHY

SEACAS is a modular system based upon a common binary datafile format called EXODUS that includes the mesh description and the timeplanes of the computed results. A subset of this format, called GENESIS, is used to refer to the mesh description portion of the EXODUS format.

All of the preprocessing, analysis, postprocessing, and translation codes can read and/or write EXODUS database files. A schematic of this is shown in Fig. 1.

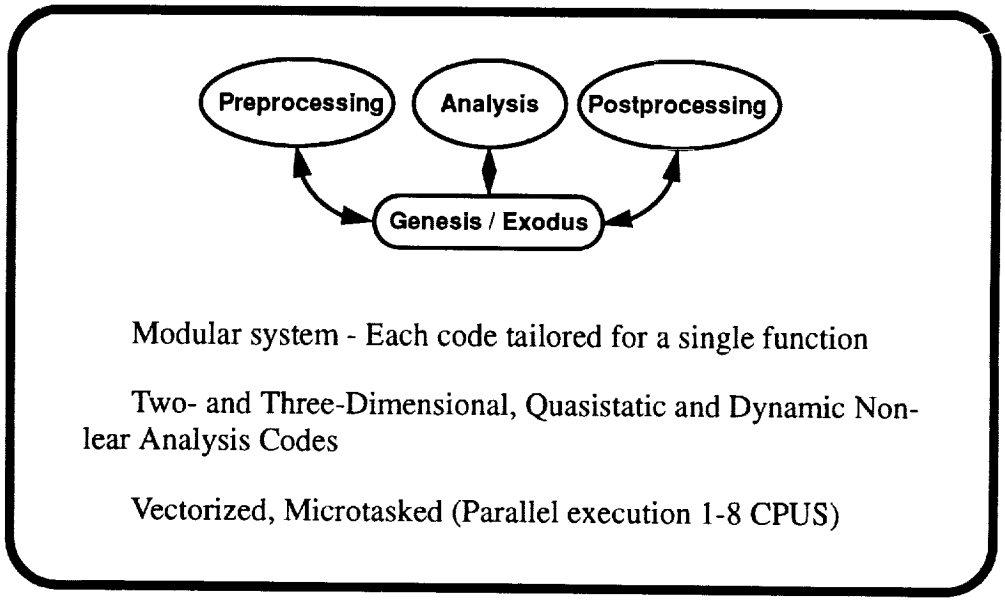

Figure 1 - Modular structure of SEACAS

With this structure, codes can be tailored for a single function. For example, an analysis code can be added to the system without writing new mesh generation and postprocessing programs. Also, code-to-code data transfers and restarts use the EXODUS format. A more complete view of this is shown in Fig. 2.

This modular concept is used extensively in the mesh generation process. Several special-purpose codes have been written that perform a certain mesh generation task. Examples of this are:

- generate two-dimensional mesh (fastq)
- transform two-dimensional mesh into a three-dimensional mesh (gen3d, genshell)
- join two or more 2D or 3D meshes into a single mesh (gjoin)
- reposition a 2D or 3D mesh (grepos).

Finite element meshes of several complicated three-dimensional geometries have been successfully generated using this system, which is built up of codes that singly provide a limited capability, but when used as a system are very powerful.

All of the codes are written in as portable a form as possible. Fortran codes are written in ANSI Standard FORTRAN-77 and C language codes are written in ANSI Standard C where possible. Machine-specific routines are limited in number and are grouped together to minimize the time required to adapt them to a new system.

A code management system is used for all of the SEACAS codes to provide traceability and retrievability for quality assurance. The change logs include who changed the code, when it was changed, what was changed, and why the change was made. If required, a previous version can be retrieved.

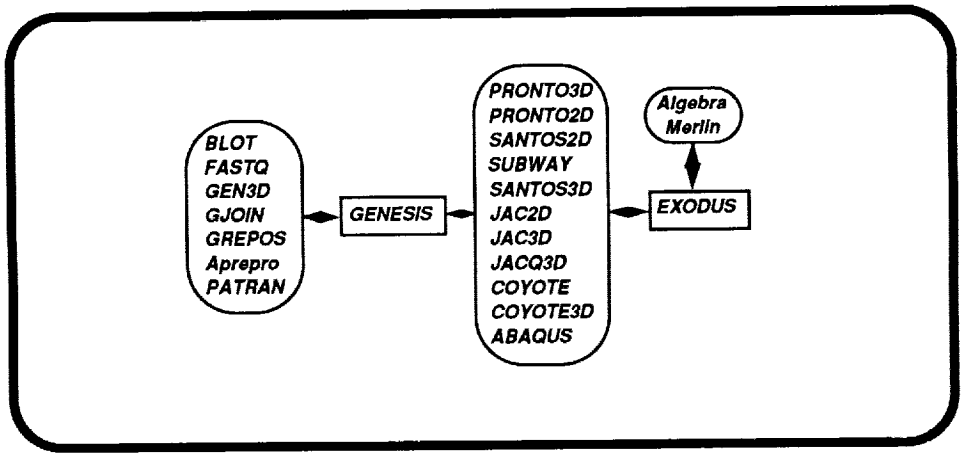

Figure 2 - Code-to-code data transfers using SEACAS

Figure 3 shows an example of a very large, complex problem solved using the codes in SEACAS.

SEACAS is divided into five broad categories of codes: prepost, analysis, translator, libraries, and scripts. The categories can be roughly defined as follows:

prepost: Pre- and postprocessing codes including mesh generation, visualization, preprocessors, and database manipulation codes.

analysis: Finite element analysis codes including quasistatic, transient dynamics, thermal, and electromechanics. Two- and three-dimensional, nonlinear, large deformation.

translators: Translation codes for editing output files (EXODUS), inter-machine translation, and exodus from/to commercial database translation.

libraries: Support libraries including database routines, common machine-specific routines, plot routines, graphics device drivers, and interactive help routines.

scripts: Unix shell scripts for executing the prepost, analysis, and translation codes. Also includes support and installation routines.

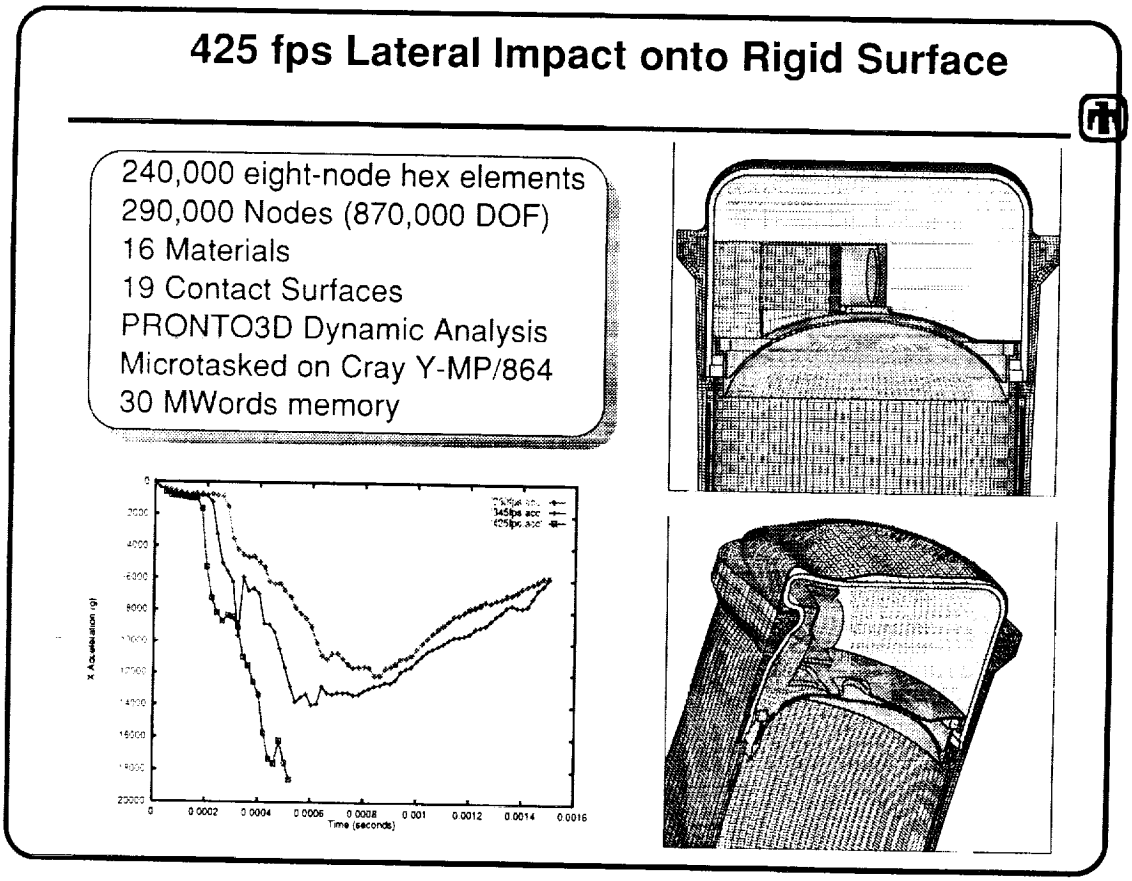

Figure 3 - Example of a complex problem solved using the codes in SEACAS

EXODUS DATA FILE

Figure 4 summarizes the features of the EXODUS data file. An EXODUS data base contains data entities used for input and output of finite element analyses. The features of this data base are:

- self-describing data file -- the file includes information that describes the contained data (dimensionality, size, type, etc.).
- random read (and write) access -- data can be written or read in an arbitrary order.
- machine-independent binary representation -- data is stored in eXternal Data Representation (XDR) format and thus can be transferred to, or accessed from, any machine without being concerned about what machine generated the data.
- FORTRAN and C subroutine interface -- application programs access the data via simple calls to a subroutine library.
- extensible -- because the file is self-describing and not just a rigid file format, new data entities and features can be added without modifying all of the application codes using the data base.
- translators -- capability to translate to/from many commercial application codes (i.e., ABAQUS, PATRAN, etc.).

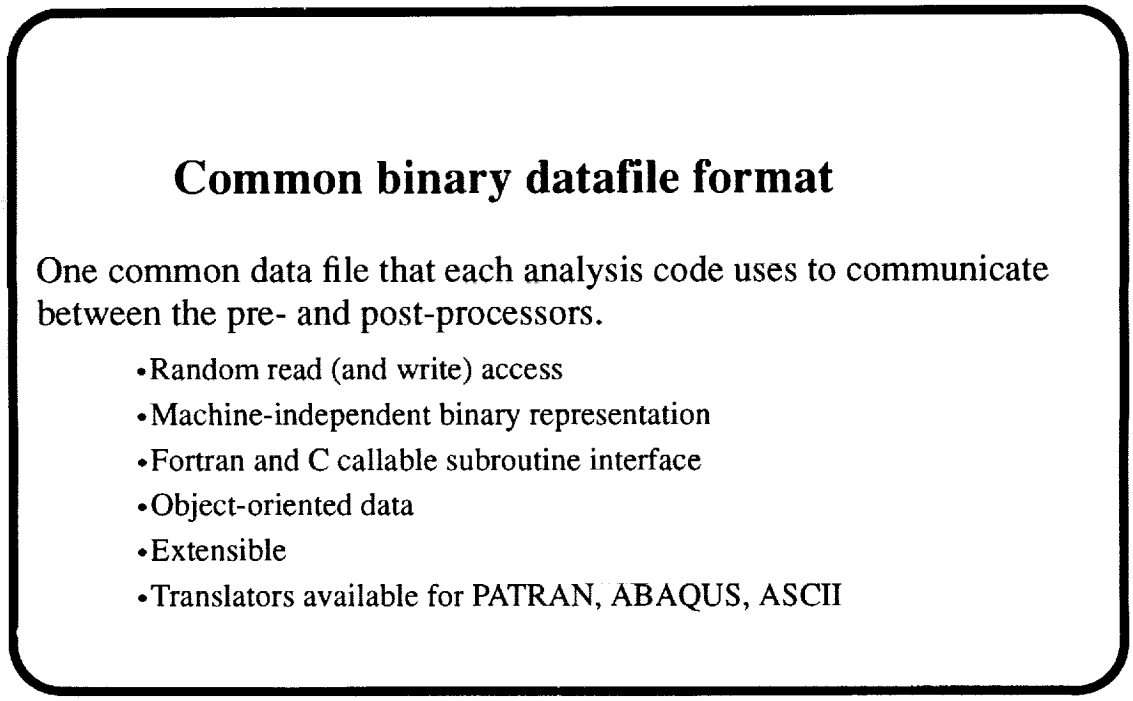

Figure 4 - Summary of Exodus data file

The EXODUS II data model is used for transferring finite element analysis data among application codes. It includes data to define the finite element mesh and label both boundary condition and load application points. EXODUS II accommodates multiple element types and is sufficiently general to service many different analysis codes, providing a very general format for analysis results. The data is stored in a machine-independent format and is randomly accessible. A FORTRAN and C application programming interface is also specified.

Translation codes are used to convert EXODUS databases into different formats and to edit EXODUS databases.

algebra: Manipulates EXODUS finite element output data by evaluating algebraic expressions. Equation variables are dependent on the input database variable names.

seaexo: Converts SEACO files (the binary file format that preceded EXODUS) into EXODUS files.

exotxt and **txtexo:** Converts EXODUS to and from a specially formatted ASCII text file.

exoexo: A program which simply translates an EXODUS file to the same EXODUS file. It is used as a base program for writing new translators or database manipulation programs.

abaexo: Converts ABAQUS (commercial finite element code) results files to EXODUS.

conex: Concatenates several EXODUS files into a single EXODUS file. Used to create a single EXODUS file from analyses that have been restarted.

exoxdr and **xdrexo:** Converts EXODUS to and from an external data representation format (XDR) which can be transmitted between computers of different architectures, word lengths, and byte orders.

exogen: Creates a GENESIS mesh database from a specific time step of an EXODUS file. Used when an analysis of a deformed geometry is required. For example, an impact analysis followed by a thermal analysis.

merlin2: Transfers (maps) nodal data between finite element meshes. For example, thermal output from a heat conduction code to a thermal input file for a quasistatic mechanics code.

exopat and **patexo:** Converts EXODUS to and from PATRAN (commercial mesh generation program) neutral file format.

exo1exo2 and **exo2exo1:** Converts EXODUS I format files to EXODUS II format.

In addition to the translation codes listed above, the Cray Unicos system provides the capability to translate EXODUS files into a VMS and IEEE format using the exoexo code.

PREPROCESSING AND POSTPROCESSING CODES

The pre- and postprocessing codes are comprised of the following mesh generation, visualization and database manipulation codes.

gjoin: Join together two or more GENESIS databases into a single GENESIS database.

gen3d: Transform a two-dimensional GENESIS database into a three-dimensional GENESIS database. Several transformations are supported and additional transformations can be easily added.

grepos: Reposition or scale a GENESIS database.

fastq: Interactive two-dimensional finite element mesh generation program. Includes several mesh generation options including paving.

aprepro: An algebraic preprocessing program.

genshell: Transform a two-dimensional GENESIS database into a three-dimensional shell GENESIS database. Several transformations are supported and additional transformations can be easily added.

numbers: Calculates several properties of an EXODUS file, including mass properties, timesteps, condition numbers, cavity volumes, and others.

grope: Interactively examine an EXODUS database. Grope is also contained within the blot program. Grope is primarily used to validate EXODUS files.

blot: The primary graphical two-dimensional and three-dimensional postprocessing code. It includes deformed mesh plots, contour plots, shaded fringe plots, variable versus variable and time history plots, and distance versus variable plots.

Figure 5 summarizes the capability the new pre-processing tool APREPRO gives a user. We have found that when APREPRO is combined with the other pre-processing tools, a great increase in productivity can be obtained. APREPRO can also be used to convert data in a variety of different units to one set of units.

APREPRO: Algebraic Preprocessor

- Define problem in terms of variables
- Global variables in one file that is *included* in all other files
- Trigonometric and algebraic functions -- C-like syntax
- Portable: versions available on Unicos/VMS/Ultrix/Unix/Amiga

Easily extendable (we wrote, have source)

INPUT

```
{include(one.inc)}
point 1 {x1 = rad*cosd(angle)} {y1=rad}
point 2 {x2 = x1 + rad} {y1 = y1 + 1/2}

{units(inch-lbf-sec)}
point 3 {5*m} {2.54*cm}
point 4 {1*ft} {1*ft + 1*in}
velocity = {1*mile/hour}
```

OUTPUT

```
$ Aprepro (version #) date
$ include file ONE.INC
$ rad = 1.
$ angle = 30.
point 1 0.866025 1.0
point 2 1.866025 1.5

$ units: inch-lbf-sec
point 3 196.85039 1.0
point 4 12.0 13.0
velocity = 17.6
```

- Any expression inside { } will be evaluated and printed
- Can be used with any file that does not contain { } for other reasons

Figure 5 - Aprepro: An algebraic preprocessor

MESH GENERATION

Figure 6 lists the current areas of active research on mesh generation at Sandia. The first goal of the CUBIT mesh generation project is to develop advanced meshing algorithms for meshing arbitrary three-dimensional surfaces and volumes. The paving algorithm was developed as a 2D all-quadrilateral meshing algorithm. It was recently extended to meshing general 3D trimmed surfaces. The plastering algorithm is being developed as a 3D all-hexahedral volume meshing algorithm.

The development of all of the 3D meshing algorithms cannot progress rapidly without the inclusion of a general geometry generation mechanism. A solid modeling package, ACIS, is being used for this purpose. A nonmanifold topology and meshing database was designed and coded. This database serves all the mesh generation and geometry generation tools. The database allows direct access to modeling functionality and should prove an extremely useful prototype for commercial development of a true solid-model-based meshing program.

The second goal is to develop for the first time an extremely robust adaptive finite element analysis capability for use in all fields of mechanics. The paving and plastering algorithms are very well suited for use in adaptive solution algorithms.

The mesh generation algorithms for paving and plastering are superior to any mesh generation algorithms currently available to private industry. Consequently, several software houses have entered into Cooperative Research and Development Agreements (CRADA) with Sandia.

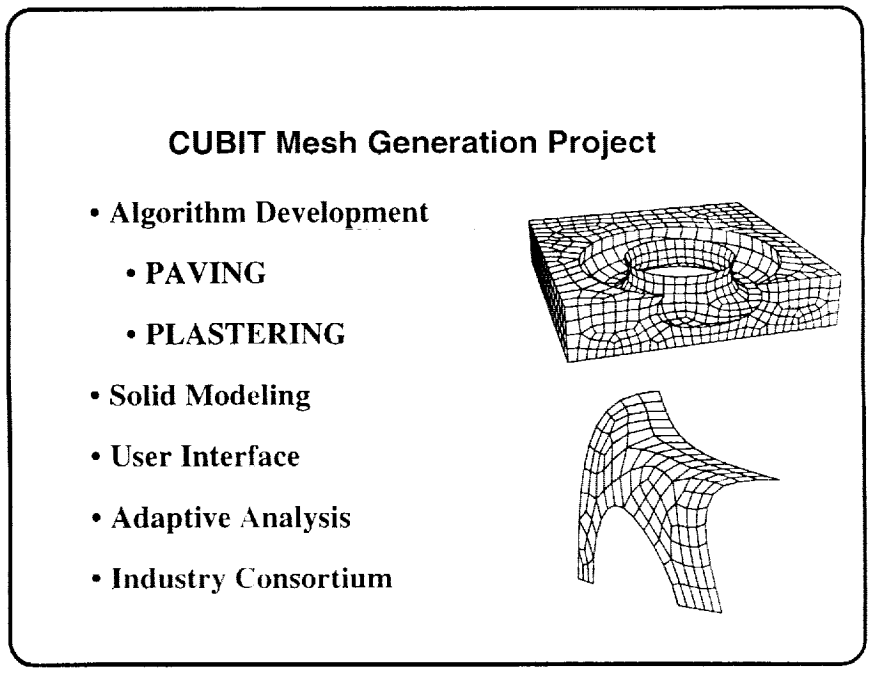

Figure 6 - Current areas of active research on mesh generation at Sandia

Paving Technique

Figure 7 shows examples of the paving technique. The paving algorithm fills an area by first placing elements along the interior and exterior boundaries. Row endpoints are determined so that elements can be fitted into and around corners. Once the row is generated, it is smoothed. The new row can be adjusted based on curvature. Elements called wedges are inserted into rows that are being generated concave outward to keep element sizes from becoming larger. Tucks or the removal of elements are performed for rows that are concave inward to keep elements from becoming smaller. When two rows of elements overlap, the overlap is sewn together by being pushed apart and attached together. When two rows are close but not touching, they are pulled together with a process called seaming cracks. The meshing process produces nearly square elements with corners of approximately 90 degrees. Smoothing and deletions of elements are performed until each element passes a "goodness" test.

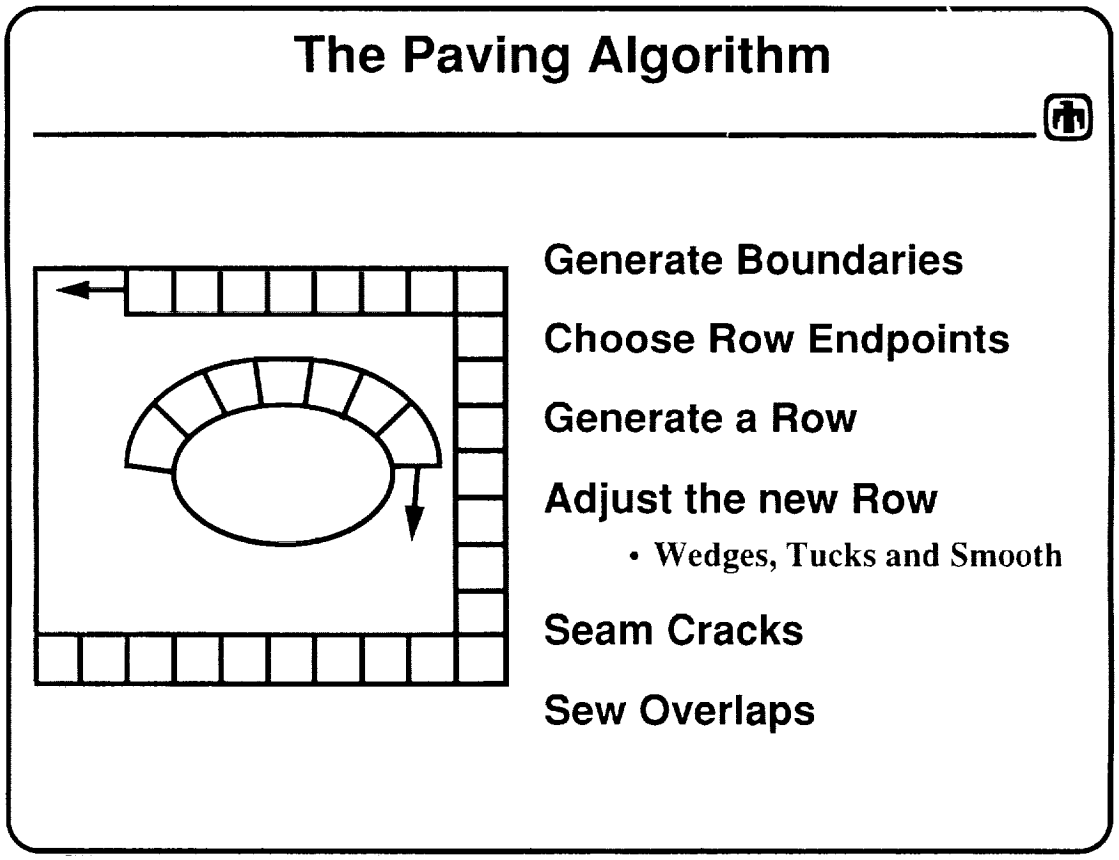

Figure 7 - Examples of the paving technique

As an example, one of the meshing tools is the sewing process. First intersections are found and then candidates for connection are chosen. Then the connections are formed between nodes that are close together. Seams are then formed by pushing rows of element edges together to remove the overlaps. Seams are then formed by pushing rows of element edges together to remove the overlaps. The mesh is then smoothed and elements not meeting a set of "goodness" tests are deleted. Figure 8 shows how sewing can be used to mesh a part with multiple holes.

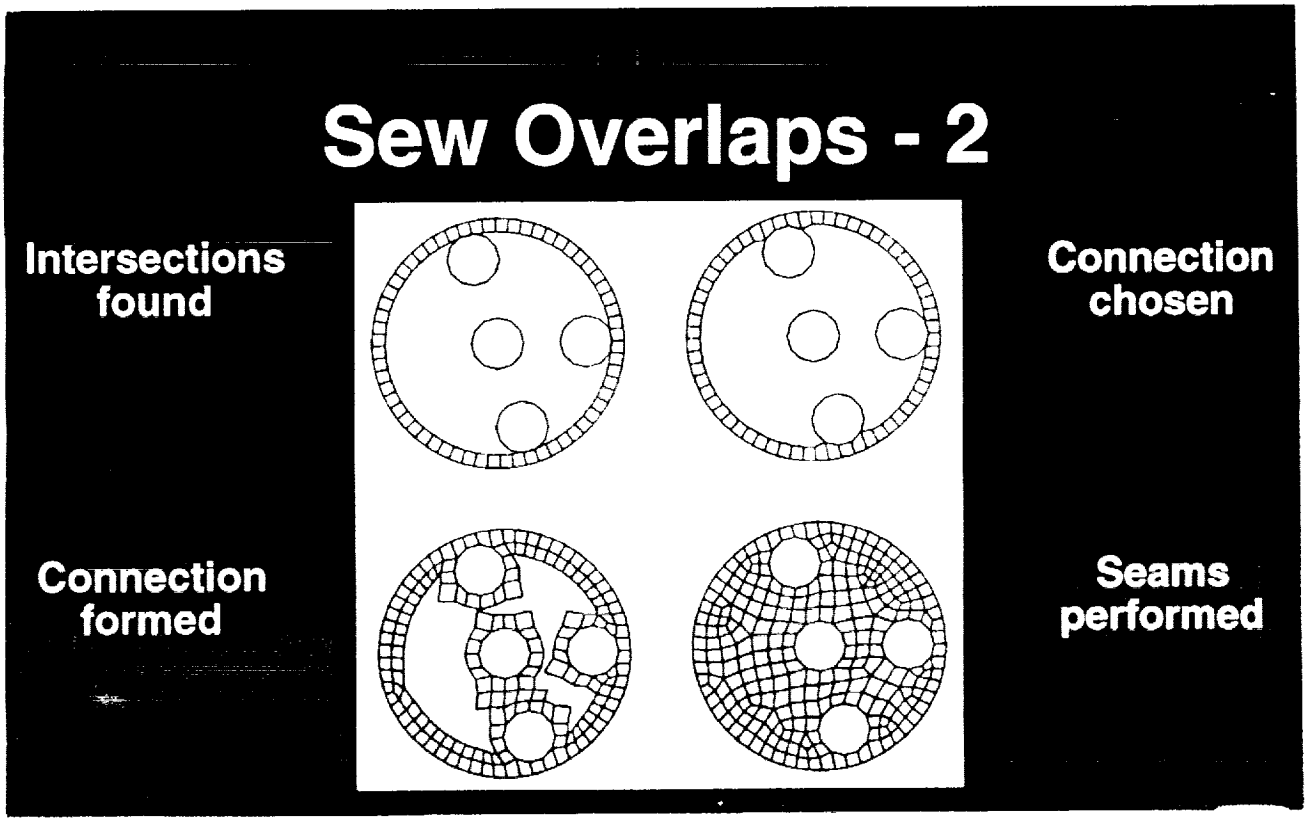

Figure 8 - Example of sewing process used in paving

Plastering Algorithm

Figure 9 shows an example of a plastered volume. The plastering algorithm will be capable of filling an arbitrary three-dimensional solid with an all-hexahedral mesh. The algorithm will first mesh the exterior and interior surfaces with the surface paving algorithm. Then the volume will be filled by projecting the surface elements toward the interior to form elements. Once this is done to all the surfaces, this will define a new volume to be meshed. The process is continued until the volume is filled. Concepts analogous to sewing, seaming, wedges and tucks for surface paving will be used to place elements on the interior of surfaces. This is an example test case for a transition from a 2×2 mesh to a 4×4 mesh with nonregular meshing of the other sides.

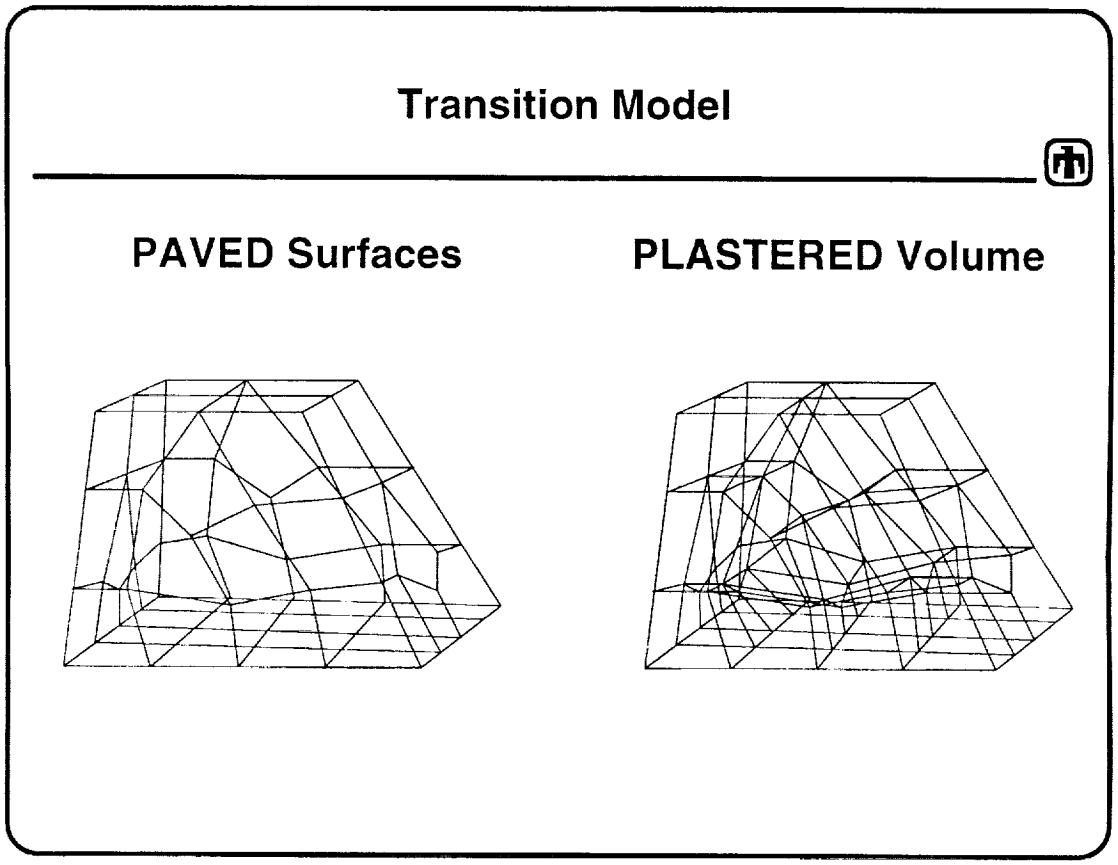

Figure 9 - Example of a plastered volume

Figure 10 shows 3D paving of a NURBS surface (nonuniform rational B-spline). The development of the surface paving algorithm has progressed to being capable of paving NURBS surfaces. This is an example of the surfaces being defined by a solid model and the paving algorithm meshing the surfaces. The example demonstrates the ability to pave arbitrary non-planar surfaces that have complex intersections with other surfaces.

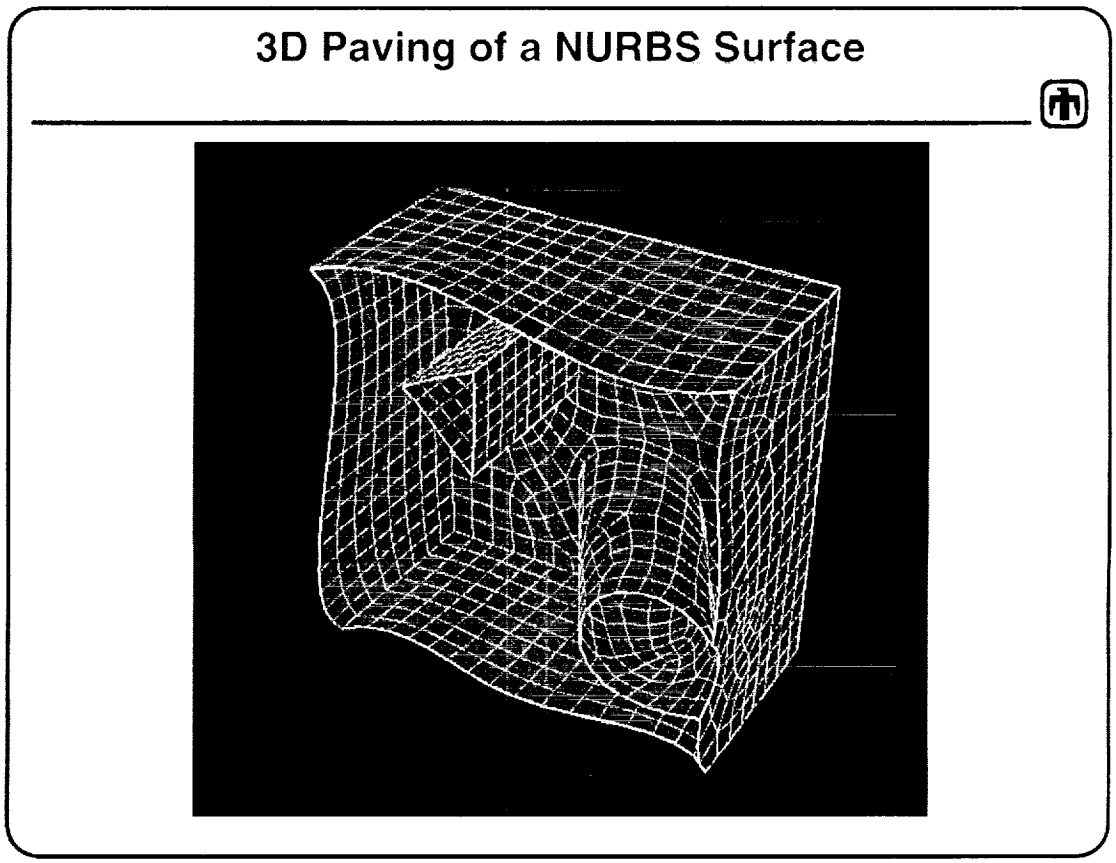

Figure 10 - Three-dimensional paving of a NURBS surface

Adaptive Analysis

The paving algorithm has been used in a proof of concept example to perform an adaptive analysis. The geometry and loads for the problem are first defined. Then an initial mesh of approximately equal size elements is generated with the paving algorithm, and a static analysis is performed. The resulting stresses are used to determine an error function at the nodes of the initial mesh. The surface is then remeshed with the paving algorithm by sizing and placing elements based on an error function. This results in smaller elements where the error is large. This process was repeated four times and the final mesh produces an accurate answer to the problem. An example of adaptive meshing using paving is shown in Fig. 11.

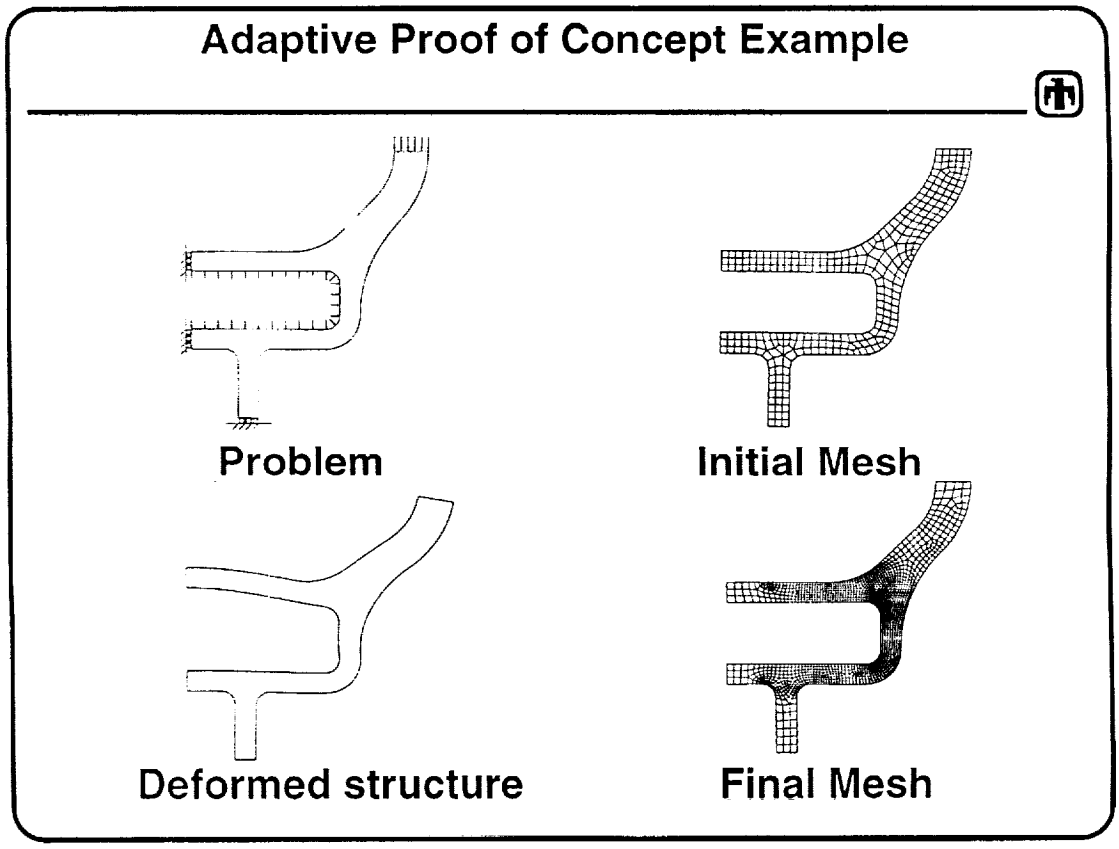

Figure 11 - Adaptive proof of concept example

Mesh Generation Consortium

Sandia has created a Mesh Generation Consortium of computer-aided software companies and Sandia National Laboratories to develop and commercialize the meshing algorithms. To date, ten companies have shown active interest and Cooperative Research and Development Agreements (CRADAs) have been completed. The work is supported with industry funds and matching Department of Energy research funds. The paving algorithm will appear in commercial software within three months. Figure 12 lists the important points of the mesh generation consortium.

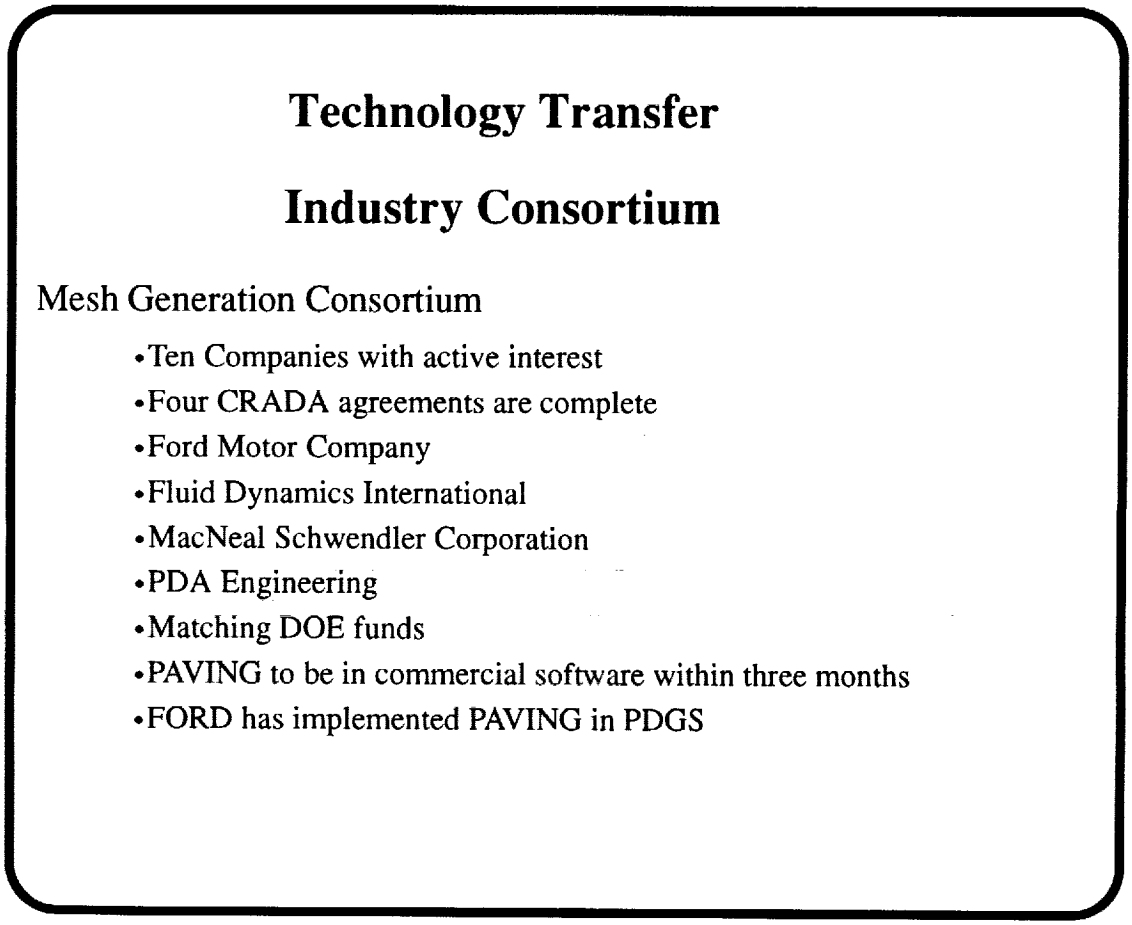

Figure 12 - Mesh generation consortium

PRODUCTION VISUALIZATION ENVIRONMENT

Applied Visualization Group

Supercomputers, both traditional and massively parallel machines, are capable of generating enormous amounts of data. Scientific visualization techniques are necessary to obtain useful information from these massive amounts of data. These techniques include responsive manipulation and animation of three-dimensional objects.

Resulting databases can be on the order of several gigabytes, so high speed access to these large remote databases is required. Another challenge is that analysts want to perform these visualization activities at their desks.

An **Applied Visualization Group (AVG)** has been established at Sandia National Laboratories to design and implement a production scientific visualization environment for use by the staff in the engineering sciences discipline. These analysts perform finite element and finite difference calculations in the areas of high velocity impact physics, shock wave physics, structural dynamics, structural mechanics, thermodynamics, and fluid mechanics. Visual results are critical in the data analysis functions performed.

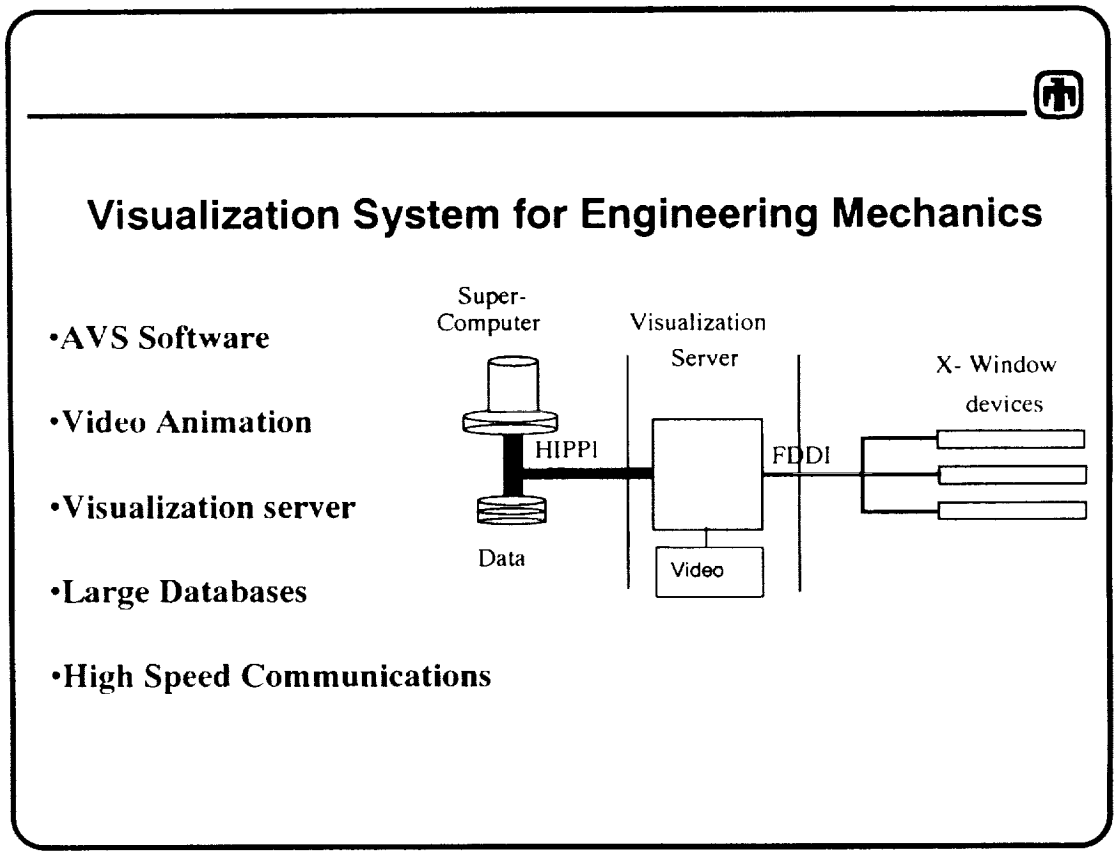

Figure 13 - Visualization system for engineering mechanics

Visualization Server

A visualization server concept is being implemented. One or several very high powered graphics machines will be used to perform the graphics manipulations, with the resulting images transmitted to an X-window display on the analyst's desk. We also expect to use PEX to distribute 3D graphics between the visualization server and the desktop display. We anticipate having a 100MByte channel connection from the visualization server to a supercomputer and network storage system, with FDDI connections from the visualization server to the offices. A block diagram of the proposed production environment consisting of hardware, software, and output capabilities integrated in an easy to use fashion is shown in Fig. 13. (See previous page.)

A video animation system, using a component video laser disk recorder, high resolution converter, video transcoder, and a workstation, allows workers to rapidly create videos of analyses. The use of component and composite video technology allows for high quality when results are played directly from the laser disc, while still allowing for VHS video tapes to be created for showing at remote locations. A block diagram of the system currently in use at Sandia is shown in Fig. 14.

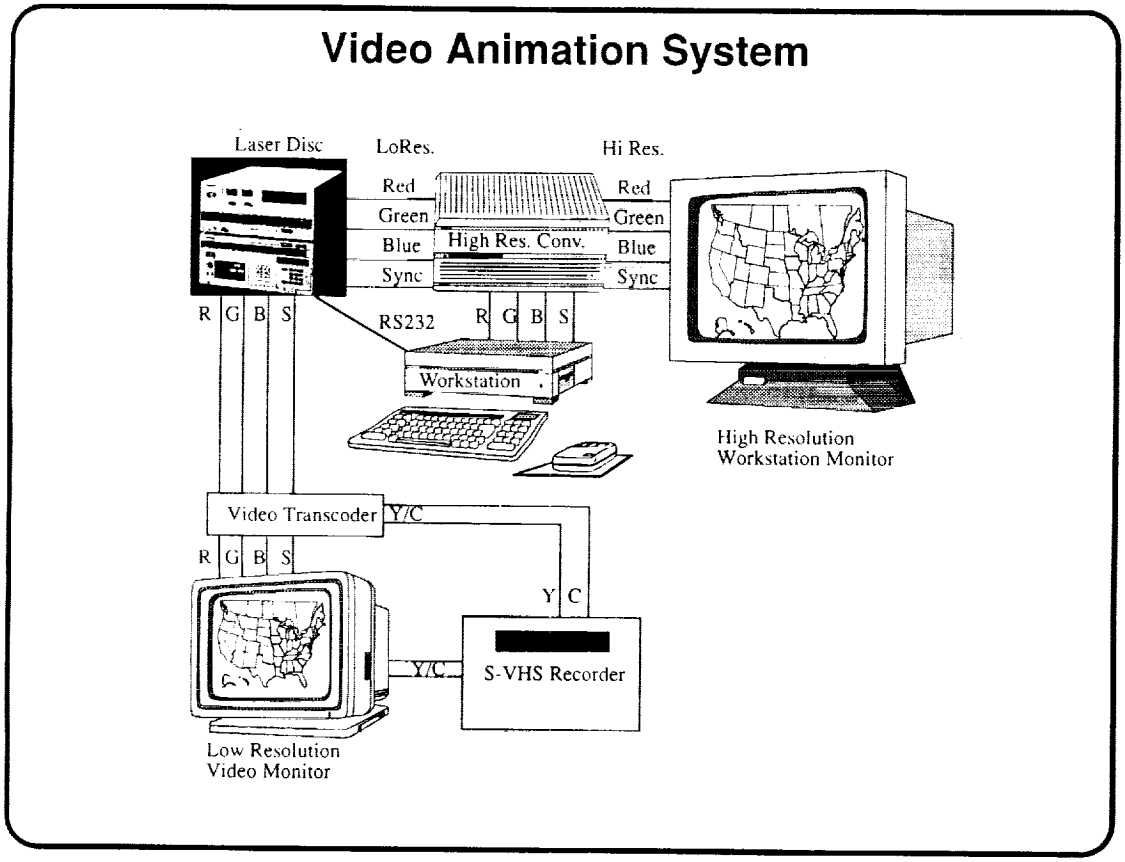

Figure 14 - Video animation system at Sandia

Visualization Software

Figure 15 shows a typical screen of what an analyst might expect to see while working at his desk. Application Visualization System (AVS), a commercial scientific visualization environment, was selected as the foundation for our software development effort after an extensive survey of the market two years ago. It was chosen because of the following characteristics:

- functional -- contains a full-featured suite of modules which operate on numerous data types, including two-dimensional images and scalar and vector fields associated with structured or unstructured grids.
- extensible -- due to its modularity, functions can be readily added/customized.
- ubiquitous -- available on wide variety of hardware platforms.
- distributable -- AVS modules can execute on different machines within an application.

Figure 15 - Example of the AVS visualization tool

ANALYSIS PROGRAMS

The analysis codes in SEACAS include quasistatic, transient dynamics, thermal, and electromechanics codes.

jac2d: Jac2d is a finite element program which uses a nonlinear conjugate gradient technique to solve the large displacement, large strain, temperature-dependent and material nonlinear problem for two-dimensional plane or axisymmetric solids.

jac3d: Jac3d is a finite element program which uses a nonlinear conjugate gradient technique to solve the large displacement, large strain, temperature-dependent and material nonlinear problem for three-dimensional solids.

jacq3d: Jacq3d is a finite element program which uses a nonlinear conjugate gradient technique to solve the thermal conduction problem for three-dimensional solids.

pronto2d: Pronto2d is a two-dimensional (planar and axisymmetric) transient solid dynamics code. Lagrangian formulation with explicit time integration is used for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates.

pronto3d: Pronto3d is a three-dimensional transient solid dynamics code. Lagrangian formulation with explicit time integration is used for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates.

sancho: Sancho is a finite element program which uses a dynamic relaxation technique to compute the quasistatic, large deformation, temperature-dependent, inelastic response of planar or axisymmetric solids.

santos: Santos is a finite element program which uses a dynamic relaxation technique to compute the quasistatic, large deformation, temperature-dependent, inelastic response of planar or axisymmetric solids. Newer than sancho.

santos3d: Santos3d is a three-dimensional version of santos. It is currently in development.

conchas: Conchas is a linear structural analysis code for axisymmetric structures with loads that are symmetric about a plane.

subway: Subway is a three-dimensional finite element program for calculating the transient electro-mechanical response of dielectric materials.

Constitutive Models

Many constitutive models are implemented in the analysis codes.

Elastic: Typical linear elastic material using Hooke's Law.

Elastic-Plastic with Combined Hardening: Standard von Mises type yield condition with combined kinematic and isotropic linear hardening.

Viscoplastic: Simple rate-dependent plasticity.

Viscoelastic: Thermoviscoelastic model for glass solidification.

Damage: Dynamic fracture of brittle rock.

Soils and Crushable Foam: Volumetric plasticity model.

Low Density Foam: Low density polyurethane foam behavior.

Hydrodynamic: Used with Equations of State in PRONTO

Elastic-Plastic Hydrodynamic: Combination of elastic-plastic combined hardening with hydrodynamic pressure response: Mie_Gruneisen type equation of state, JWL High Explosive Equation of State.

Rate and Temperature Dependent Plasticity: Unified creep plasticity.

Secondary Creep: Power hardening steady-state creep with elastic bulk response. Isotropic Elastic/Plastic Power-Law Hardening with Luders Strain.

Johnson-Cook Strength: Rate and temperature-dependent plasticity.

Elastic Plastic Power Law Hardening with Luder's Strain: Describes post-yield strain hardening by a power law equation and includes a yield plateau.

Hyperelastic: Stress based on the principal strains.

Salt Consolidation: Power hardening steady-state creep deviatoric response with consolidation bulk response.

Not all of the constitutive models are implemented in all of the analysis codes. The quasistatic analysis codes typically implement the constitutive models using temperature-dependent properties to facilitate thermal-stress calculations.

TRANSIENT DYNAMICS DEVELOPMENT

A summary of the transient dynamics code PRONTO and the development objectives for PRONTO are shown in Fig. 16.

PRONTO Description:

A two- and three-dimensional transient solid dynamics code for analyzing large deformations of highly non-linear materials subjected to high strain rates

- Explicit mid-point time integration
- One point element integration
- 2D - four node quad
- 3D - eight node hexagon, 3D - four node shell
- Hourglass control
- Adaptive time step control
- Symmetric contact algorithm
- Objective material coordinate system

Code Development Objectives:

- Well documented and consistent
- Accurate numerical algorithms which minimize error
- Dependable and aborts executions by providing diagnostics
- Executes rapidly
- Uses existing pre- and post- processing software
- Easy to add new constitutive models

Figure 16 - PRONTO description and development objectives

New Contact Detection Algorithm

An increasingly important aspect of large-scale finite element simulations is the efficient and accurate determination of contact between deformable bodies. Generally there are several aspects of the numerical model that make contact detection difficult. This section reviews some currently used contact detection techniques and outlines some difficulties associated with these algorithms. Then, a new algorithm is proposed which circumvents these difficulties.

The key points of the new contact detection algorithm are: i) uses the existing contact penalty/kinematic constraint method (including partitioned contact); ii) easily models self-contacting surfaces; iii) automatically defines all surfaces given the mesh connectivity; iv) models tearing and eroding surfaces; v) reduces user input; vi) uses a fast, memory-efficient global search to decide what slave nodes are in proximity to a master surface; and vii) does a detailed contact check using projected movements of both the master and slave to determine the magnitude and direction of slave node penetration of the master surface. See Fig. 17.

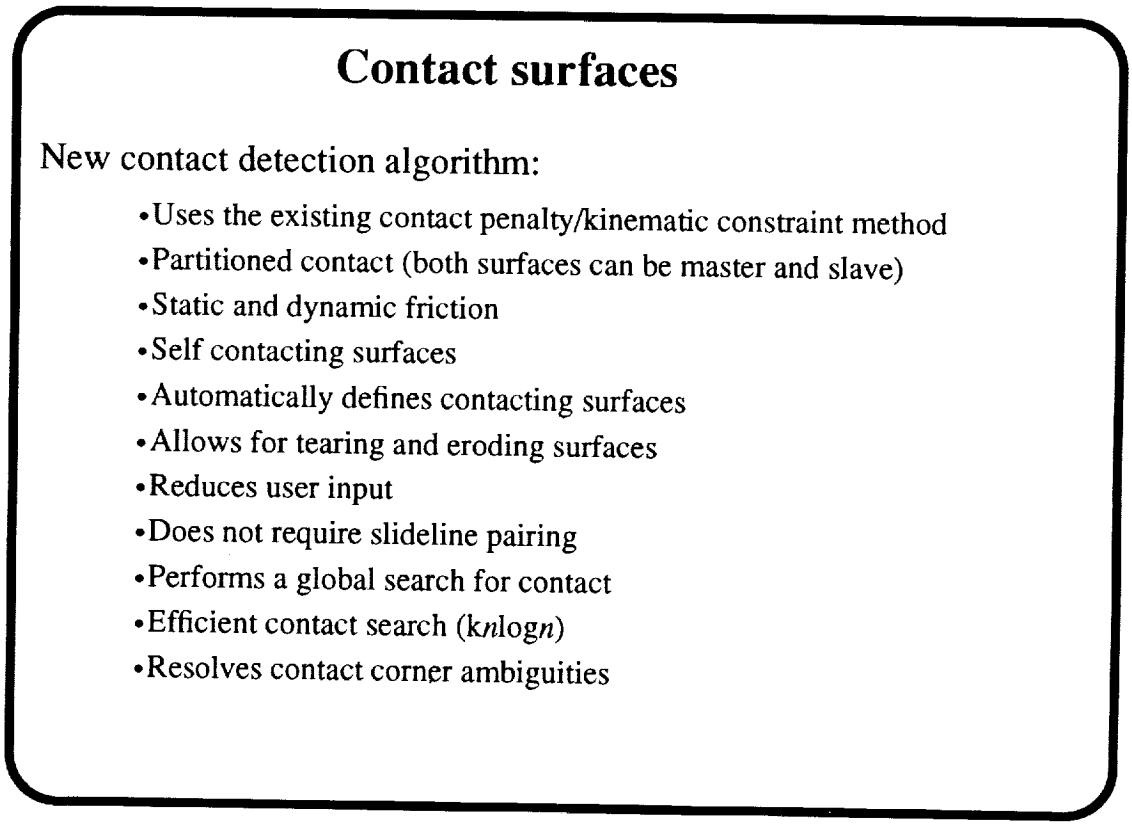

Figure 17 - Contact surfaces

Multibody Impact: Elastic-Plastic Bar Impacting Bricks

The following example demonstrates two features of the contact detection algorithm that are important to this particular class of problems. The example shown in Fig. 18 has an elastic-plastic bar impacting a stack of 17 elastic bricks. A stationary elastic-plastic wall is also resting against the stack of bricks.

An important feature of the contact detection algorithm is the spatial independence of the kn Log$n$ vectorized sorting algorithm. The speed of the algorithm is independent on the location of the bricks. This becomes particularly important late in this example since the volume that the bodies occupy is increasing.

Another feature of the contact detection algorithm that is both more efficient and useful is that the surfaces are not required to be paired. In fact, the surfaces were automatically defined using a new surface definition algorithm. For problems where random contact is anticipated, as in this example, each body could potentially impact any other body. For a contact-pairing algorithm, this implies that n_b^2 contact pairs would be necessary, with each pair having 2m slave nodes. For the current global contact searching algorithm, it would imply one search with $n_b m$ slave nodes. Consider in this example 19^2 pairs would have to be defined since random contact is expected. Assuming that each block has $m \approx 50$ slave nodes, 19^2 pairs would require $19^2(2m log(2m)) = 239{,}843$ comparisons, whereas the current global contact searching algorithm would require only $19m log_2(19m) = 9397$ comparisons.

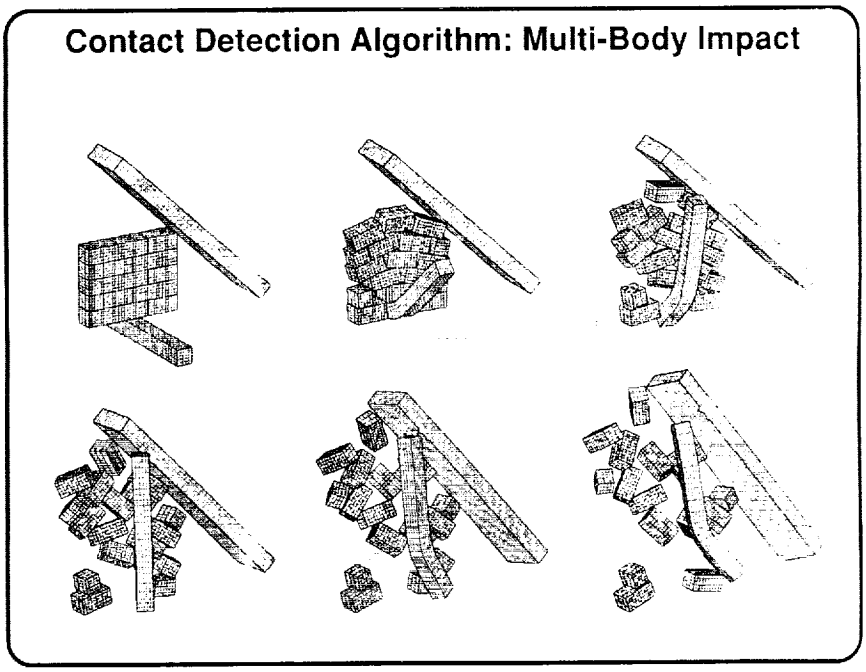

Figure 18 - Elastic-plastic bar impacting bricks

Self-Contacting Impact: Buckling of Shell-Like Structures

The following example demonstrates the self-contacting capability of the contact detection algorithm. This feature is important to modeling crash dynamics where buckling, tearing and self-contact is common. The example shown in Fig. 19 has an elastic plate impacting an elastic-plastic can (shell-like structure). The can is 0.7 inches thick and has an inside radius of 12.5 inches. After 10 milliseconds the can is nearly crushed flat with numerous folds and buckles that self contact.

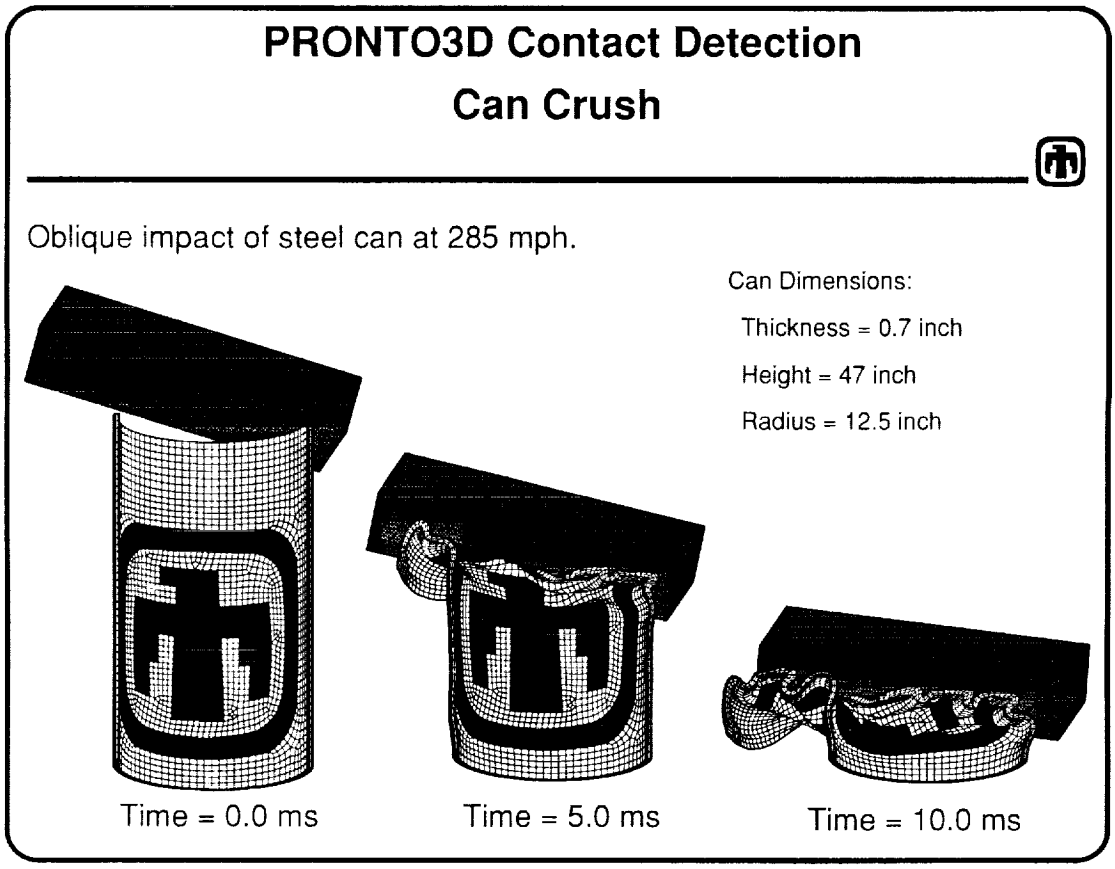

Figure 19 - Self-contacting impact of a buckling shell-like structure

Automatic Contact Surface Redefinition: Tearing/Petalling of a Plate

In the following example, the capability of automatically redefining the contact surface is essential. This feature is also important for modelling crash dynamics where buckling, tearing and self-contact is common. The example shown in Fig. 20 has a solid elastic sphere impacting a thin elastic-plastic plate. The plate is 0.25 inches thick and has an overall dimension of 12.0 inches by 14.0 inches. After a few milliseconds the plate is damaged such that two tears develop orthogonal to one another. The tears are actually a series of deleted elements where the damage has accumulated to 1.0. The newly created surface is automatically included in the contact algorithm by redefining the surface after any elements are deleted. At late time, the edges of the tears are in contact with the elastic sphere.

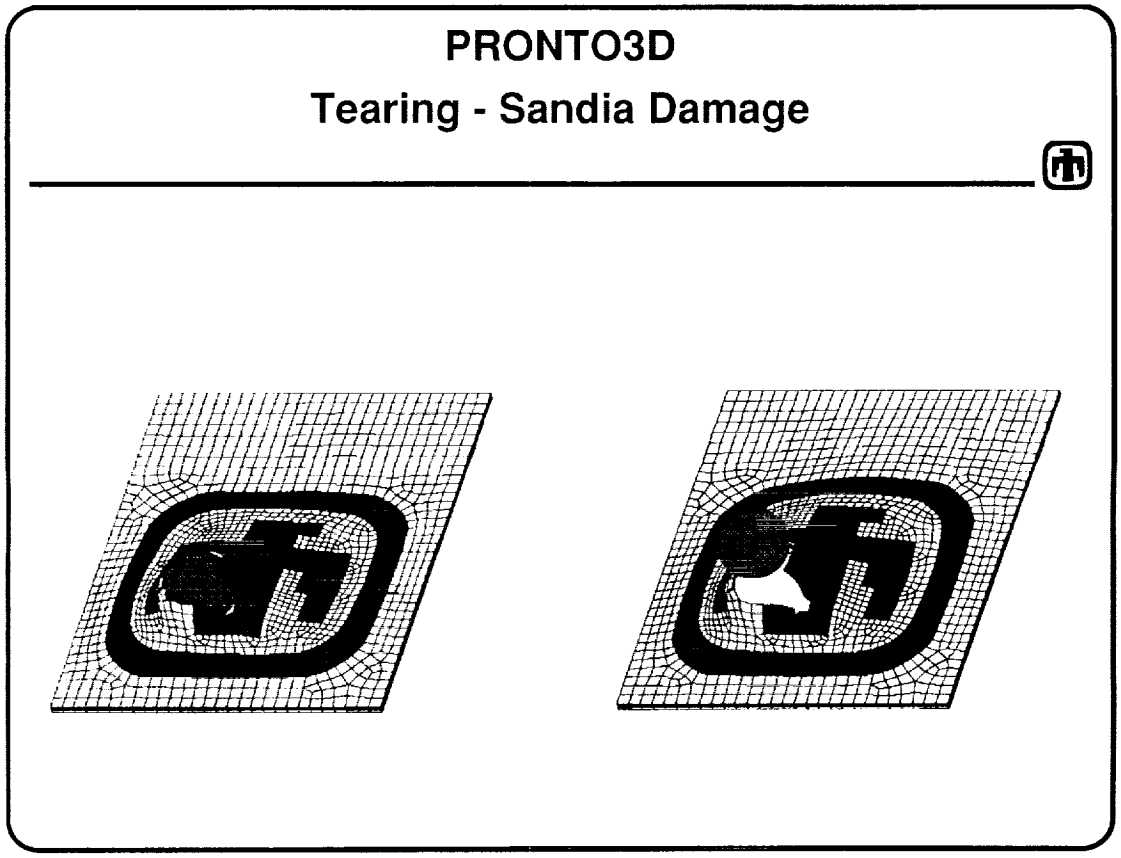

Figure 20 - Automatic contact surface redefinition: tearing/petalling of a plate

Contact Algorithm Implementation: Reduced User Input

Reduced user input is a convenience that is an outcome of the global searching and automatic surface definition capability. One can, for example, selectively include all surfaces of a body composed of a material and/or include only a subset of the surface. Figure 21 shows the typical input required for a problem. Reducing the input required from the user minimizes the number of mistakes, which minimizes cost.

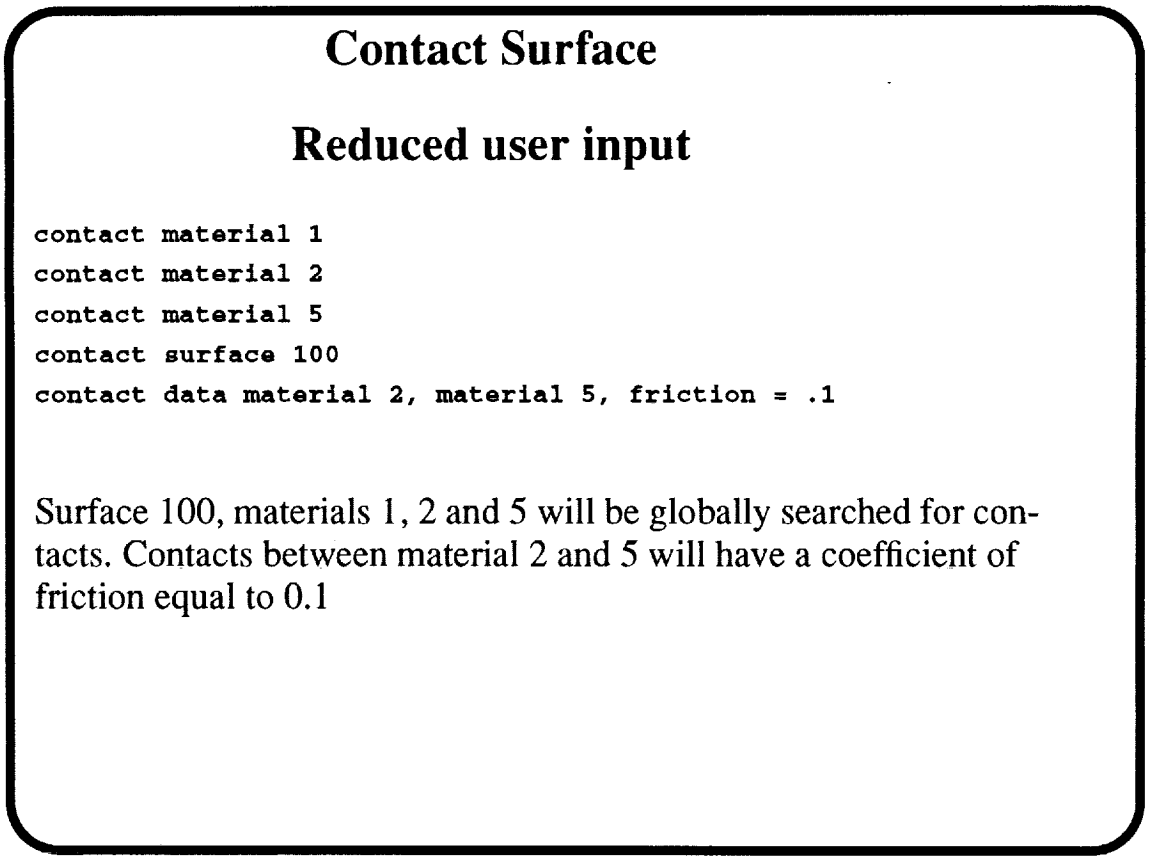

Figure 21 - Reduced user input

Contact Detection Algorithm: Capturing Slave Nodes for a Master Surface

It is widely held that the location phase is the most time-consuming part of the contact detection algorithm. Obviously, the most robust approach would be to check every slave node against every master surface every time step. This exhaustive global searching approach requires nodal distance calculations on the order n^2, where n is the number of nodes. Several algorithms have been proposed to speed up the location phase.

The proposed contact detection algorithm uses a new global search algorithm. It uses 7n memory locations and requires (k nLogn) comparisons and depends on only the number of entities (n) to be compared. It takes advantage of the known positions of the slave nodes and master surfaces as well as the predicted positions in the next time step. After the slave nodes are sorted by x, y and z coordinate, the master surfaces are processed sequentially. This processing involves bounding the space occupied by a master surface at its known location and its predicted location. Figure 22 shows a bounding box for a master surface over one time step. Clearly, a slave node could only make contact with this master surface if it is no further from the bounding box than the distance it could move in the time step. Therefore, any slave node inside the capture box should be considered for potential contact with the master surface.

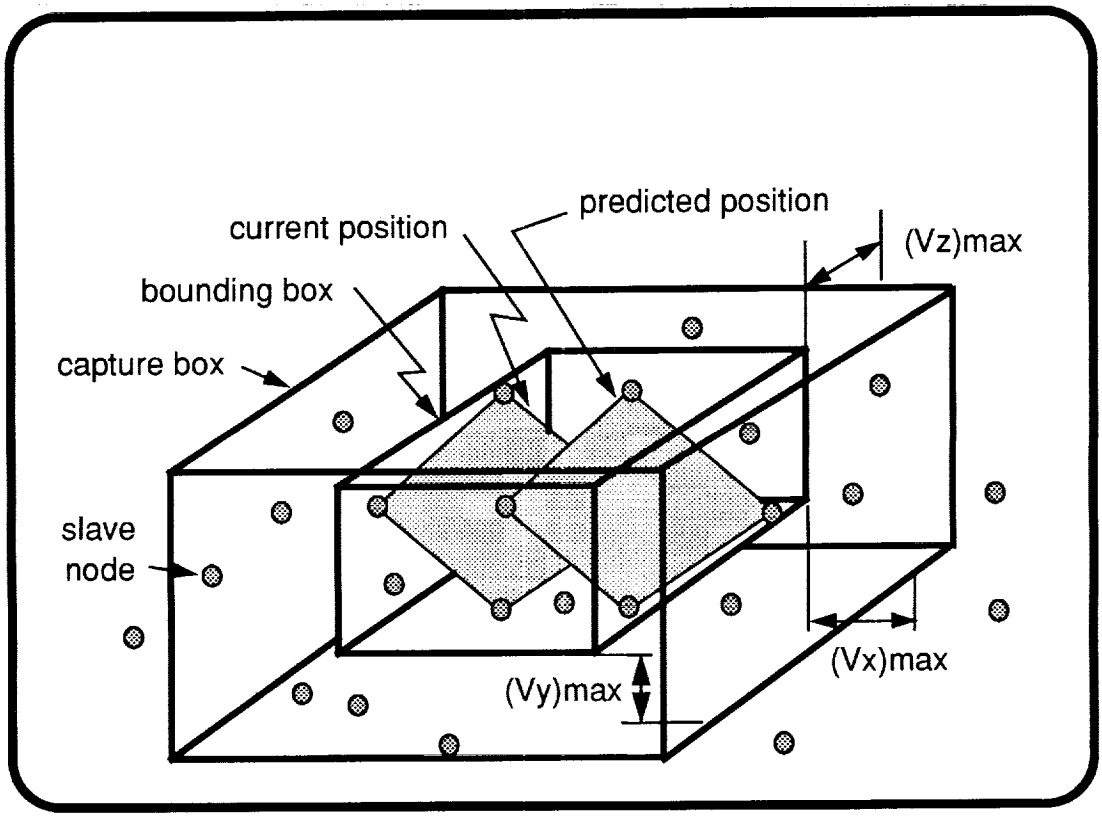

Figure 22 - Contact search bounding box

Contact Detection Algorithm: Slave Node/Master Surface Contact

Unfortunately there are many ambiguous cases when determining contact between a slave node and a master surface. One such ambiguity arises because the surface normal is not continuous. This can result in either missed contacts or multiple solutions. In the proposed algorithm, the motion of the slave node and the master surface are considered and these ambiguities are avoided. As shown in Fig. 23, the time of contact and the point of contact is determined for each possible master surface that a slave node can penetrate. The algorithm solves for the time when a slave node and master surface all lie in the same plane. When the slave node can contact more than one master surface, the minimum time to contact determines the correct master surface contact.

Further ambiguity is introduced when a contact algorithm performs detailed contact checks based only on the estimated (deformed) configuration. If an algorithm does not make use of information about how the slave node deforms to the estimated position, the algorithm must decide which contact violation to enforce. This is commonly done by determining which surface is most opposed to the slave node surface normal. When two surfaces are already in contact this so-called strength of contact check can be effective. However, when the surfaces are just coming into contact this type of static contact check cannot consistently predict the correct contact.

In the proposed algorithm, the motion of the slave node and the master surface are considered and these ambiguities are avoided. In determining the push back direction, a distinction is made between convex and concave surfaces. The push back direction for a convex surface is determined simply by the minimum distance to the master surface. The push back direction can be along the master surface normal or it may be defined by the minimum distance to a vertex. For a concave surface the push back direction is always along the normal of the master surface that the slave node was previously in contact with.

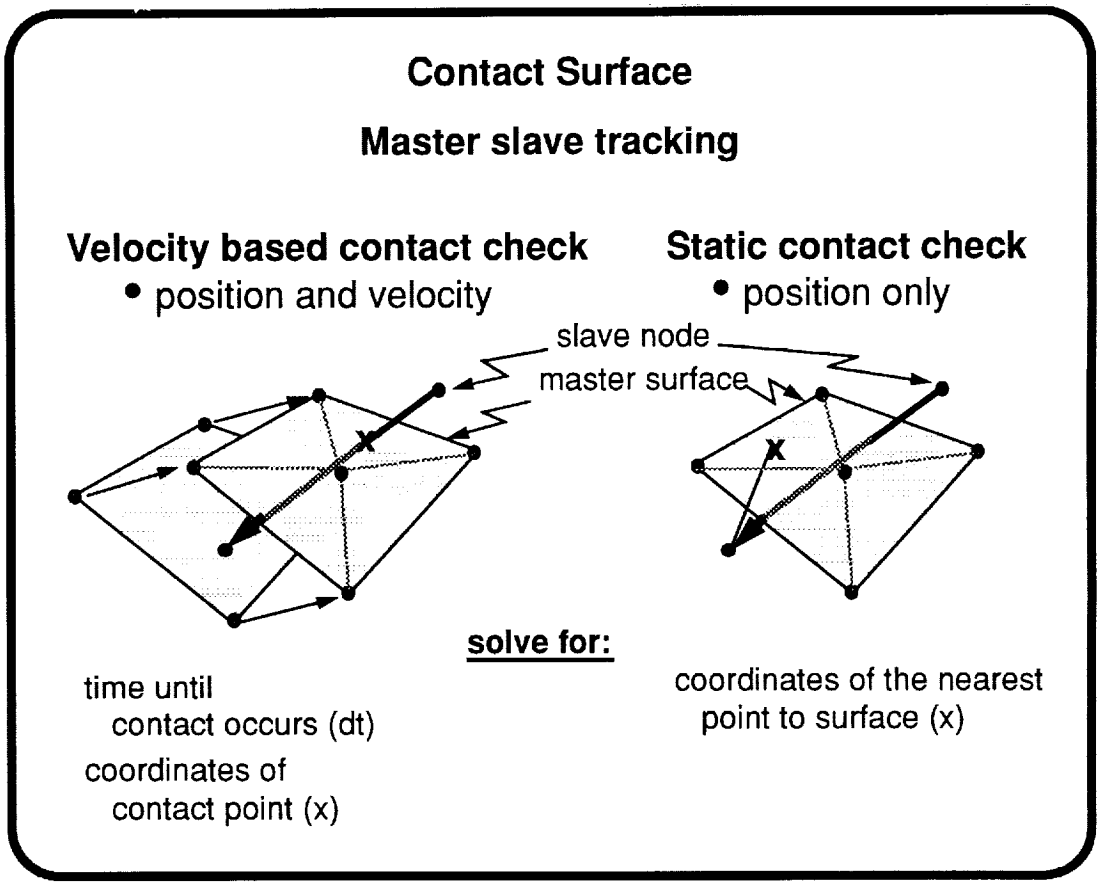

Figure 23 - Master/slave tracking

FUTURE CODE DEVELOPMENT

Figure 24 lists some of the active areas of code development and research at Sandia.

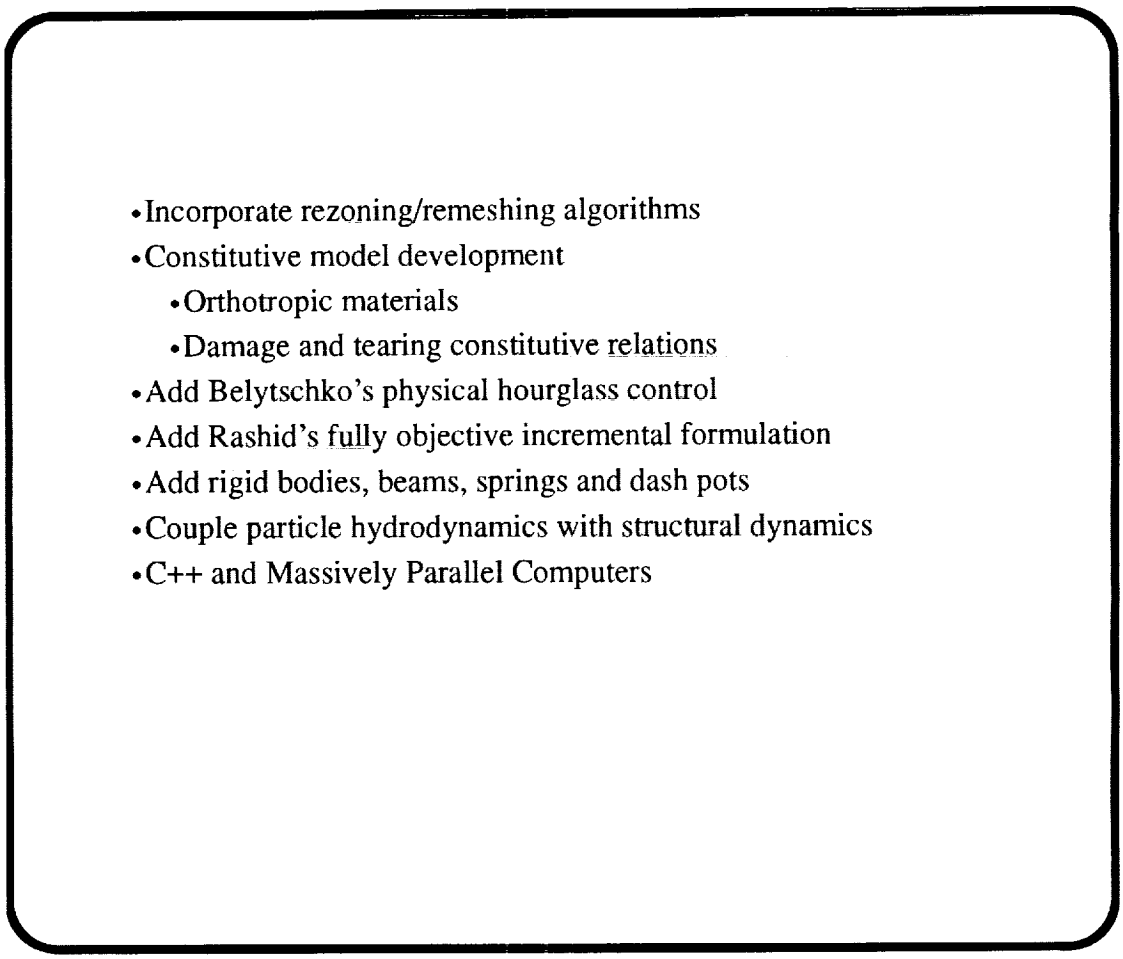

Figure 24 - Future code development

Couple Particle Hydrodynamics with Structural Dynamics

Figure 25 shows an example problem using a combination of finite elements and particle elements. One major difficulty associated with the Lagrangian finite element method is modeling materials with no shear strength; for example, gases, fluids and explosives biproducts. Typically these materials can be modelled for only a short time with a Lagrangian finite element code. Tangling of the mesh will eventually lead to numerical difficulties such as negative element area or "bow tie" elements. Remeshing will allow the problem to continue for a short while, but the lack of shear strength causes instabilities that prevent a complete analysis.

Smooth particle hydrodynamics is a gridless Lagrangian hydrodynamic technique. Requiring no mesh, SPH can model material fracture, large shear flows, and penetration. SPH treats material as particles that have their masses smoothed in space. The density is computed at a point by summing the contributions of the smoothed particle masses in the vicinity of the point. SPH computes the strain rate and the stress divergence based on the nearest neighbors of a particle. The nearest neighbors are determined using an efficient particle sorting technique.

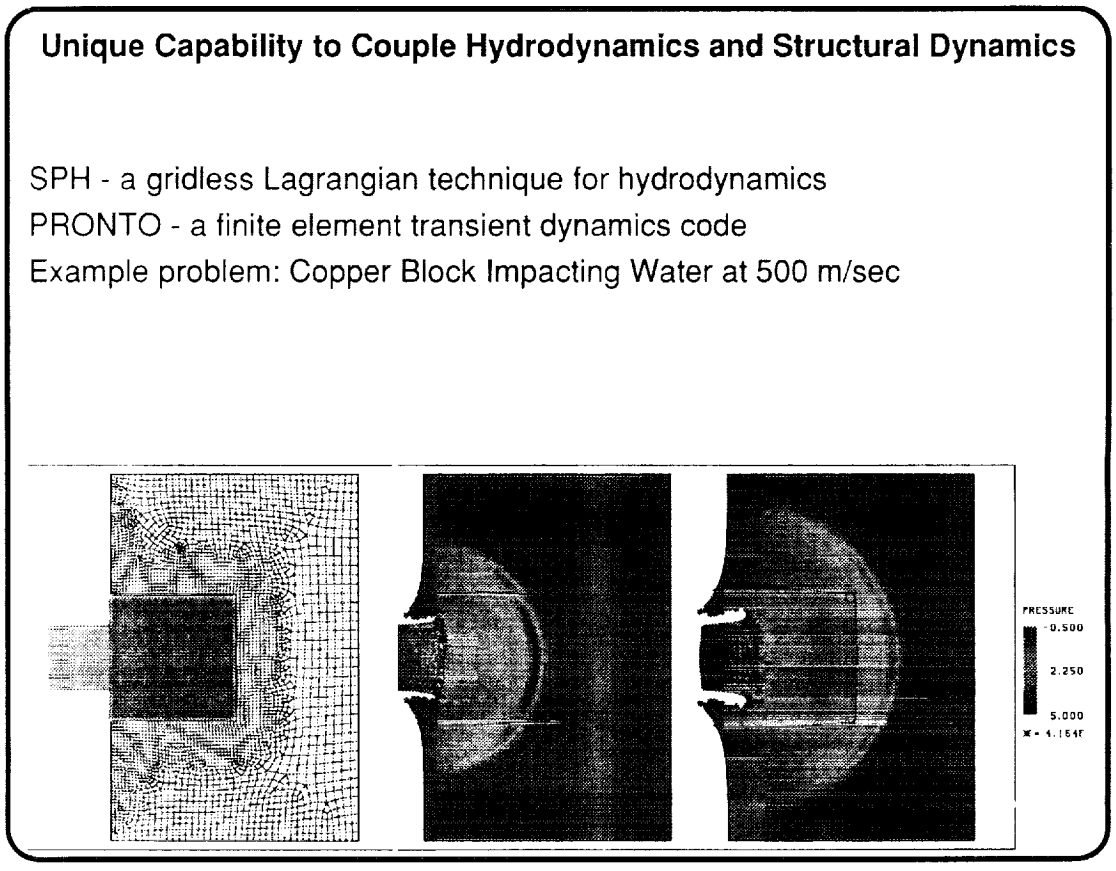

Figure 25 - Coupling particle hydrodynamics and structural dynamics

The SPH computational technique was embedded within the PRONTO computer code. SPH elements are modeled within PRONTO as elements that have only one node. Each element has the typical element variables associated with it (stress, strain, rotation, stretch, density, energy and other state variables). By using the existing PRONTO architecture, the SPH elements used the same constitutive equations as used by the quadrilateral elements in PRONTO.

The embedding of the SPH method within PRONTO allows part of the problem to be modeled with quadrilateral elements while other parts of the problem are modeled with the gridless SPH method. The quadrilateral elements are coupled to the SPH elements through a contact-like algorithm.

Object-Oriented Code

Figure 26 outlines the objectives for a new code that would combine solid, fluid and heat transfer problems into one code architecture. The goal of this effort is to apply object-oriented concepts to the design of a code architecture for solving problems in solid mechanics, fluid mechanics, and heat transfer. Consolidating these areas into a single package has the potential of reducing the time required for code maintenance and adding new features. Such an architecture should facilitate development of the capability to solve strongly coupled problems among these three areas of engineering science. The code will be implemented in the object-oriented language C++ to simplify code reuse and provide for extensibility. The code architecture will be designed to run efficiently on message passing MIMD computers and to take advantage of vector capabilities that exist on some machines in that class.

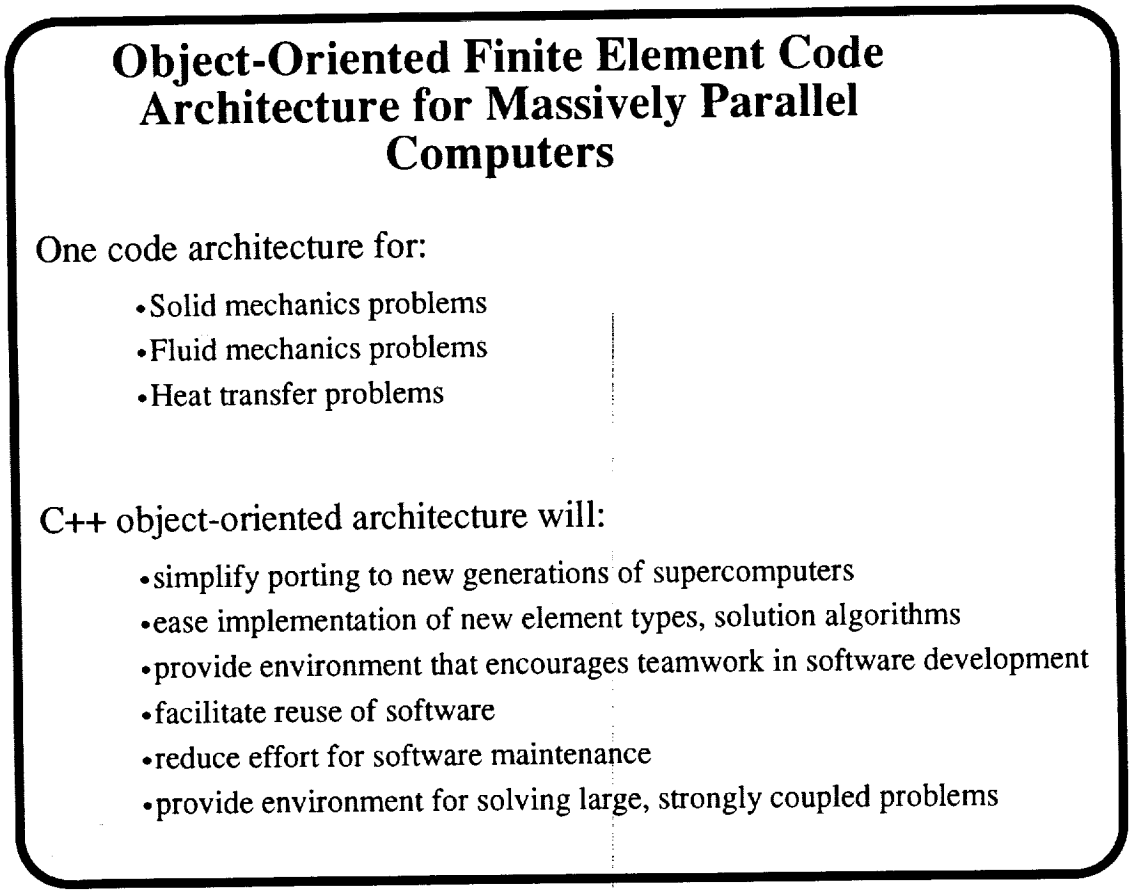

Figure 26 - Object-oriented finite element code architecture for massively parallel computers

ABSTRACTS AND REFERENCES

References and complete abstracts for selected analysis, preprocessing, postprocessing, translation, and library codes are listed in this section. The abstracts and references for translators are not listed.

Algebra

The Algebra program allows the user to manipulate data from a finite element analysis before it is plotted. The finite element output data is in the form of variable values (e.g., stress, strain, and velocity components) in an EXODUS database. The Algebra program evaluates user-supplied functions of the data and writes the results to an output EXODUS database which can be read by plot programs.

Amy P. Gilkey, "ALGEBRA - A Program that Algebraically Manipulates the Output of a Finite Element Analysis (EXODUS Version)," SAND88-1431, Sandia National Laboratories, Albuquerque, New Mexico, August 1988.

Blot

Blot is a graphics program for postprocessing of finite element analyses output in the EXODUS database format. It is command driven with free-format input and can drive any graphics device supported by the Sandia Virtual Device Interface.

Blot produces mesh plots with various representations of the analysis output variables. The major mesh plot capabilities are deformed mesh plots, line contours, filled (painted) contours, vector plots of two/three variables (e.g., velocity vectors), and symbol plots of scalar variables (e.g., discrete cracks). Pathlines of analysis variables can also be drawn on the mesh. Blot's features include element selection by material, element birth and death, multiple views for combining several displays on each plot, symmetry mirroring, and node and element numbering.

Blot can also produce X-Y curve plots of the analysis variables. Blot generates time-versus-variable plots or variable-versus-variable plots. It also generates distance-versus-variable plots at selected time steps where the distance is the accumulated distance between pairs of nodes or element centers.

Amy P. Gilkey and John H. Glick, "BLOT-A Mesh and Curve Plot Program for the Output of a Finite Element Analysis," SAND88-1432, Sandia National Laboratories, Albuquerque, New Mexico, August 1991.

R. J. Myers, "Updates to the Postprocessing Program BLOT," memo to distribution dated August 21, 1990, Sandia National Laboratories, Albuquerque, New Mexico.

Conchas

CONCHAS is a linear finite element structural analysis code which is specialized for axisymmetric structures. Loads and responses are limited to those symmetric about a plane which includes the symmetric axis of the structure. CONCHAS will perform eigenanalysis, static analysis, and dynamic analysis. The element library includes consistent-mass shell, solid, and beam elements, nonlinear springs, and concentrated masses. Pre- and postprocessing is available with the separately supported utilities BLOT, PATRAN and FASTQ.

William C. Mills-Curran and Dennis P. Flanagan, "CONCHAS Users Manual," SAND88-1006, Sandia National Laboratories, Albuquerque, New Mexico, June 1989.

Exodus and Genesis File Format

William C. Mills-Curran, Amy P. Gilkey and Dennis P. Flanagan, "EXODUS: A Finite Element File Format for Pre- and Postprocessing," SAND87-2977, Sandia National Laboratories, Albuquerque, New Mexico, September 1988.

L. M. Taylor, D. P. Flanagan and W. C. Mills-Curran, "The GENESIS Finite Element Mesh File Format," SAND86-0910, Sandia National Laboratories, Albuquerque, New Mexico, May 1986.

Fastq

The FASTQ code is an interactive two-dimensional finite element mesh generation program. It is designed to provide a powerful and efficient tool to both reduce the time required of an analyst to generate a mesh, and to improve the capacity to generate good meshes in arbitrary geometries. It is based on a mapping technique and employs a set of higher-order primitives which have been developed for automatic meshing of commonly encountered shapes (i.e., the triangle, semi-circle, etc.) and conditions (i.e., mesh transitioning from coarse to fine mesh size). FASTQ has been designed to allow user flexibility and control. The user interface is built on a layered command level structure. Multiple utilities are provided for input, manipulation, and display of the geometric information, as well as for direct control, adjustment, and display of the generated mesh. Enhanced boundary flagging has been incorporated and multiple element types and output formats are supported. FASTQ includes the paving algorithm which meshes arbitrary two-dimensional geometries with an all quadrilateral mesh.

T. D. Blacker, "FASTQ Users Manual, Version 2.1," SAND88-1326, Sandia National Laboratories, Albuquerque, New Mexico, July 1988.

This paper presents a new mesh generation technique, paving, which meshes arbitrary two-dimensional geometries with an all-quadrilateral mesh. Paving allows varying element size distributions on the boundary as well as the interior of a region. The generated mesh is well formed (i.e., nearly square elements, elements perpendicular to boundaries, etc.) and geometrically pleasing (i.e., mesh contours tend to follow geometric contours of the boundary). In this paper we describe th theory behind this algorithmic/heuristic technique, evaluate the performance of the approach and present examples of automatically generated meshes.

T. D. Blacker and M. B. Stephenson, "Paving: A New Approach to Automated Quadrilateral Mesh Generation," International Journal for Numerical Methods in Engineering, Vol. 32, 1991, pp. 811-847.

Grope

Grope is a program that examines the input to a finite element analysis (which is in the GENESIS database format) or the output from an analysis (in the EXODUS database format). Grope allows the user to examine any value in the database. The display can be directed to the user's terminal or to a print file.

Amy P. Gilkey, "GROPE - A GENESIS/EXODUS Database Examination Program," RS1523/88/02, Sandia National Laboratories, Albuquerque, New Mexico.

Numbers

Numbers is a program which reads and stores data from a finite element model described in the EXODUS database format. Within this program are several utility routines which generate information about the finite element model. The utilities currently implemented in Numbers allow the analyst to determine information such as (1) the volume and coordinate limits of each of the materials in the model; (2) the mass properties of the model; (3) the minimum, maximum, and average element volumes for each material; (4) the volume and change in volume of a cavity; (5) the nodes or elements that are within a specified distance from a user-defined point, line, or plane; (6) an estimate of the explicit central-difference timestep for each material; (7) the validity of contact surfaces or slidelines; that is, whether two surfaces overlap at any point; and (8) the distance between two surfaces.

G. D. Sjaardema, "NUMBERS: A Collection of Utilities for Pre- and Postprocessing Two- and Three-Dimensional EXODUS Finite Element Models," SAND88-0737, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

Plastering

This report describes the progress of the three-dimensional mesh generation research, using plastering, during the 1990 fiscal year. Plastering is a three-dimensional extension of the two-dimensional paving technique. The objective is to fill an arbitrary volume with hexahedral elements. The plastering algorithm's approach to the problem is to remove rows of elements from the exterior of the volume. Elements are removed, one level at a time, until the volume vanishes. Special closure algorithms may be necessary at the center. The report also discusses the common development environment and software management issues.

M. B. Stephenson, S. A. Canann and T. D. Blacker, "Plastering: A New Approach to Automated, Three-Dimensional Hexahedral Mesh Generation, Progress Report I," SAND89-2192, Sandia National Laboratories, Albuquerque, New Mexico, February 1992.

Pronto2D and Pronto3D

PRONTO2D is a two-dimensional transient solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates. This Lagrangian finite element program uses an explicit time integration operator to integrate the equations of motion. Four node uniform strain quadrilateral elements are used in the finite element formulation. A number of new numerical algorithms which have been developed for the code are described in this report. An adaptive time step control algorithm is described which greatly improves stability as well as performance in plasticity problems. A robust hourglass control scheme which eliminates hourglass distortions without disturbing the finite element solution is included. All constitutive models in PRONTO are cast in an unrotated configuration defined using the rotation determined from the polar decomposition of the deformation gradient. An accurate incremental algorithm was developed to determine this rotation and is described in detail. A robust contact algorithm was developed which allows for the impact and interaction of deforming contact surfaces of quite general geometry. A number of numerical examples are presented to demonstrate the utility of these algorithms.

Lee M. Taylor and Dennis P. Flanagan, "PRONTO2D, A Two-Dimensional Transient Solid Dynamics Program," SAND86-0594, Sandia National Laboratories, Albuquerque, New Mexico, March 1987.

PRONTO3D is a three-dimensional transient solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates. This Lagrangian finite element program uses an explicit time integration operator to integrate the equations of motion. Eight-node

uniform strain hexahedral elements are used in the finite element formulation. A number of new numerical algorithms which have been developed for the code are described in this report. An adaptive time step control algorithm is described which greatly improves stability as well as performance in plasticity problems. A robust hourglass control scheme which eliminates hourglass distortions without disturbing the finite element solution is included. All constitutive models in PRONTO are cast in an unrotated configuration defined using the rotation determined from the polar decomposition of the deformation gradient. An accurate incremental algorithm was developed to determine this rotation and is described in detail. A robust contact algorithm was developed which allows for the impact and interaction of deforming contact surfaces of quite general geometry. Numerical examples are presented to demonstrate the utility of these algorithms.

L. M. Taylor and D. P. Flanagan, "PRONTO 3D, A Three-Dimensional Transient Solid Dynamics Program," SAND87-1912, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

An external code interface is defined which allows other transient applications to communicate with the PRONTO family of finite element programs. This interface is written in ANSI FORTRAN and allows an independent author to specify requirements for an external code to PRONTO. The interface is written such that updates to PRONTO will not require modifications to the external code.

L. M. Taylor and D. P. Flanagan, "An External Code Interface for the PRONTO Family of Transient Solid Dynamics Programs," SAND87-3003, Sandia National Laboratories, Albuquerque, New Mexico, September 1988.

PRONTO 2D and PRONTO 3D are two- and three-dimensional solid dynamics codes for analyzing large deformations of highly nonlinear materials subjected to high strain rates. This newsletter is issued to document changes to these codes. As of this writing, the latest version of PRONTO 2D is Version 4.5.6, and the latest version of PRONTO 3D is Version 4.5.6.

This update of the two codes discusses two major modifications to the numerical formulations, three new constitutive models, and the additions and improvements of contact surfaces. Changes in file formats, other miscellaneous revisions, and the availability of PRONTO 2D and PRONTO 3D are also discussed. In addition, updated commands for PRONTO 2D are provided in Appendix A of this newsletter.

S. W. Attaway, "Update of PRONTO 2D and PRONTO 3D Transient Solid Dynamics Program," SAND90-0102, Sandia National Laboratories, Albuquerque, New Mexico, November 1990.

PRONTO 3D is a three-dimensional transient solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to high strain rates. It is a Lagrangian finite element program with explicit integration of the equations of motion through time. This report documents the implementation of a four-node quadrilateral shell element into Version 6.0 of PRONTO 3D.

This report describes the theory, implementation and use of a four-node shell element. Also described are the required architectural changes made to PRONTO 3D to allow multiple element types. Several test problems are documented for verification of the PRONTO 3D implementation and for demonstration of computational savings using shell elements for thin structures. These problems also serve as examples for the user. A complete, updated list of the PRONTO 3D input commands is also included.

V. L. Bergmann, "Transient Dynamics Analysis of Plates and Shells with PRONTO 3D," SAND91-1182, Sandia National Laboratories, Albuquerque, New Mexico, September 1991.

This update discusses modifications of PRONTO 3D tailored to the design of fast burst nuclear reactors. A thermoelastic constitutive model and spatially variant thermal history load were added for

this special application. Included are descriptions of the thermoelastic constitutive model and the thermal loading algorithm, two example problems used to benchmark the new capability, a user's guide and PRONTO 3D input files for the example programs. The results from PRONTO 3D thermoelastic finite element analysis are benchmarked against measured data and finite difference calculations.

PRONTO 3D is a three-dimensional transient solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to high strain rates. The code modifications are implemented in PRONTO 3D Version 5.3.3.

D. S. Oscar, S. W. Attaway and J. D. Miller, "Modifications of the PRONTO 3D Finite Element Program Tailored to Fast Burst Nuclear Reactor Design," SAND91-0959, Sandia National Laboratories, Albuquerque, New Mexico, August 1991.

Subway 3D

SUBWAY 3D is a three-dimensional finite element code for numerical simulation of the transient electromechanical response of dielectric materials. The code uses a preconditioned conjugate gradient method to solve Poisson's equation governing the electric potential in the dielectrics. This field solver is embedded in a modified version of the finite element transient dynamics code PRONTO 3D and allows solution for the electric response at each time step used in the explicit integration of the equations of motion. The code is structured to allow flexibility in formulation of initial-boundary value problems by permitting specification of electrical conductors and allowing these conductors to be connected to electrical circuits isolated from the mechanical deformations. An algorithm for solution of these initial-boundary value problems is incorporated into the code with special material models to represent the response of ordinary dielectrics and electromechanically active dielectrics like piezoelectric and ferroelectric ceramics. The code has a wide range of applications and, in particular, can be used to calculate responses of shock activated power supplies, impact gages, and the change in electrical capacitance due to deformations.

S. T. Montgomery - documentation not yet available.

Sancho

SANCHO is a finite element computer program designed to compute the quasistatic, large deformation, inelastic response of planar or axisymmetric solids. Finite strain constitutive theories for plasticity, volumetric plasticity, and metallic creep behavior are included. A constant bulk strain, bilinear displacement isoparametric finite element is employed for the spatial discretization. The solution strategy used to generate the sequence of equilibrium solutions is a self-adaptive dynamic relaxation scheme which is based on explicit central difference pseudo-time integration and artificial damping. A master-slave algorithm for sliding interfaces is also implemented. A theoretical development of the appropriate governing equations and a description of the numerical algorithms are presented along with a user's guide which includes several sample problems and their solution.

Charles M. Stone, Raymond D. Krieg and Zelma E. Beisinger, "SANCHO, A Finite Element Computer Program for the Quasistatic, Large Deformation, Inelastic Response of Two-Dimensional Solids," SAND84-2618, Sandia National Laboratories, Albuquerque, New Mexico, April 1985.

Santos and Santos3D

SANTOS is a finite element computer program designed to compute the quasistatic, large deformation, inelastic response of planar or axisymmetric solids. SANTOS is based on the dynamics program PRONTO2D by L. M. Taylor and D. P. Flanagan. SANTOS utilizes a self-adaptive dynamic relaxation algorithm to achieve a quasistatic solution. The efficiency, speed, through vectorization, and state-of-the-art algorithms that Taylor and Flanagan built into PRONTO2D are maintained in SANTOS. The architecture of the code as well as the user interface is similar for both codes which improves code reliability and encourages the use of both codes by analysts. By utilizing the same material interface, the same constitutive models may be utilized by both codes which allows for coupling of the two codes.

C. M. Stone - documentation not yet available.

Jac2D, Jac3D and Jacq3D

The nonlinear conjugate gradient procedure is employed in the computer program JAC2D to solve quasi-static nonlinear mechanics problems. A set of continuum equations which describe accurately nonlinear mechanics involving large rotation and strain are very conveniently used with the conjugate gradient method to solve the nonlinear problem. The method is exploited in a two-dimensional setting while using various methods for accelerating convergence. Sliding interface conditions are also implemented. A four-node Lagrangian uniform strain element is used with hourglass stiffness to control the zero energy nodes. Materials which can be modeled are elastic and isothermal elastic-plastic with combined kinematic and isotropic hardening. The program is vectorized to perform very efficiently on CRAY computers. Sample problems described are the bending of a thin beam, the rotation of a unit cube, and a pressurized and thermally-loaded sphere and cylinder.

J. H. Biffle, "A Two-Dimensional Finite Element Computer Program for the Nonlinear Quasistatic Response of Solids with the Conjugate Gradient Method," SAND81-0998, Sandia National Laboratories, Albuquerque, New Mexico, April 1984.

JAC3D is a three-dimensional finite element program designed to solve quasi-static nonlinear mechanics problems. A set of continuum equations, which describe nonlinear mechanics involving large rotation and strain, is used. A nonlinear conjugate gradient method is used to solve the nonlinear equations. The method is implemented in a three-dimensional setting with various methods for accelerating convergence. Sliding interface conditions are also implemented. An eight-node Lagrangian uniform strain element is used with hourglass stiffness to control the zero energy modes. This release of the code contains elastic and isothermal elastic-plastic material models. Other material models can be added relatively easily. The program is vectorized to perform very efficiently on CRAY computers. Sample problems described are the bending of a thin beam, the rotation of a unit cube, and a pressurized and thermally loaded sphere.

J. H. Biffle, "JAC3D - A Three-Dimensional Finite Element Computer Program for the Nonlinear Quasistatic Response of Solids with the Conjugate Gradient Method," in preparation.

The nonlinear conjugate gradient procedure is employed in the computer program JAC3D to solve the steady-state and transient nonlinear heat conduction problem for solids in three dimensions. The program can also be used for other types of diffusion problems. The problem is formulated with the finite element method and employs an eight-node uniform gradient element with hourglass stiffness to control the zero energy modes. JACQ3D is highly vectorized to perform efficiently on the CRAY computer. Sample problems are included to verify the code and to provide examples of the use of the code.

J. H. Biffle, "JAC3D - A Three-Dimensional Finite Element Computer Program for Nonlinear Heat Conduction Problems with the Conjugate Gradient Method," in preparation.

Merlin II

The MERLIN II program is designed to transfer data between finite element meshes of arbitrary geometry. The program is structured to accurately interpolate previously computed solutions onto a given mesh and format the resulting data for immediate use in another analysis program. Data from either two-dimensional or three-dimensional meshes may be considered. The theoretical basis and computational algorithms used in the program are described and complete user instructions are presented. Several example problems are included to demonstrate program usage.

D. K. Gartling, "MERLIN II - A Computer Program to Transfer Solution Data Between Finite Element Meshes," SAND89-2989, Sandia National Laboratories, Albuquerque, New Mexico, July 1991.

Supes Library

The Software Utilities Package for the Engineering Sciences (SUPES) is a collection of subprograms which perform frequently used non-numerical services for the engineering applications programmer. The three functional categories of SUPES are: (1) input command parsing, (2) dynamic memory management, and (3) system dependent utilities. The subprograms in categories one and two are written in standard FORTRAN-77, while the subprograms in category three are written to provide a standardized FORTRAN interface to several system dependent features.

J. H. Red-Horse, W. C. Mills-Curran and D. P. Flanagan, "SUPES Version 2.1, A Software Utilities Package for the Engineering Sciences," SAND90-0247, Sandia National Laboratories, Albuquerque, New Mexico, May 1990.

Gen3D

GEN3D is a three-dimensional mesh generation program. The three-dimensional mesh is generated by mapping a two-dimensional mesh into three-dimensions according to one of four types of transformations: translating, rotating, mapping onto a spherical surface, and mapping onto a cylindrical surface. The generated three-dimensional mesh can then be reoriented by offsetting, reflecting about an axis, and revolving about an axis. GEN3D can be used to mesh geometries that are axisymmetric or planar, but, due to three-dimensional loading or boundary conditions, require a three-dimensional finite element mesh and analysis. More importantly, it can be used to mesh complex three-dimensional geometries composed of several sections when the sections can be defined in terms of transformations of two-dimensional geometries. The code GJOIN is then used to join the separate sections into a single body. GEN3D reads and writes two-dimensional and three-dimensional mesh databases in the GENESIS database format; therefore, it is compatible with the preprocessing, postprocessing, and analysis codes used by the Engineering Analysis Department at Sandia National Laboratories, Albuquerque, New Mexico.

Amy P. Gilkey and Gregory D. Sjaardema, "GEN3D: A GENESIS Database 2D to 3D Transformation Program," SAND89-0485, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

This memo describes the changes that have been made to the GEN3D program since the manual (SAND89-0485) was published. The changes include: (1) new mesh generation options: spline, project, twist, interval, and rotcen transformations; (2) new mesh modification options: change

material, change sideset, and change nodeset; (3) new mesh orientation option: scale; and (4) miscellaneous changes.

G. D. Sjaardema, "Updates to the mesh generation program GEN3D," memo to distribution dated April 11, 1990, Sandia National Laboratories, Albuquerque, New Mexico.

Gjoin

Gjoin is a computer program which is used to merge two or more GENESIS databases into a single GENESIS database.

G. D. Sjaardema, "GJOIN: A Program for Merging Two or More GENESIS Databases," (memo to distribution dated June 19, 1991), Sandia National Laboratories, Albuquerque, New Mexico.

Grepos

GREPOS is a mesh utility program that repositions or modifies the configuration of a 2D or 3D mesh. Grepos can be used to change the orientation and size of a 2D or 3D mesh; change the material block, nodeset, and sideset IDs; or "explode" the mesh to facilitate viewing of the various parts of the model. Grepos also updates the EXODUS QA and information records to help track the codes and files used to generate the mesh. GREPOS reads and writes 2D and 3D mesh databases in the GENESIS database format; therefore, it is compatible with the preprocessing, postprocessing, and analysis codes in SEACAS.

G. D. Sjaardema, "GREPOS: A GENESIS Database Repositioning Program," SAND90-0566, Sandia National Laboratories, Albuquerque, New Mexico, April 1990.

Aprepro

APREPRO is a translator that reads a text file containing both general text and algebraic expressions. It echoes the general text to the output file, along with the results of the algebraic expressions. The syntax used in APREPRO is such that all expressions between the delimiters '{' and '}' are evaluated and all other text is simply echoed to the output file.

G. D. Sjaardema, "Aprepro: An Algebraic Preprocessor for Input Files," memo to distributed dated May 18, 1990, Sandia National Laboratories, Albuquerque, New Mexico.

The initial implementation of a units conversion capability in Aprepro has been completed. The units conversion is a very useful capability for analysts and it is required for the implementation of the MATS material database system.

G. D. Sjaardema, "Implementation of Units Conversion in Aprepro," memo to distribution dated April 24, 1992, Sandia National Laboratories, Albuquerque, New Mexico.

Constitutive Models

B. J. Thorne, "A Damage Model for Rock Fragmentation and Comparison of Calculations with Blasting Experiments in Granite," SAND90-1389, Sandia National Laboratories, Albuquerque, New Mexico, October 1990.

E. P. Chen, "A Computational Model for Jointed Media with Orthogonal Sets of Joints," SAND86-1122, Sandia National Laboratories, Albuquerque, New Mexico, March 1987.

M. K. Neilsen, H. S. Morgan, R. D. Krieg, "A Phenomenological Constitutive Model for Low Density Polyurethane Foams," SAND86-2927, Sandia National Laboratories, Albuquerque, New Mexico, April 1987.

R. S. Chambers, "A Viscoelastic Material Model for Computing Stresses in Glass," SAND90-0645, Sandia National Laboratories, Albuquerque, New Mexico, July 1990.

G. D. Sjaardema and R. D. Krieg, "A Constitutive Model for the Consolidation of WIPP Crushed Salt and Its Use in Analyses of Backfilled Shaft and Drift Configurations," SAND87-1977, Sandia National Laboratories, Albuquerque, New Mexico, October 1987.

C. M. Stone, G. W. Wellman and R. D. Krieg, "A Vectorized Elastic/Plastic Power Law Hardening Material Model Including Luders Strain," SAND90-0153, Sandia National Laboratories, Albuquerque, New Mexico, March 1990.

J. R. Weatherby, R. D. Krieg and C. M. Stone, "Incorporation of Surface Tension into the Structural Finite Element Code SANCHO," SAND89-0509, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

S. W. Attaway, "A Local Isotropic/Global Orthotropic Finite Element Technique for Modeling the Crush of Wood," SAND88-1449, Sandia National Laboratories, Albuquerque, New Mexico, September 1988.

R. D. Krieg, "A Simple Constitutive Description for Soils and Crushable Foams," SC-DR-72-0883, Sandia National Laboratories, Albuquerque, New Mexico, 1987.

D. J. Bammann, "An Internal Variable Model of Viscoplasticity," International Journal of Engineering Science, Vol. 22, 1984, pp. 1041-1053.