NASA/CP—1998-208525

Visual Computing Environment Workshop

Proceedings of a conference held at and sponsored by
NASA Lewis Research Center
Cleveland, Ohio
November 6, 1997

National Aeronautics and
Space Administration

Lewis Research Center

November 1998

Available from

NASA Center for Aerospace Information
7121 Standard Drive
Hanover, MD 21076
Price Code: A04

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22100
Price Code: A04

Introduction

The Visual Computing Environment (VCE) is a framework for intercomponent and multidisciplinary computational simulations. Many current engineering analysis codes simulate various aspects of aircraft engine operation. For example, existing computational fluid dynamics (CFD) codes can model the airflow through individual engine components such as the inlet, compressor, combustor, turbine, or nozzle. Currently, these codes are run in isolation, making intercomponent and complete system simulations very difficult to perform. In addition, management and utilization of these engineering codes for coupled component simulations is a complex, laborious task, requiring substantial experience and effort. To facilitate multicomponent aircraft engine analysis, the CFD Research Corporation (CFDRC) is developing the VCE system. This system, which is part of NASA's Numerical Propulsion Simulation System (NPSS) program, can couple various engineering disciplines, such as CFD, structural analysis, and thermal analysis.

The objectives of VCE are to (1) develop a visual computing environment for controlling the execution of individual simulation codes that are running in parallel and are distributed on heterogeneous host machines in a networked environment, (2) develop numerical coupling algorithms for interchanging boundary conditions between codes with arbitrary grid matching and different levels of dimensionality, (3) provide a graphical interface for simulation setup and control, and (4) provide tools for online visualization and plotting.

VCE was designed to provide a distributed, object-oriented environment. Mechanisms are provided for creating and manipulating objects, such as grids, boundary conditions, and solution data. This environment includes parallel virtual machine (PVM) for distributed processing. Users can interactively select and couple any set of codes that have been modified to run in a parallel distributed fashion on a cluster of heterogeneous workstations. A scripting facility allows users to dictate the sequence of events that make up the particular simulation.

The purpose of this Workshop was to bring together VCE developers and end users to share experiences. Presentations were given by CFD Research Corporation on the overall VCE project, VCE system architecture, interfacing technology, a data transfer format (DTF), and VCE applications. NASA Lewis personnel presented uses of VCE for inlet-engine simulations and the National Combustor Code (NCC). Pratt & Whitney and Allied Signal presented their use of VCE for engineering applications at their respective companies.

Lewis contacts: Dr. Charles Lawrence, (216) 433-6048, CLawrence@lerc.nasa.gov; and Don Van Drei, (216) 433-9089, Donald.E.VanDrei@lerc.nasa.gov

TABLE OF CONTENTS

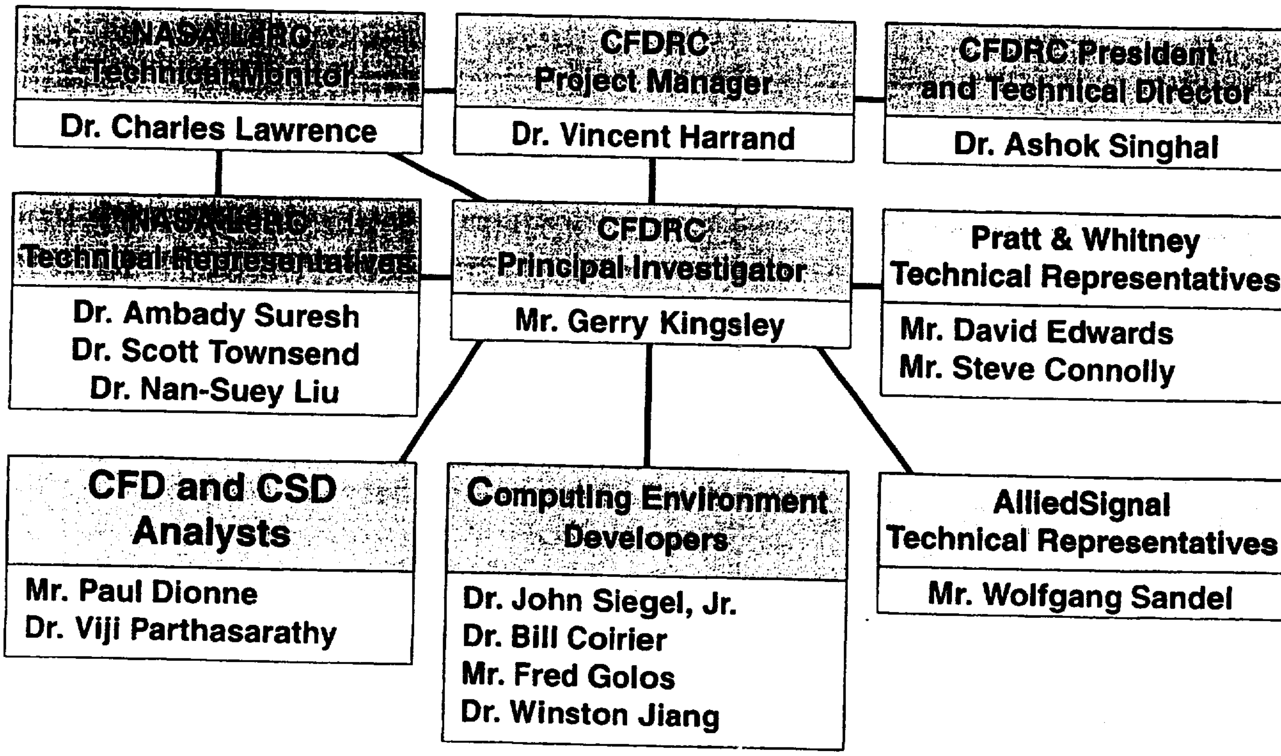
VISUAL COMPUTING ENVIRONMENT
VCE Project Overview

Gerry Kingsley
CFD Research Corporation
Huntsville, Alabama

Outline

- Project Overview
    - Key Personnel
    - Objectives
    - Task Descriptions
    - Development Approach
    - Time Line
    - Project History
- FY 97 Accomplishments
    - VCE System Overview
    - VCE Applications
- Fy 98 Plans
    - VCE Improvements
    - Code Integration
- Concluding Remarks

Key Personnel



Objectives

- Enhance the VCE software
- Facilitate Easier Incorporation of NASA Computer Codes
- Provide a Controlled Execution Environment
- Utilize a Network of Distributed Workstations
- Multi-Domain Capability
- Time Accurate Simulations
- Multi-Disciplinary Simulations
- Standard API

Task Descriptions

- Design of the Computing Environment
  - Critical Study of the Current Analysis Process and Tools
  - Identification of Current Deficiencies
  - Finalize Design of Proposed System
- Development of the Software System
  - Development of the Computing Environment
  - Inclusion of Analysis Software (CFDRC, NASA, Third Party)
- Time Dependent Visualization of Distributed Data Sets
- Testing, Including Evaluation and Demonstration of:
  - Correct Functioning of Core Software Modules
  - Flexibility of Including Various Codes into Environment
  - Computational Efficiency
  - User Effectiveness in the Overall Analysis Process
- Documentation

Development Approach

Iterative Software Development Process

Develop Prototype Environment

Include Selected Codes from CFD, CSD, Geometry, Grid, and Visualization
- CFDRC
- NASA LeRC
- Third Party

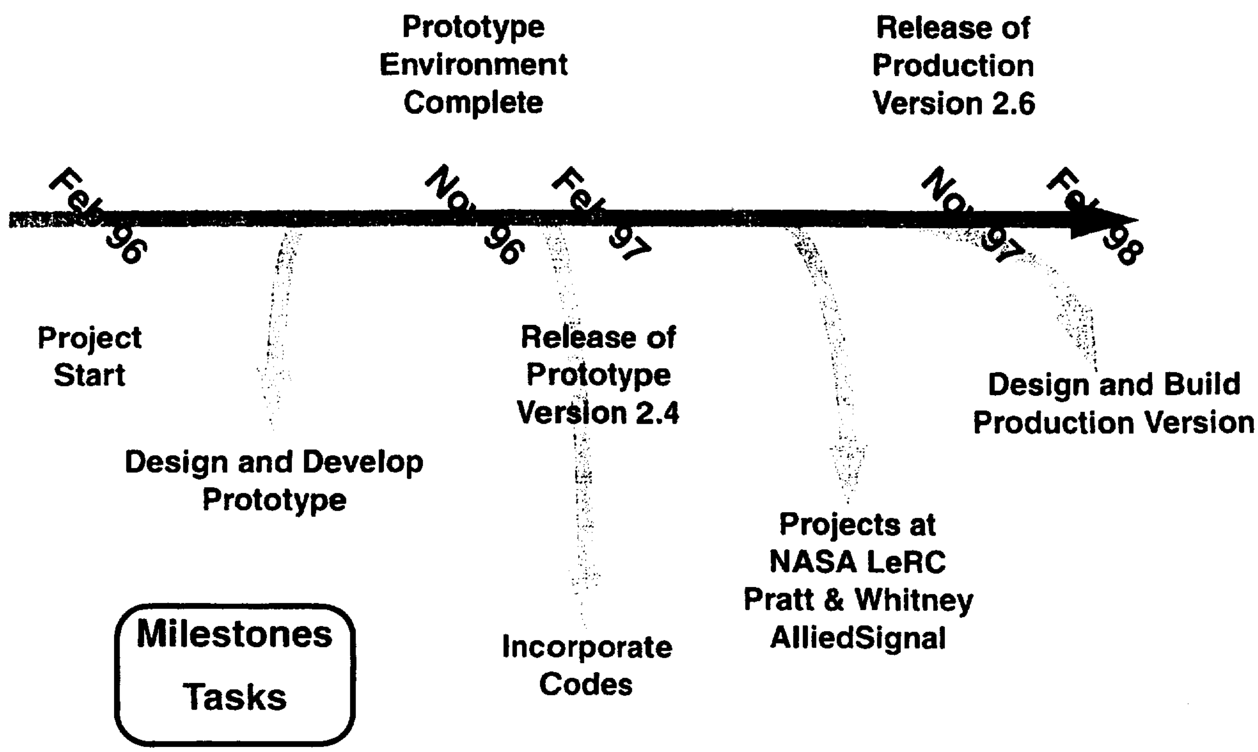
Demonstrate Environment for Selected Problems

Provide Interim Versions to Interested Users
- NASA LeRC
- Pratt & Whitney
- AlliedSignal

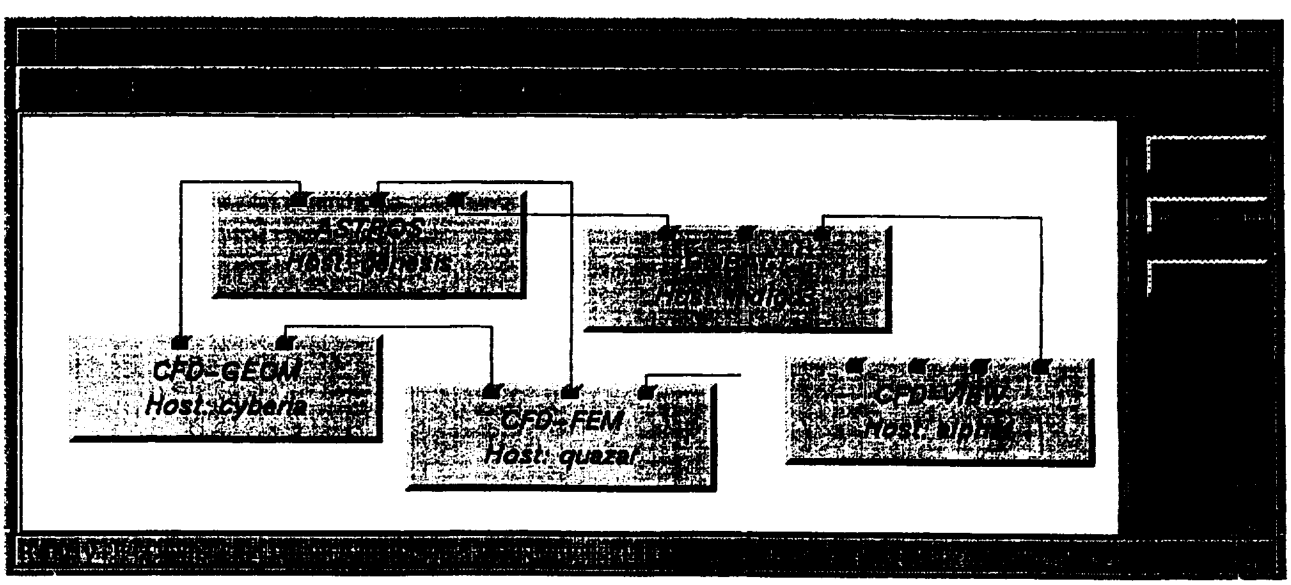
Users Provide Feedback

Refine and Revalidate Computing Environment

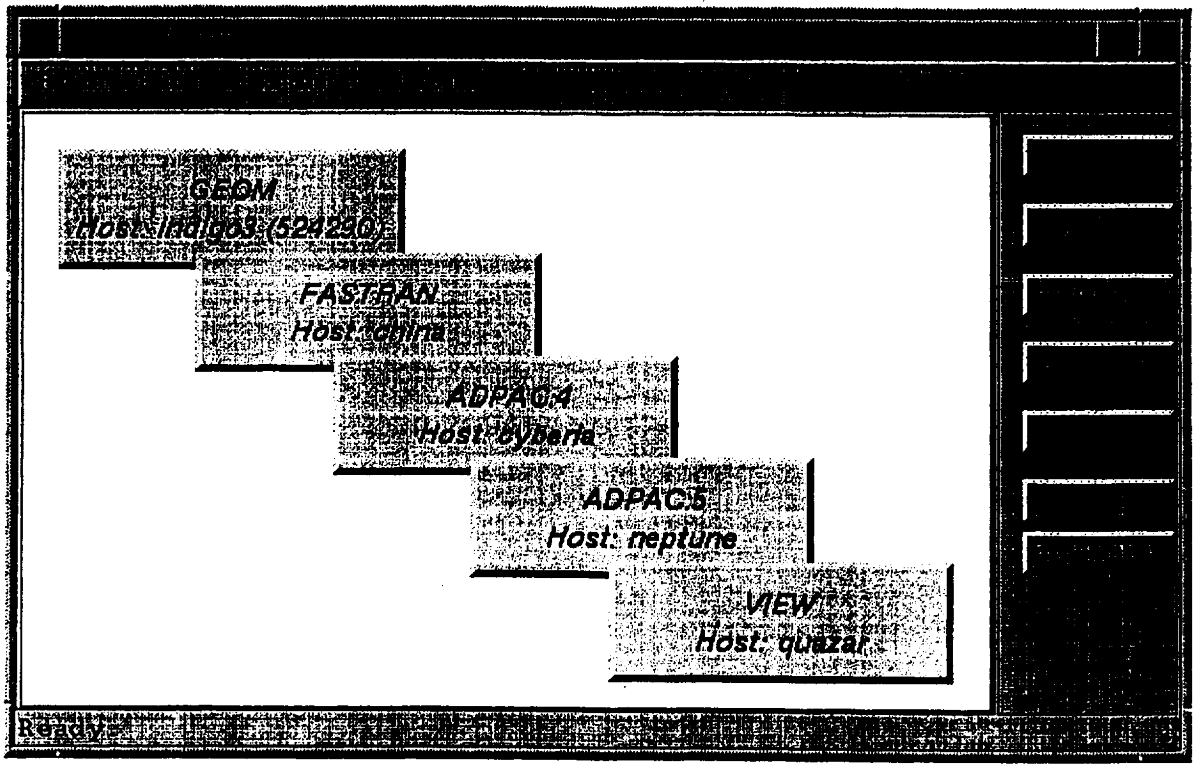
Time Line

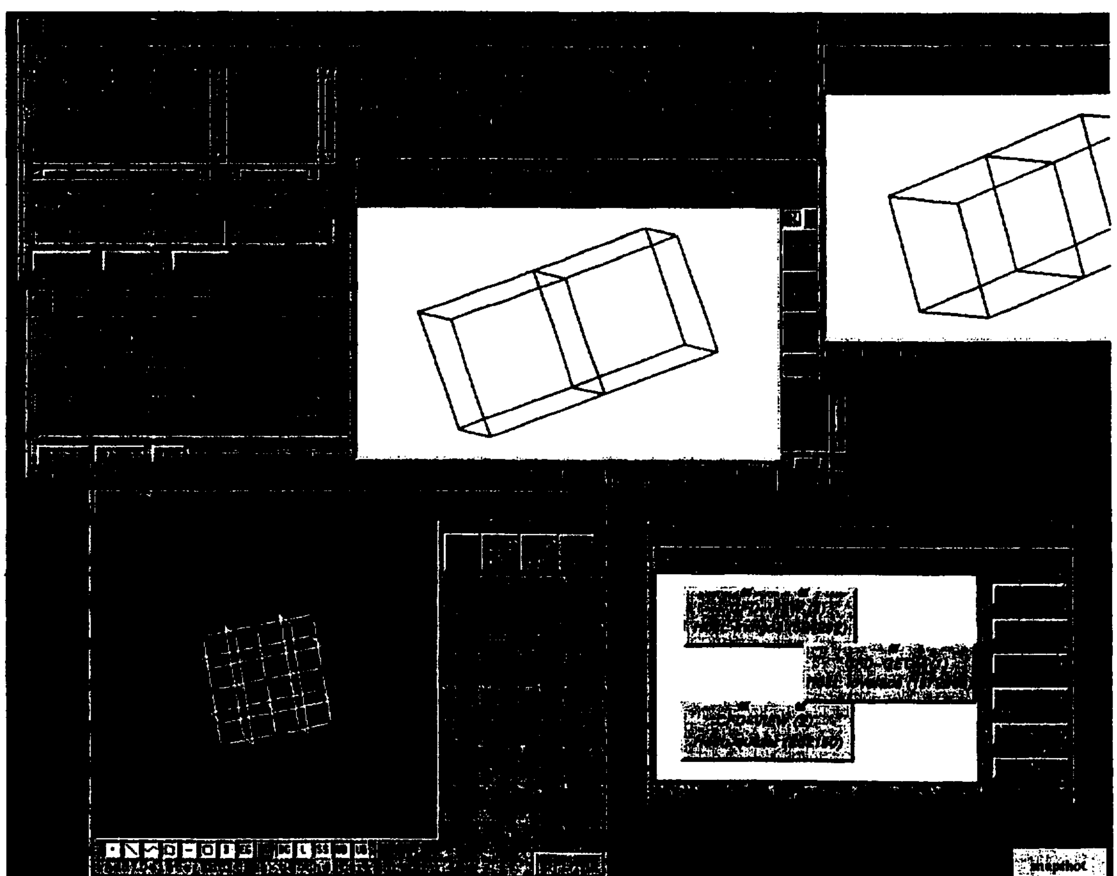


Ancient History

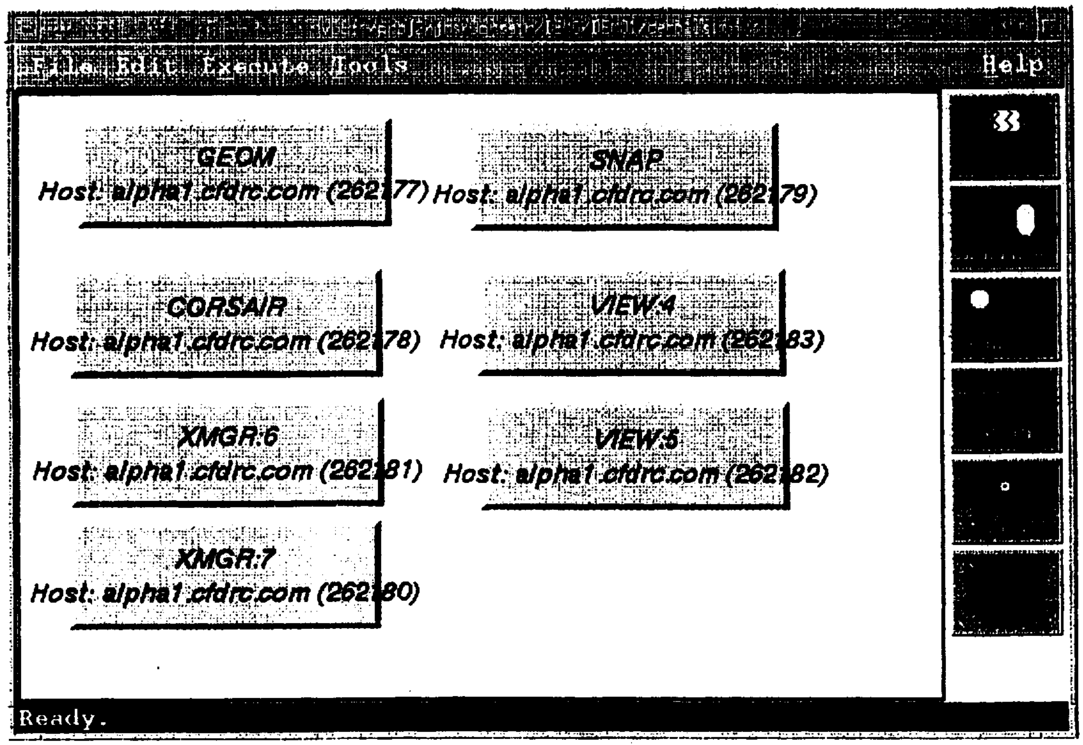


A Little More Recent

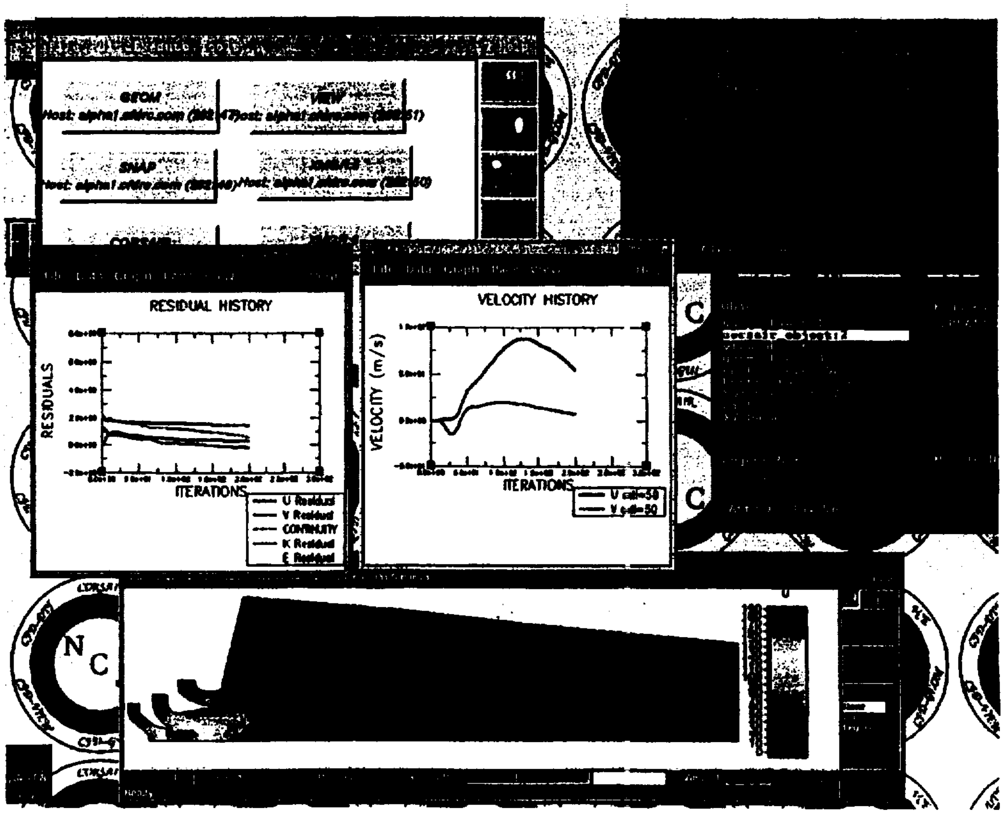




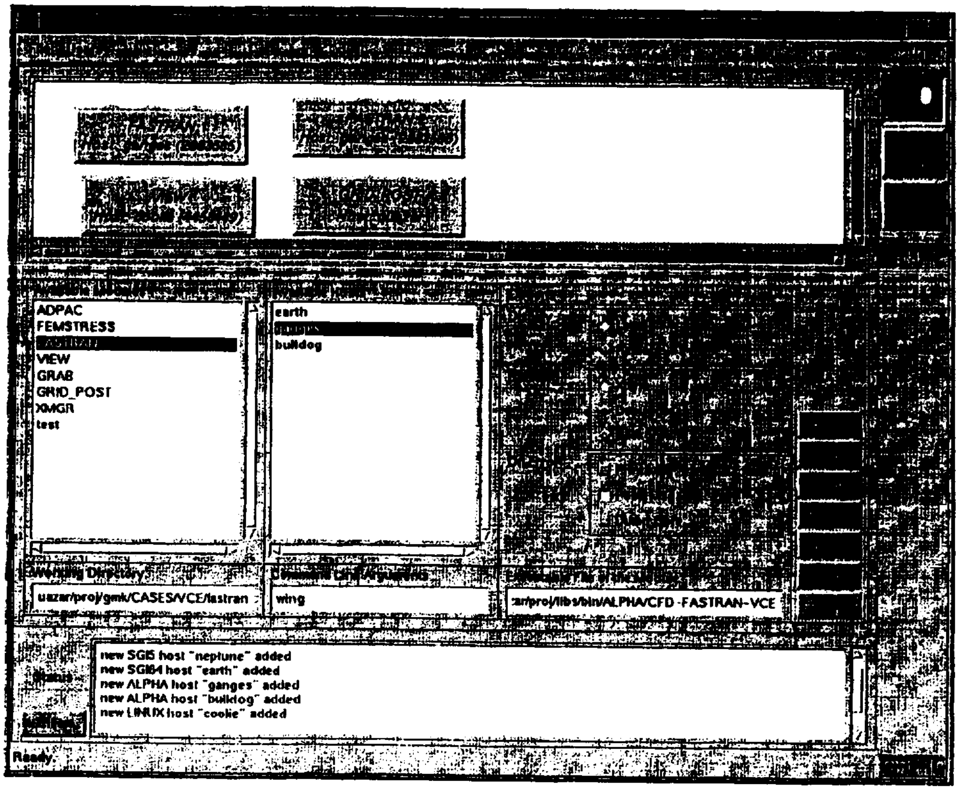
Version 2.4 Released February 1997



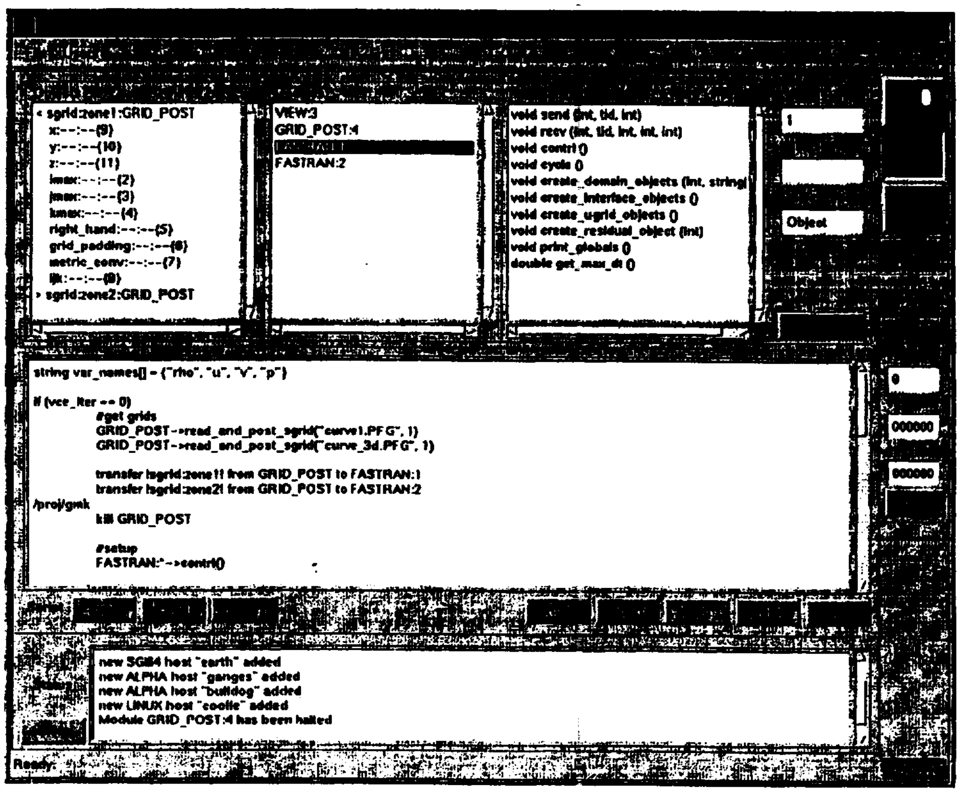
An Application Using Version 2.4



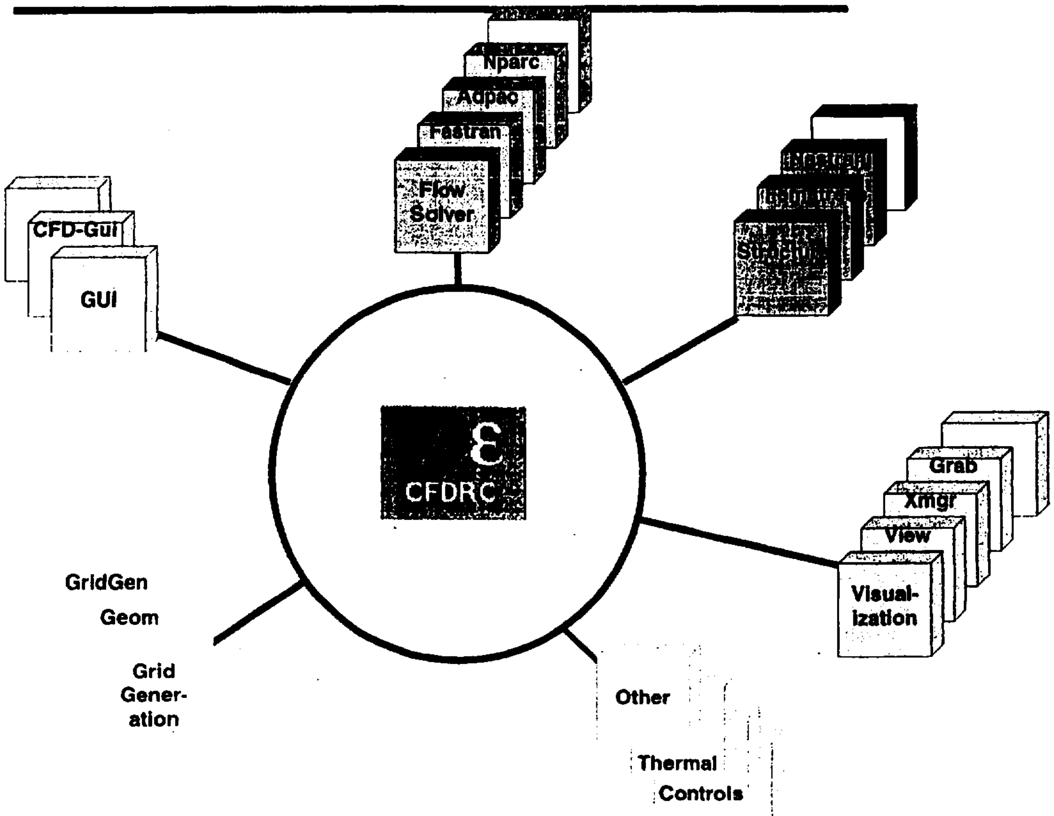
The VCE Graphical User Interface: Module View



VCE Graphical User Interface:
Object / Script View



VCE Conceptual Overview



VCE Basic Concepts

Multi-Disciplinary Simulation Consisting of a Set Of Interoperating Codes

Code Integration is Obtained by

- Dynamic Sharing of Data
- Dynamic Execution Control and Synchronization
- Multi-Disciplinary Interfacing Technology

Generality, Flexibility, and Extensibility are Achieved Using an Object Oriented Paradigm

Full End-User Control by Means of a Script

The user should be able to add/remove/exchange Modules in the environment and tailor the environment for specific needs without modification to the modules or the environment.

Advantages of the VCE Approach

Avoid Monolithic Giant Codes

- Difficult to Develop and Maintain
- No Up-to-Date Technology

Reuse of Existing (Trusted) Codes

Flexibility, Exchange of Modules, Apply Best Technology for the Case at Hand

Parallel Distributed Heterogeneous Computing Environment

Process Improvements

- Higher Level of Automation
- Efficiency
- End-User Effectiveness

Other VCE Applications

Domain Decomposition
- Parallel execution of non-parallelized codes
- Different Flow Solvers per domain
- Structured & Unstructured Grids
- Arbitrarily Matched Grids
- Single & Double Precision Domains
- Zooming (Coupling 2D and 3D Domains)
- Multi-Scale Analysis

Time Dependent Visualization with Multiple Views

Visualization of Distributed Data Sets

Animation / Capturing of Transient Simulations

Integration with Controls

FY 98 Plans

VCE Improvements
- Ability to run in batch mode
- Ability to run under the control of a job scheduler
- Better support for independently parallelized codes
- Checkpointing / Process Migration

Code Integration
- CFDRC codes have been integrated
- Focus during FY98 on other codes

New Applications
- Turbo Machinery
- ???

Collaboration

Concluding Remarks

A Variety of Multi-Disciplinary Interface Technologies has been implemented

Several Applications have been Demonstrated

The Component-Based Simulation Software Enables Method-to-Method Comparisons

VCE is an Open-Ended System. Incorporation of New Technologies, Disciplines, and Codes is Straightforward

New Applications are Desired !

Collaborations are Encouraged !

VISUAL COMPUTING ENVIRONMENT
VCE System Architecture

Gerry Kingsley
CFD Research Corporation
Huntsville, Alabama

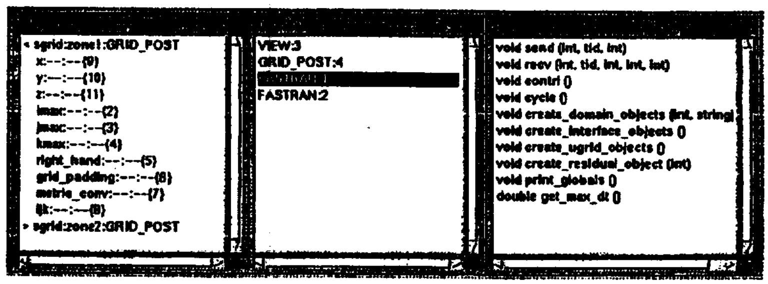
A Visual Computing Environment

VCE provides a visual interface to a facility allowing users to utilize a network of heterogeneous computers to:

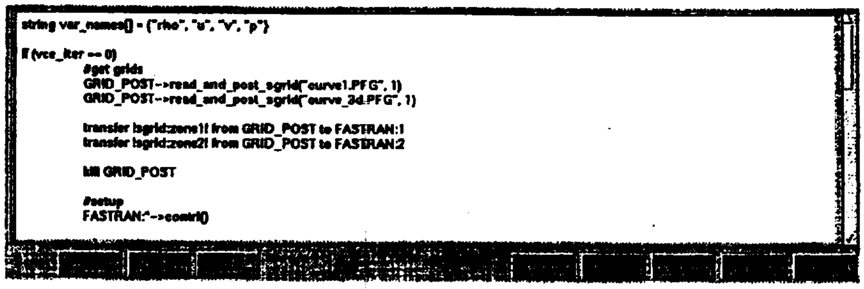
- **Run many programs simultaneously**
- **Control the programs**
  - **set up each individual program**
  - **issue commands to the programs**
- **Cause the programs to communicate**

Components of the Visual Interface

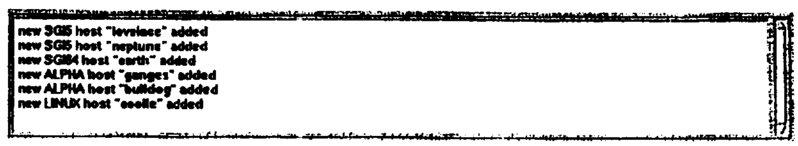
- **Object Clipboard**



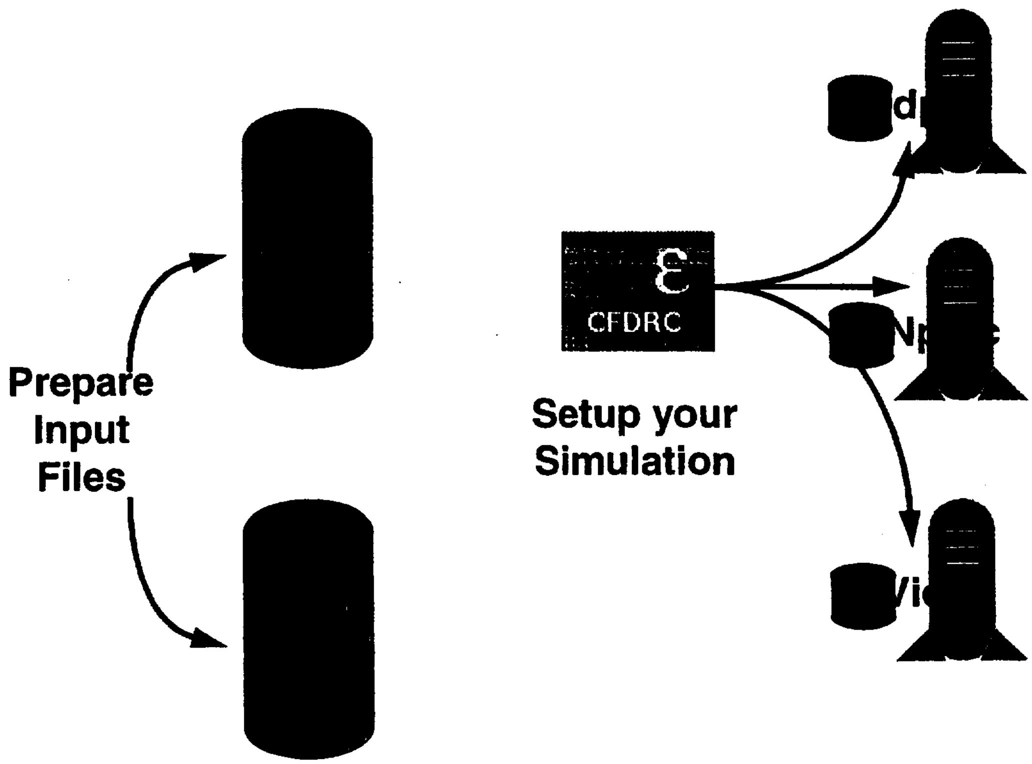
- **Script Editor**



- **Status Window**



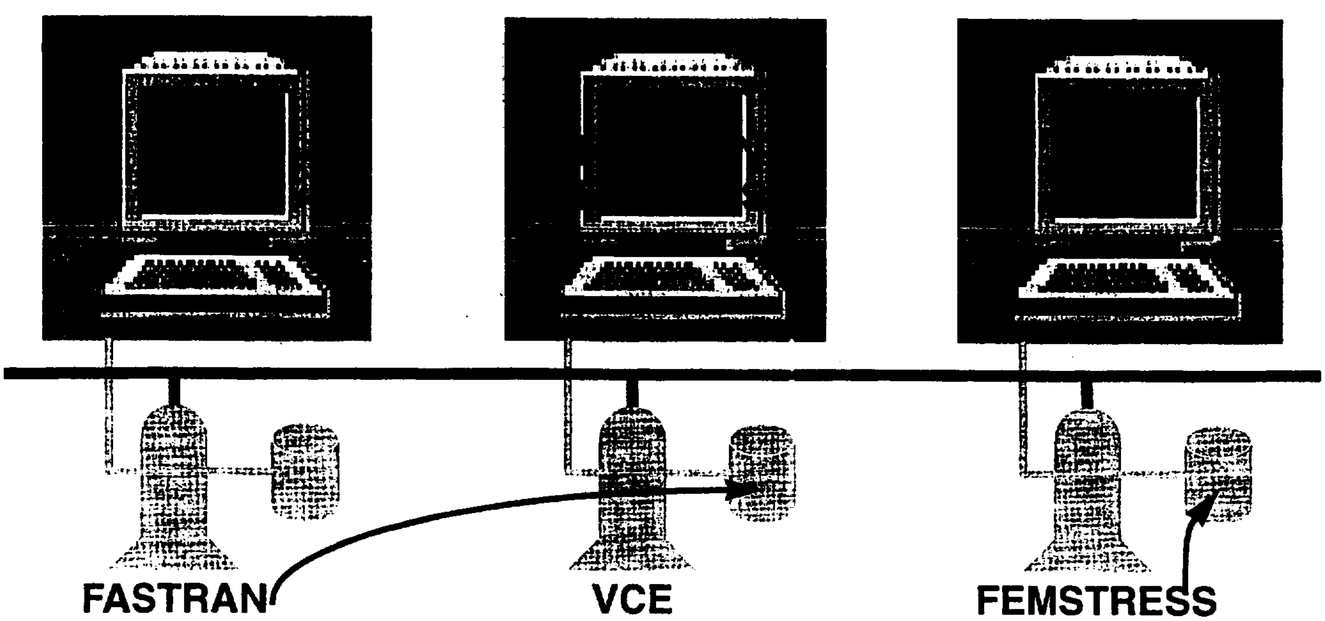
How Do I Use VCE?



VCE Features

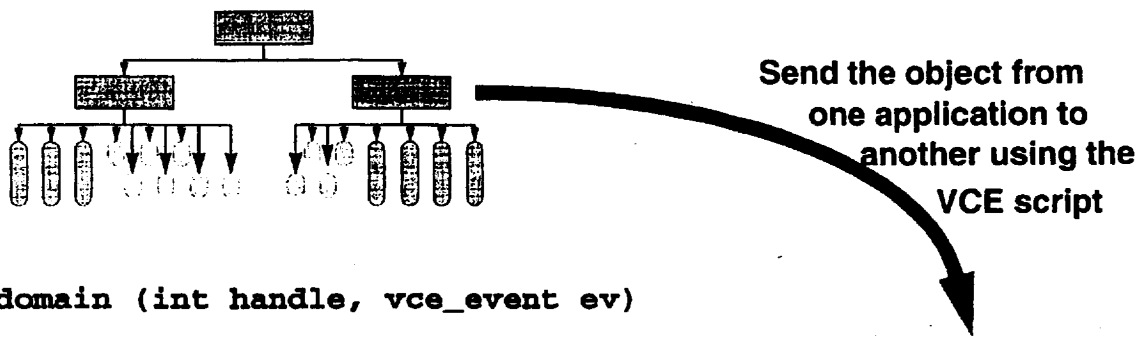
- **Application Control** — Features enabling the end user to control various application programs
- **Object Clipboard** — A centralized location accessible by all applications where descriptions of data may be stored
- **Event Handling** — Applications may request notification of certain events; control is transferred to the application when the event occurs
- **Remote Procedure Calls** — The ability for an application to execute an internal subroutine as the result of an external command
- **Scripting Mechanism** — A facility whereby the engineer can specify a series of remote procedure calls

A VCE Simulation Using 3 Workstations

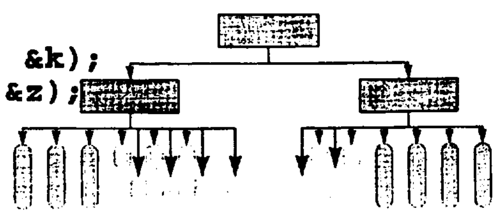


VCE Objects

```
DO i = 1, nblks
  CALL vcef_create_sdata (i, size, nvars, data, names, data_h)
  CALL vcef_create_sgrid (i, imax, jmax, kmax, x, y, z, grid_h)
  CALL vcef_create_domain (grid_h, data_h, domain_h)
ENDDO
```




```
recv_domain (int handle, vce_event ev)
{
    vce_get_domain_grid (handle, &grid_h);
    vce_get_sgrid_indices (grid_h, &i, &j, &k);
    vce_get_sgrid_coords (grid_h, &x, &y, &z);

    doView();
}
```



Event Handling

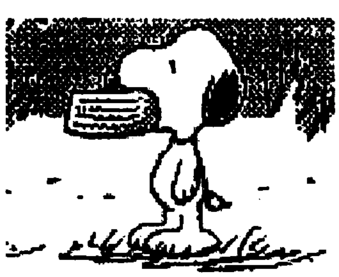
Windows programs are Event Driven programs. They spend most of their time waiting for the user to do something interesting:

- **Press a button**
- **Choose an option from a menu**
- **Move or click the mouse**

Programs integrated into VCE must *become* Event Driven:

- **Subroutines will be called from a VCE script rather than from** `PROGRAM MAIN`
- **Object related events such as** `create`, `send`, `receive`, **and** `destroy` **must be handled.**

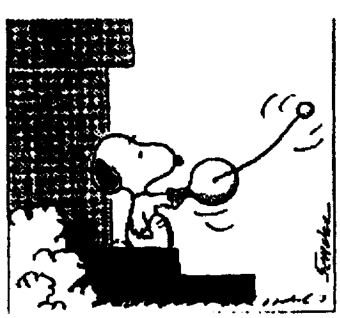
Event Handling



Prepare for the Event:

when I receive a grid object, call subroutine recv_grid

```
call vcef_add_callback
  ("grid",receive,recv_grid)
```

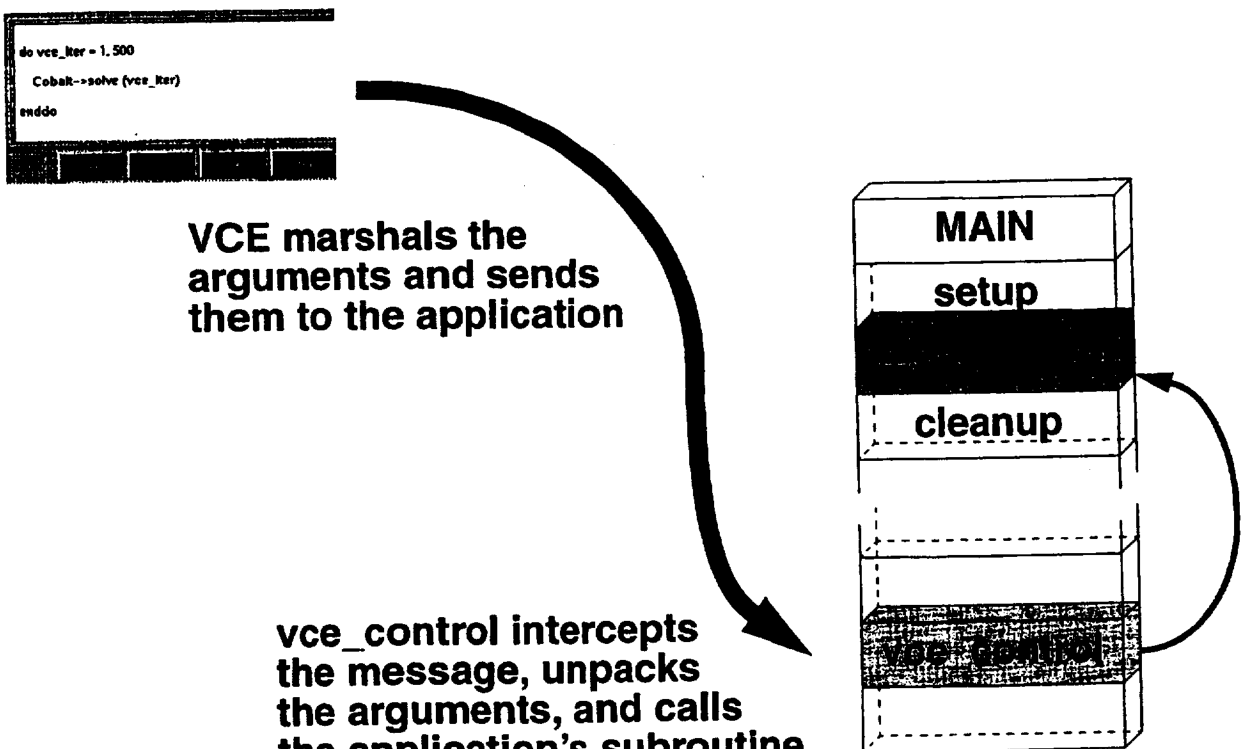


Wait for the event:

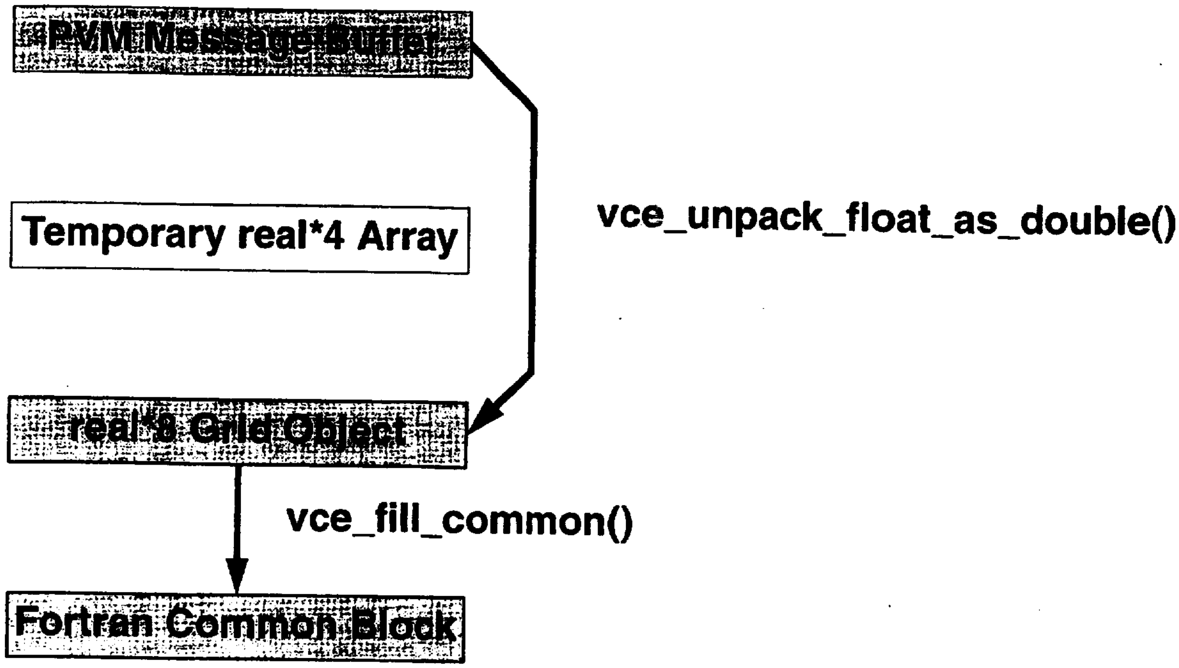
transfer control to VCE

```
call vcef_control
```

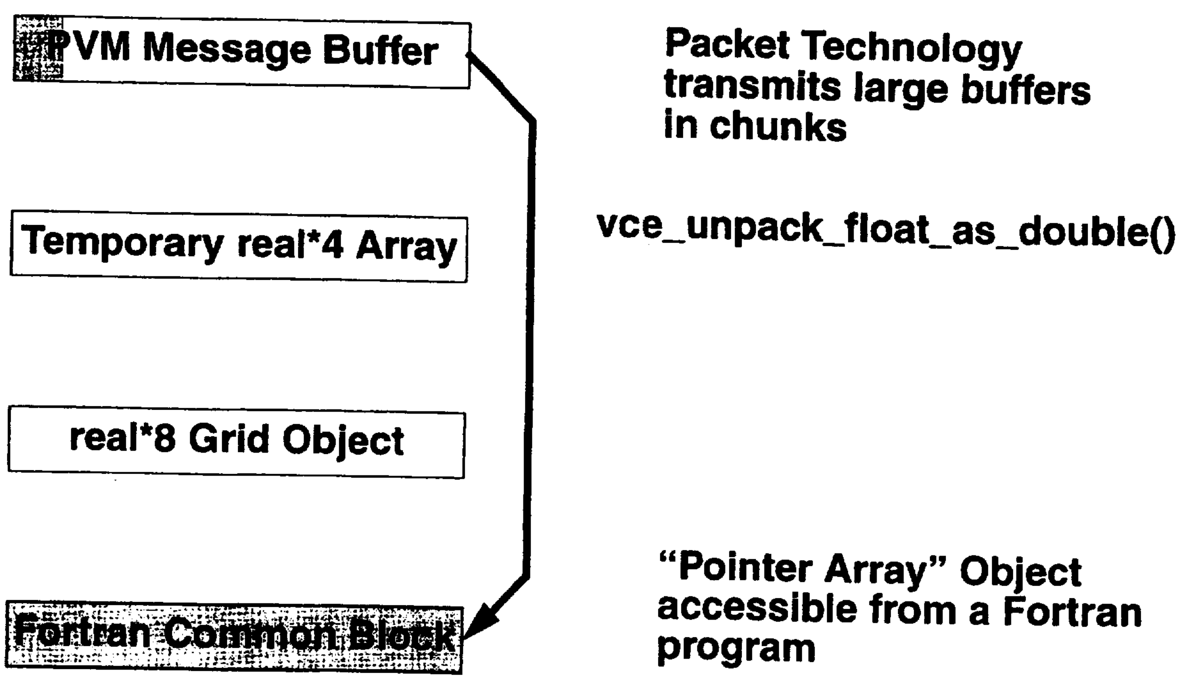
Remote Procedure Calls



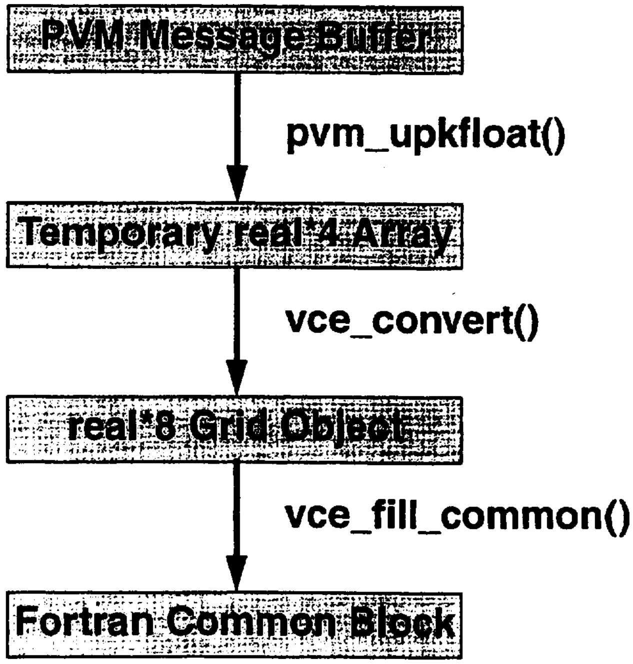
Avoiding Data Duplication



Avoiding Data Duplication



Avoiding Data Duplication



The Use of PVM in the Communication Layer

Changing to CORBA now is not feasible:

- **CORBA does not provide a FORTRAN interface; the C interface is weak.**

- **CORBA requires that C++ implementations provide exception handling. The current release of C++ from SGI does not implement exception handling, the most recent version of GNU g++ does not implement exception handling; others are even farther behind.**

- **The effort required would be enormous; the only benefit would be that application programs integrated into MDICE would also be able to communicate with other CORBA compliant codes. This does not imply that these other codes would be able to make use of MDICE interpolation abilities.**

VCE INTERFACING TECHNOLOGY

John M. Siegel, Jr.
CFD Research Corporation
Huntsville, Alabama

VCE INTERFACING TECHNOLOGY

OUTLINE

- Introduction
- Interpolation methods
- Interfacing in the VCE environment
- Applications

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Laplacian Interpolation-Circumferential Averaging

Circumferential Averaging is implemented by:

- Creating a 'phantom' axisymmetric grid
- Interpolating local 3D data to the local axisym grid (zooming)
- Interpolating the local axisym values to opposing 3D interface (zooming)

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS
Flux-conserving interpolation

CFDRC's flux conserving interpolation method uses geometrical clipping to distribute fluxes from one interface to another. The basic algorithm is described below:

- First, face centered flux and gradients of flux are computed using the Laplacian interpolation algorithm

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Laplacian Interpolation-Zooming

The CFDRC interpolation library currently supports zooming (2d->3d & axi-sym->3d). Zooming is implemented as follows (described for cell-centered codes):

- All geometry is entered in 2D ADT search trees. For the case of axi-3D, the nodal coordinates are converted to cylindrical coords before enrollment in the ADT tree.
- Face centered data is interpolated to the nodes & gradients are constructed in the same fashion as would be done with an equi-dimensional opposing interface (using Laplacian interpolation)

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Laplacian Interpolation-Zooming

- For interpolating 2D data to a 3D grid, the nearest 2D face to an opposing 3D face is determined through the 2D ADT tree. The 3D value is then interpolated from the 2D face (ignoring the third dimension--either z or theta)
- For interpolating 3D data to a 2D grid, all 3D faces which are 'near' to the 2D face are selected using the 2D ADT tree. Each of the 'near' 3D faces is then interpolated to the 2D face. Each of the interpolated 3D->2D face values is averaged to obtain the final 2D value.
- In the case of axi-sym to 3D (or visa versa), velocity vectors are appropriately converted from cartesian to cylindrical coordinates (or visa versa).

VCE INTERFACING TECHNOLOGY

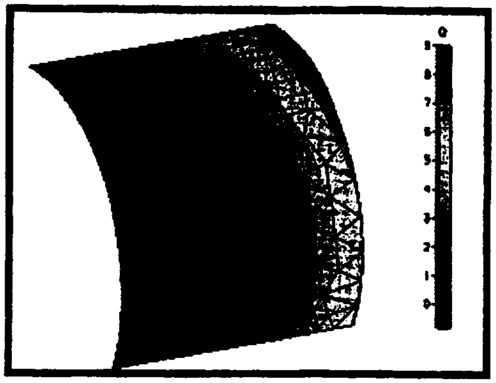
INTERPOLATION METHODS:
Laplacian Interpolation (cont)

Relevant features of the Laplacian interpolation methods:

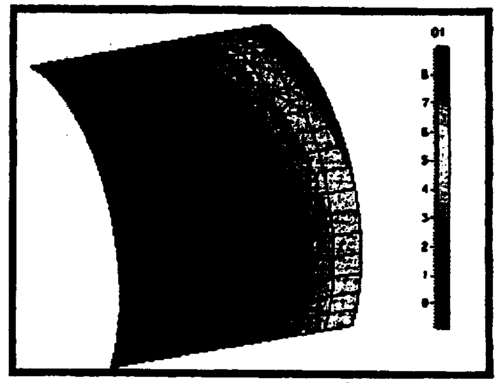
- 100% accurate for the interpolated of linear functions.
- 2D and 3D interface grids are supported
- Weighting factors are only computed once (therefore interpolation is fast)
- All grid alignment is automatic (accomplished by using fast geometrical searching algorithms)
- Unstructured and structured grids are supported.
- Data can be cell-centered or vertex-based.

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Laplacian Interpolation (cont)



Input: Node-based data for fxn: x^3 + y^3 + z^3



Output: center-based data (interpolated back to nodes for visualization)

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
FASIT Methods

FASIT methods employ geometrical-based surface fitting of values on the interface. Available methods:

- Thin plate spline
- Infinite plate spline
- NURBS

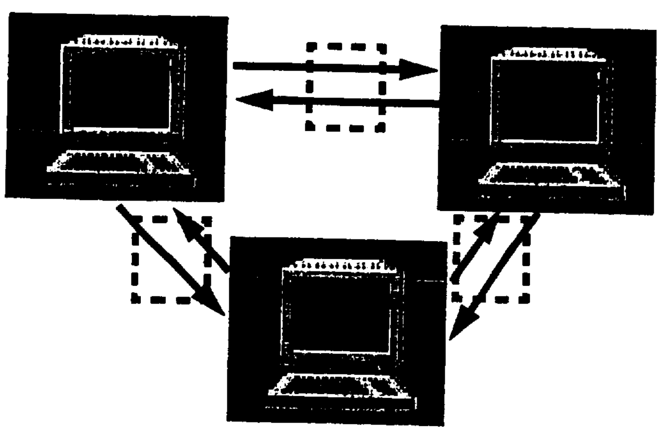
These methods are currently limited to structured grids. The FASIT gui requires the end-user to manually align grids.

VCE INTERFACING TECHNOLOGY

Interpolation methods currently available in the MDICE environment:

- FASIT methods (to be integrated)
- Laplacian interpolation (function matching)
- Flux interpolation (flux conserving interpolation for fluid-fluid interfaces)
- Consistent interpolation (conservative and consistent interpolation for fluid-structure interfaces)

VCE INTERFACING TECHNOLOGY



VCE interfaces add functionality to computer codes:

- Automated interface geometry definition and interpolation
- Distributed computing
- Task-focused module communication
- Mismatched grids & zooming
- Controls

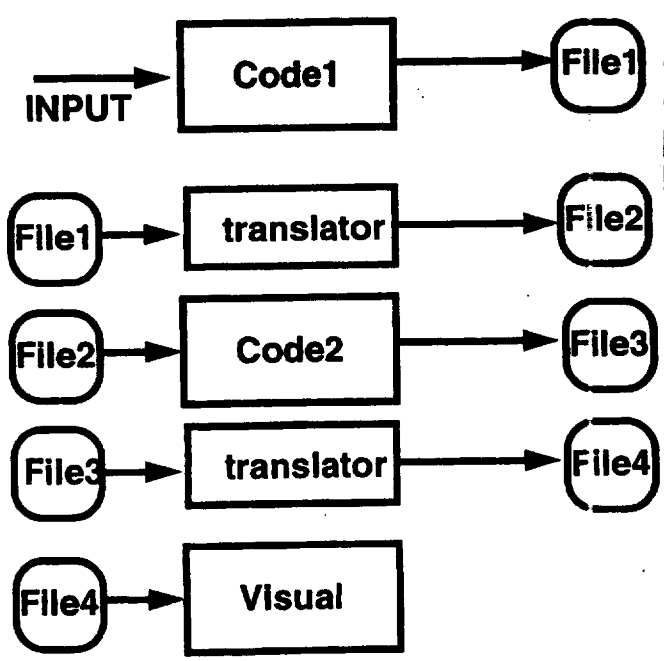
VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS

Issues with interfacing & interpolation:

- **Function matching ("pretty picture")**
- **Conservativeness & consistency**
- **Speed of the interpolation**
- **Generality (structured & unstructured meshes, 2d vs. 3d interfaces, etc.)**
- **Ease of use (for application programmer & end user)**
- **Time synchronization**

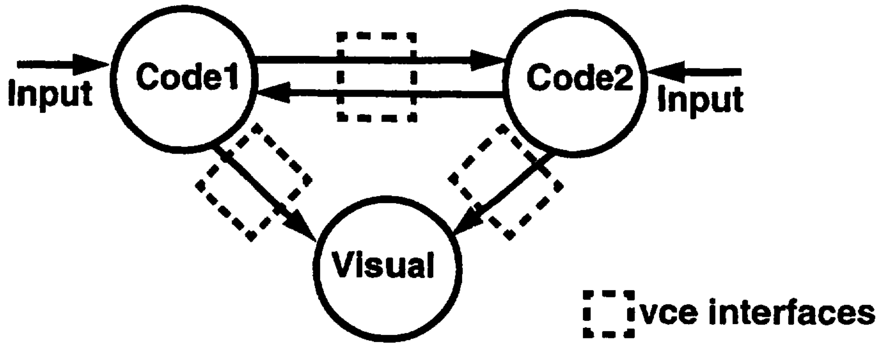
VCE INTERFACING TECHNOLOGY



File-based code to code coupling **places a heavy burden on the user.**

Communication between the codes is often unidirectional and is typically manual

VCE INTERFACING TECHNOLOGY



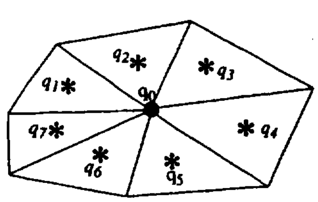
- *VCE interfaces* take the burden of code-code communication
- Communication is multidirectional, when desired
- Communication is fast because is occurs in object space, rather than on disk

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Laplacian Interpolation

CFDRC's Laplacian Interpolation algorithms provide excellent matching of analytical functions. The basic algorithm is as follows (described for center-based codes):

- Face centered values of a given function are interpolated to the nodes using a Laplacian operator in the node normal plane:



$$L(q_0) = \sum_{i=1}^{n} w_i(q_i - q_0)$$

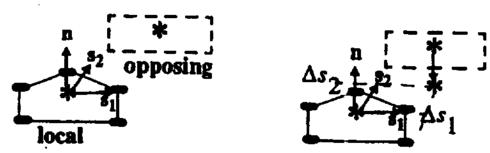
(weights are determined by minimizing a cost function w.r.t the above equation--equivalent to a multi-dimensional least squares algorithm)

- Nodal values are used to obtain a gradient (also using a multi-dimensional least squares fit)

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Laplacian Interpolation (cont)

- The nearest 'local face' is found for each opposing face (ADT search algorithm)
- The opposing face is projected into the local face normal plane
- Opposing interface face centered values are interpolated using the local face center value & gradient along with the opposing projected face centroid

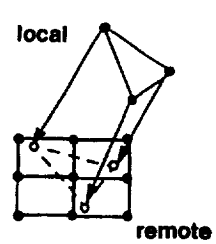


$$q_{opposing} = q_{local} + \nabla q \bullet \{\Delta s_1, \Delta s_2\}$$

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS
Flux-conserving interpolation

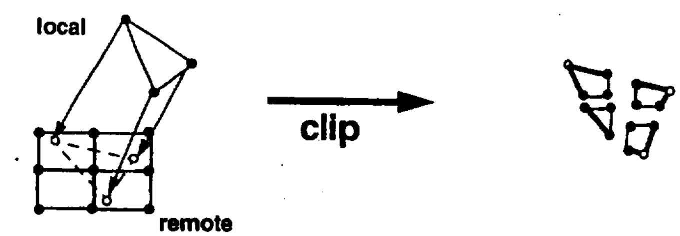
- All nodes from the 'local' interface are then projected into the 'opposing' interface. This assures that all of the local geometry will reside within the remote interface.



VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS
Flux-conserving interpolation

- The Sutherland-Hodgman Clipping algorithm is used to determine the intersection geometry between each 'local' face and multiple remote faces (all computations are performed in the two dimensional space defined by the 'local' face normal plane)



VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS
Flux-conserving interpolation

- The local face flux is divided out to each of the opposing faces by evaluating the value of flux (local face value + Laplacian gradients) at each of the clipped centroids.

- This process is repeated for all local faces

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS
Flux-conserving interpolation

Relevant features of CFDRC's flux-conserving Interpolation method:

- **The division of flux between a given local face and multiple opposing faces is determined using geometrical clipping of opposing faces (Sutherland-Hodgman Clipping algorithm)**
- **A high order distribution of flux is obtained using the Laplacian Interpolation algorithm**
- **The interpolation is fast (clipping & weighting functions are only computed once)**
- **All local flux is projected into the opposing interface (100% flux conserving)**

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation

Definitions:

- **Conservative: the sum of all forces and moments on the fluid interface is equivalent to the sum of all forces and moments on the structure interface**

$$\sum_{\text{fluid faces}} \vec{F}_{\text{fluid}} = \sum_{\text{solid nodes}} \vec{F}_{\text{solid}}$$

$$\sum_{\text{fluid faces}} \vec{M}_{\text{fluid}} = \sum_{\text{solid nodes}} \vec{M}_{\text{solid}}$$

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)

Definitions (cont):

- **Consistent: the virtual work performed by the solid interface is equivalent to the virtual work performed by the fluid interface**

$$\sum_{\text{fluid faces}} W_{\text{fluid}} = \sum_{\text{solid nodes}} W_{\text{solid}}$$

$$W_{\text{fluid}} = \vec{F}_{\text{fluid}} \bullet \Delta\overrightarrow{Cen}$$

$$\Delta\overrightarrow{Cen} = \{x_{\text{cen, n}} - x_{\text{cen, o}}, y_{\text{cen, n}} - y_{\text{cen, o}}, z_{\text{cen, n}} - z_{\text{cen, o}}\}$$

$$W_{\text{solid}} = \vec{F}_{\text{solid}} \bullet \Delta\vec{x} + \vec{M}_{solid} \bullet \vec{\omega}_{\text{solid}}$$

the above equalities apply *ONLY* to degrees of freedom supported by the structures code

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)

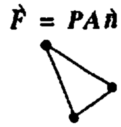
CFDRC's fluid-structure interpolation method is based upon 3 guiding principles:

1) Interpolation should be conservative and consistent
2) The fluid face pressure is the only correct pressure
3) The structural nodal deformation is the only correct deformation

These guiding principles dictate a rigorous method for f-s interface interpolation.

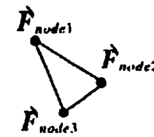
VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)



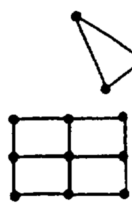
fluid face

The force on a given fluid face is determined by pressure, area, and the face normal. This must be interpolated to the fluid nodes (how?)



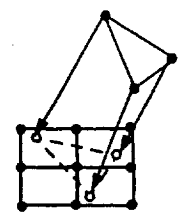
fluid face

The only correct pressure is the fluid face pressure.



solid grid

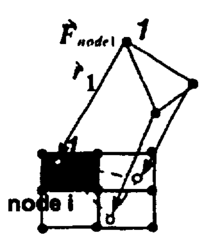
If the grids are mismatched, the fluid nodes must be projected to the nearest solid face



The only correct deformation is the solid nodal deformation

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)



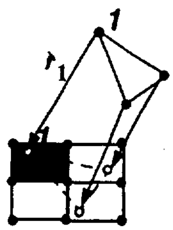
Fluid nodal forces are translated to the solid nodes using finite element shape functions (note that projection of fluid nodes introduces a moment on the solid grid)

$$\vec{F}_{solid\ node\ i} = \vec{F}_{fluid\ node\ 1} N_i(\xi_{fluid,\ 1}, \eta_{fluid,\ 1})$$

$$\vec{M}_{solid\ node\ i} = (\vec{r}_1 \times \vec{F}_{fluid\ node\ 1}) N_i(\xi_{fluid,\ 1}, \eta_{fluid,\ 1})$$

VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)



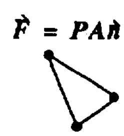
The remaining parameters are determined by enforcing consistency. For example, finite element shape functions give the deflection of fluid node 1 as:

$$\vec{\Delta x}_{\text{fluid},1} = \sum_{\text{solid nodes}} N_i(\xi_{\text{fluid},1}, \eta_{\text{fluid},1})(\vec{\Delta x}_{\text{solid},i} + \vec{r}_1 \times \vec{\omega}_{\text{solid},i})$$

The interpolation must be conservative and consistent

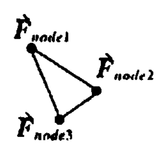
VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)



fluid face

The final step is to determine the method for interpolating from fluid face pressures to fluid node forces. For consistency, we must find nodal weighting factors which result in an accurate approximation of the new face centroid position. The Laplacian interpolation provides an excellent approximation:



fluid face

$$\vec{\Delta x}_{cen} \equiv \sum_{\text{fluid nodes}} w_i \vec{\Delta x}_{\text{node } i}$$

where w_i = Laplacian node to center weight

$$\therefore \vec{F}_{\text{node } i} = w_i \vec{F}_{\text{cen}}$$

The interpolation must be conservative and consistent

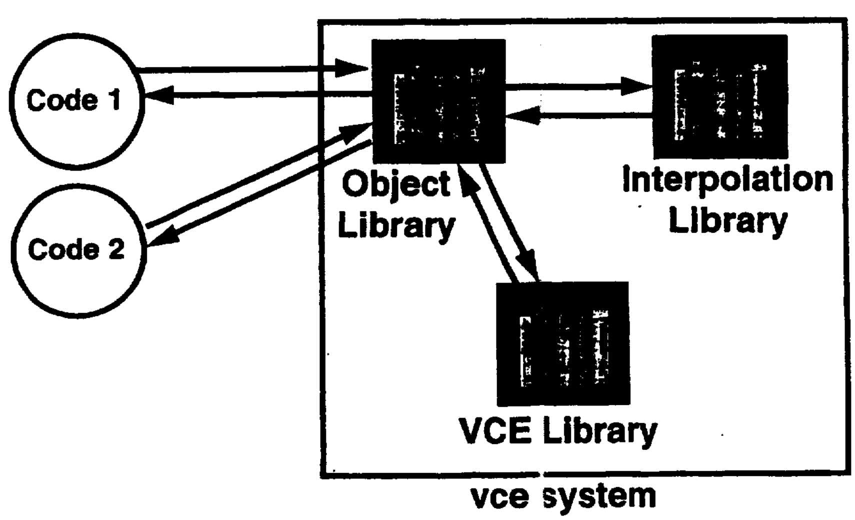
VCE INTERFACING TECHNOLOGY

INTERPOLATION METHODS:
Fluid-Structure Consistent Interpolation (cont)

Relevant features of the CFDRC consistent interpolation method:
- 100% conservative and consistent
- Solid forces are directly projected from the face pressure
- Fluid nodal deflections are directly projected from the solid nodal deflections
- All grid alignment is automatic (accomplished by using fast geometrical searching algorithms)
- Unstructured and structured grids are supported
- Fast searches (ADT algorithm)

VCE INTERFACING TECHNOLOGY

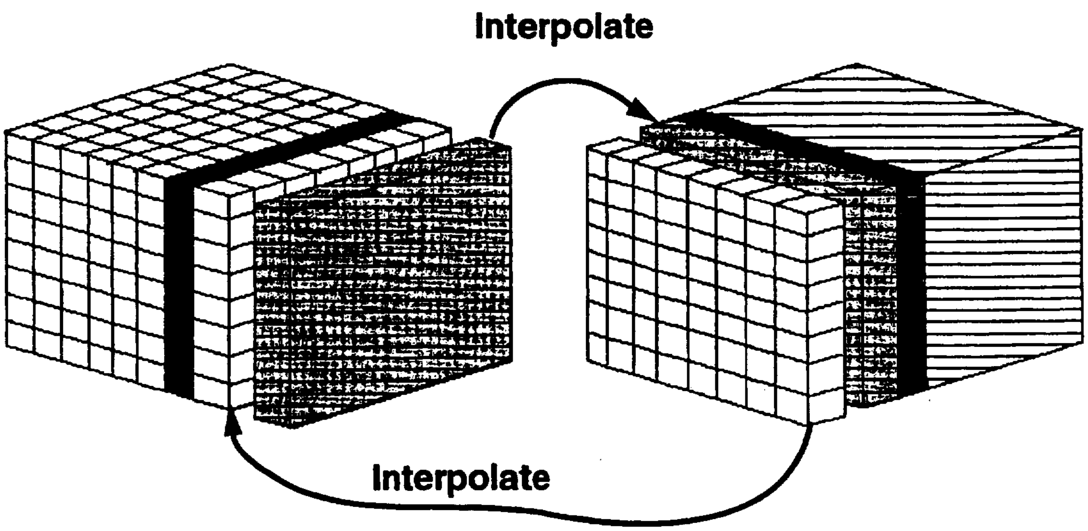
Interfacing in the VCE environment



All interactions between the vce system and the application code are executed through simple access functions--interpolation, conversions, etc. are automatic

VCE INTERFACING TECHNOLOGY

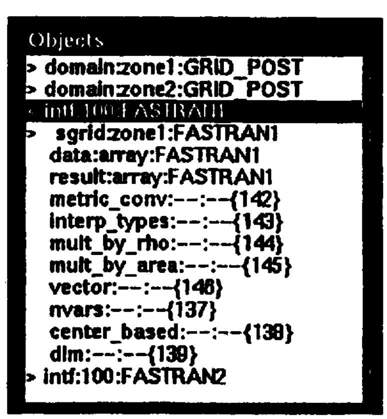
Interfacing in the VCE environment



Interface creation

VCE INTERFACING TECHNOLOGY

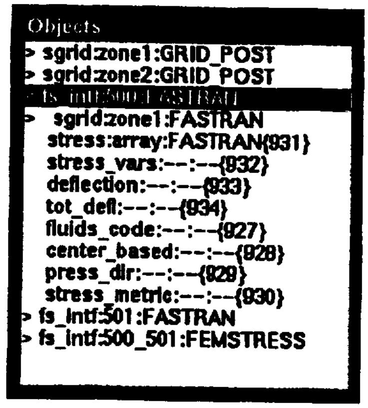
Interfacing in the VCE environment



Fluid-Fluid Interface

VCE INTERFACING TECHNOLOGY

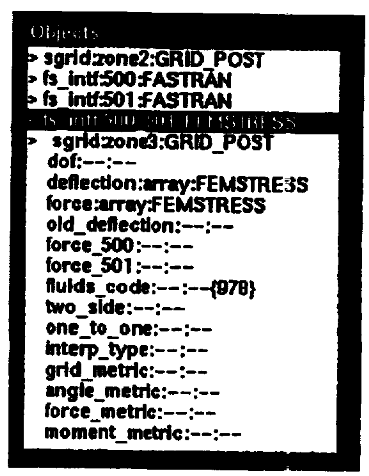
Interfacing in the VCE environment



Fluid-Structure Interface (fluid)

VCE INTERFACING TECHNOLOGY

Interfacing in the VCE environment



Fluid-Structure Interface (solid)

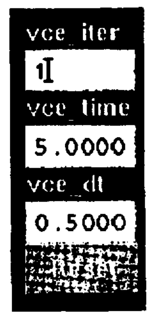
VCE INTERFACING TECHNOLOGY

Interfacing in the VCE environment

- **Temporal synchronization is achieved using global parameters: vce_iter, vce_time, vce_dt**
- **Each application can be queried for time step limitations (due to stability, temporal resolution, etc)**

```
# Get max allowable dt
dt1 = FASTRAN->get_max_dt()
dt2 = FEMSTRESS->get_max_dt()
vce_dt = dt1
if ( dt2 < dt1 )
   vce_dt = dt2
endif

# solve fastran and transfer data
FASTRAN->cycle()
transfer |fs_intf:500:FEMSTRESS|, |fs_

# solve femstress and transfer data
```

ipt Save Load Clear



VCE INTERFACING TECHNOLOGY

Interfacing in the VCE environment

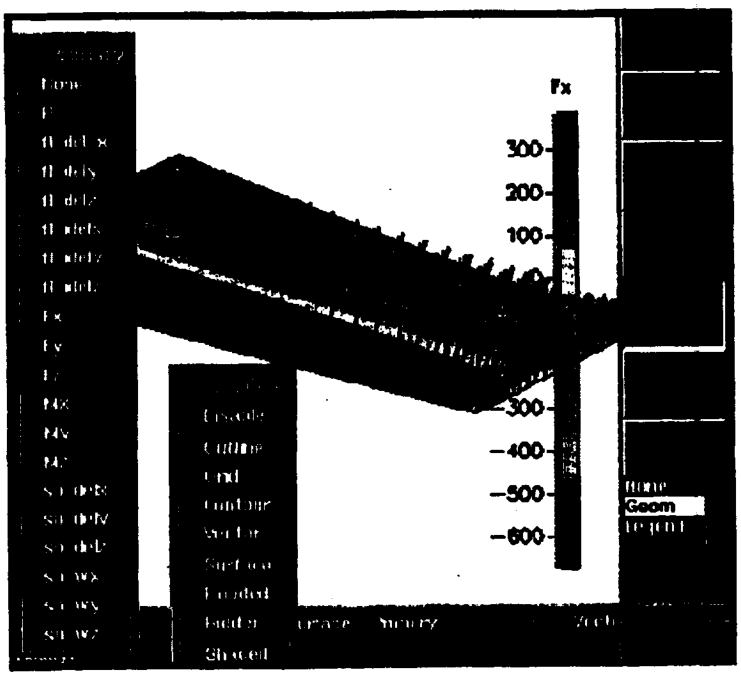
Complex functionality such as exciting a particular mode shape can be easily integrated into the VCE environment

```
        # Apply an impulse to the Wing

endif

# Start the transient flutter loop
do vce_iter = 1, 500

        # Get max allowable dt
        dt1 = FASTRAN->get_max_dt()
        dt2 = FEMSTRESS->get_max_dt()
        vce_dt = dt1
        if ( dt2 < dt1 )
           vce_dt = dt2
        endif
```

Script Save

VCE INTERFACING TECHNOLOGY

Interfacing in the VCE environment



Visualizing fluid-structure interfaces is as simple as transferring the interface to CFD-VIEW

VCE INTERFACING TECHNOLOGY

Interfacing in the VCE environment

Summary of interfacing in the VCE environment:

- ***Simple access functions*** **allow the application program to define interfaces**
- **FS** ***Interpolation*** **methods are fully** ***conservative and consistent***
- ***Temporal synchronization*** **is automatic when the global time parameters are used.**
- ***Automated conversions*** **are available for precision and metric units**
- **Any** ***user-specified functions*** **such as mode shape excitation are easily integrated into the environment**
- ***Visualization*** **is achieved automatically by sending the interfaces to CFD-VIEW**

CFD-DTF DATA TRANSFER FACILITY

William J. Coirier
CFD Research Corporation
Huntsville, Alabama

CFD-DTF: Relation to VCE

- DTF and VCE are Complementary Technologies

- Both Integrate Different Applications Together into a Single "System"

- VCE:
  - Dynamic, coordinated by scripting language
  - Message Passing of Data and what to do with it
  - Concurrent

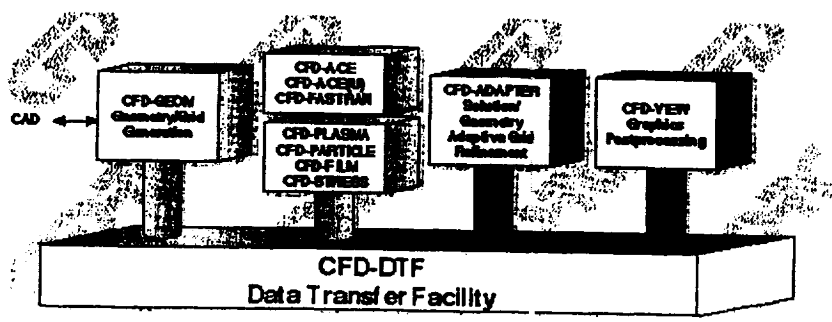
- DTF:
  - Static
  - File-based Sharing of Data
  - Sequential

DTF: Data Transfer Facility

- Independent Evaluation by GTRI of Common File Formats Indicates DTF "years ahead" of all the rest.

- Chosen by Wright Labs/FIMC to integrate their CFD codes:
  - Cobalt90, Mercury, TEAM, FASIT

- Used on a daily basis at CFDRC:



DTF: Simulations, Zones . . .

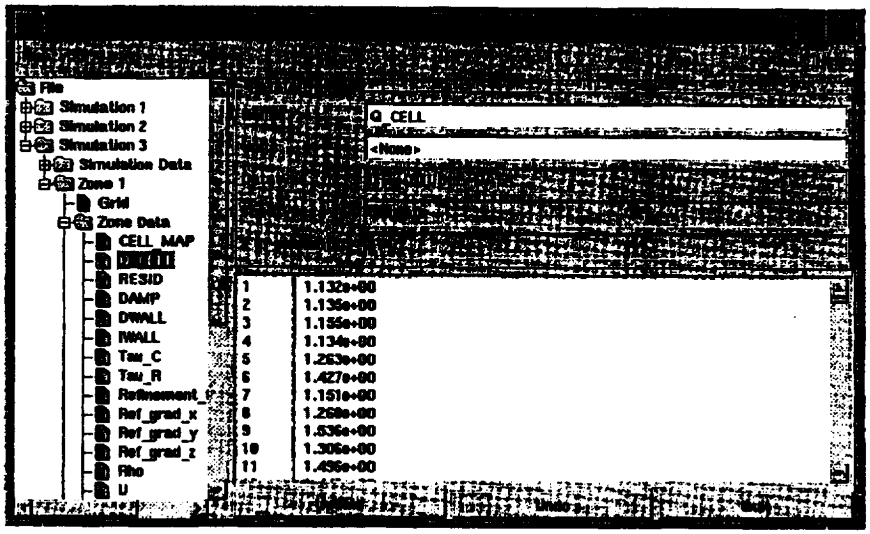
- Simulation used to store:
  - Multiple Time Snapshots of Solution (unsteady)
  - Restart Data, or Different Flow Conditions (steady)
  - Adaptively-refined grid levels
  - Simulation Data: General data arrays of ints, reals, strings
  - Zones
- Simulations may be linked to one another (share data)

- Zones may be Structured, Unstructured, Polyhedral, Virtual

- Zones are fully defined:
  - Boundary Condition Data and Associativity
  - Volume Condition Data and Associativity
  - All Restart, Visualization, Post-Processing Data (Zone Data)

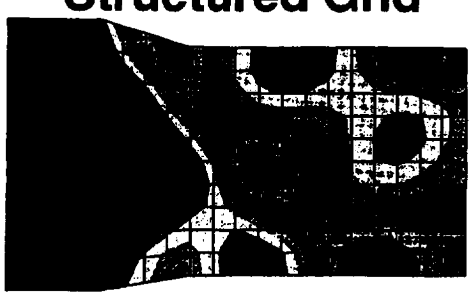
DTF-GUI

- GUI to view/modify data in DTF file
- Still Under Development



Cobalt90_DTF: Computed Mach

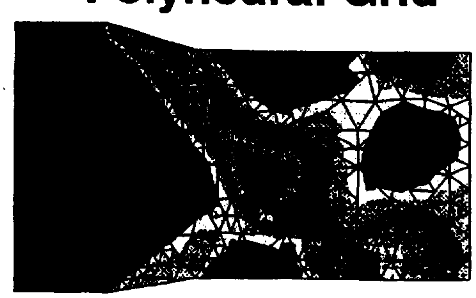
Structured Grid



Unstructured Grid

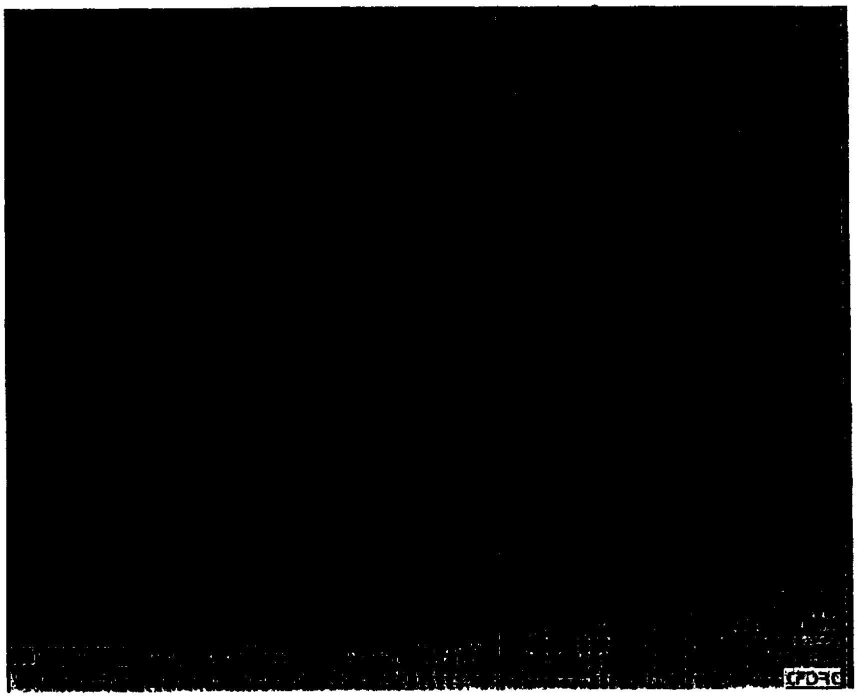
“Hybrid” Grid

Polyhedral Grid



ACE(+): Hemodialysis Simulation

- **Multi-block Grid:Single-Zone Unstructured Solver**



DTF && VCE: Recommendations

- **Incorporate VCE Object Library concepts and DTF File Access Mechanisms into a Generalized Object Library**
    - Provide file-based analogy of VCE Object Sends/Receives
    - Permit VCE System Restarts, Images
    - Support "Design-your-own" Data Structures
- **DTF is becoming a standard for CFD**
- **VCE can leverage this by incorporating DTF technology**

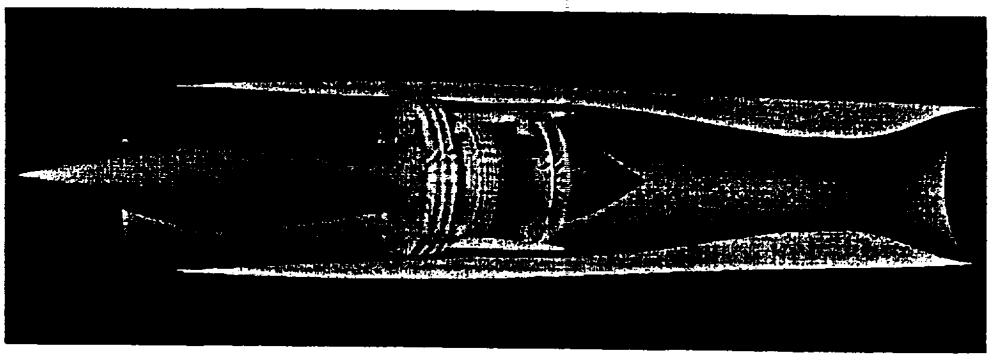
VCE APPLICATIONS AT NASA LEWIS RESEARCH CENTER
Inlet-Engine Simulation

Gary Cole
NASA Lewis Research Center
Cleveland, Ohio

VCE APPLICATIONS AT NASA LEWIS
Inlet-Engine Simulation

- Technical Motivation & Objectives
- Description of Application - Sajben Compressor Experiment
- Approach
- Why VCE and Objectives
- Issues Affecting Integration of Codes with VCE
- Validation of Code Coupling
- Preliminary (Uncoupled) Results
- Status
- Conclusions

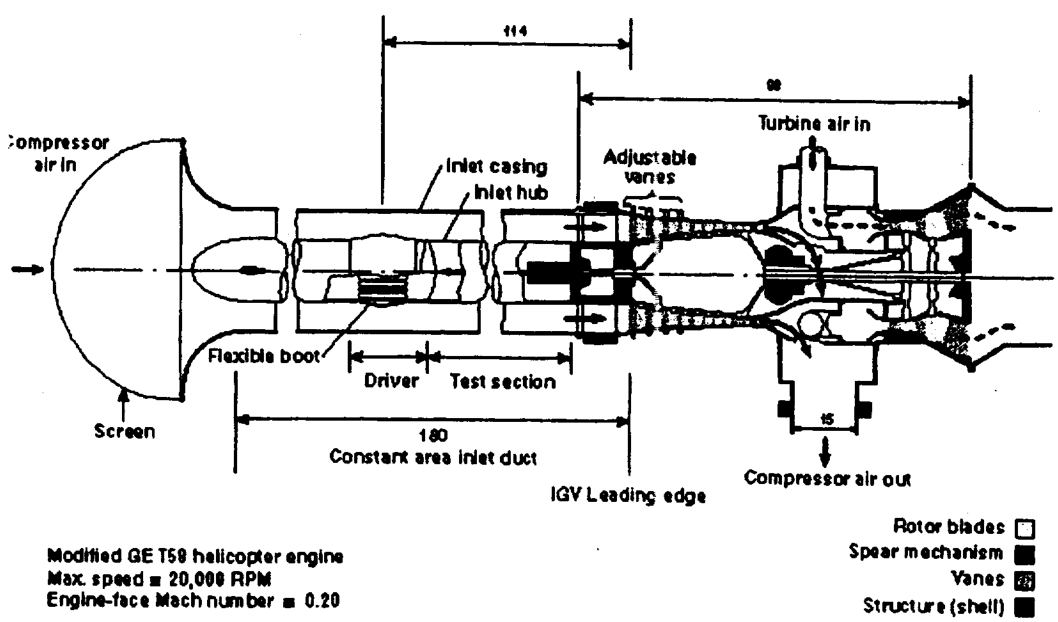
Motivating Application
(HSCT-type propulsion system)



- Modern supersonic-inlet-engines are susceptible to an unstart due to flow field perturbations
- An unstart is unacceptable on a commercial (passenger) aircraft
- CFD simulations are used to design and evaluate inlet controls to prevent unstart
- Shock dynamics and tolerance to unstart are highly dependent on the compressor-face boundary condition (CFBC)

Objectives:
- Gain a better understanding of inlet-engine interaction physics
- Formulate a more realistic CFBC for use with CFD simulations of inlets

Compressor-Face Boundary Condition Experiment
(Sajben and Freund)



Selected Application - Sajben Compressor Experiment

- Doable CFD problem that includes wheel speed, stagger angle, solidity, and Mach number as variables
- Problem of interest but requires coupled-code solution
- Experiment produces rapid, well-defined incident and reflected pulses
- Experimental data provides means of validating CFD solutions

APPROACH

- Use existing time-accurate codes from NPSS suite - NPARC for inlet (duct with collapsing bump) and ADPAC for turbomachinery
- Initially simulate only first-stage rotor of turbomachinery (T58)
- Use swirl component to simulate effect of inlet guide vane (IGV)
- Explore effect of rotor exit boundary condition on inlet dynamics
- Validate coupling strategy with test cases
- Perform *coupled* unsteady inlet-engine simulation

Why VCE and Objectives

- Suitable application for VCE because of requirement for coupled-code solution
- Provides general & distributed framework for code coupling
- Provides more flexible control of codes than direct coupling

Objectives:

- Incorporate NPARC and ADPAC codes into VCE
- Evaluate process for incorporating codes into VCE
- Demonstrate code coupling under VCE
- Evaluate VCE as a tool for coupling codes

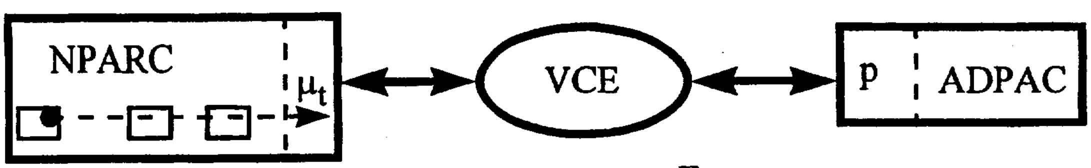
VCE-Specific Code Integration Issues

- Restructure for script (external) control
- Add hooks for script commands
- Parallel codes must be compatible with PVM
- Parallel codes must be able to be spawned by PVM
- Potentially support grid input via CFD-GEOM

Code-Coupling-Specific Integration Issues

$\rho, \rho u, \rho v, \rho w, E \mid \mu_t, k, \varepsilon$

$\rho, \rho u, \rho v, \rho w, E, \mu_t, p \mid k, \varepsilon$

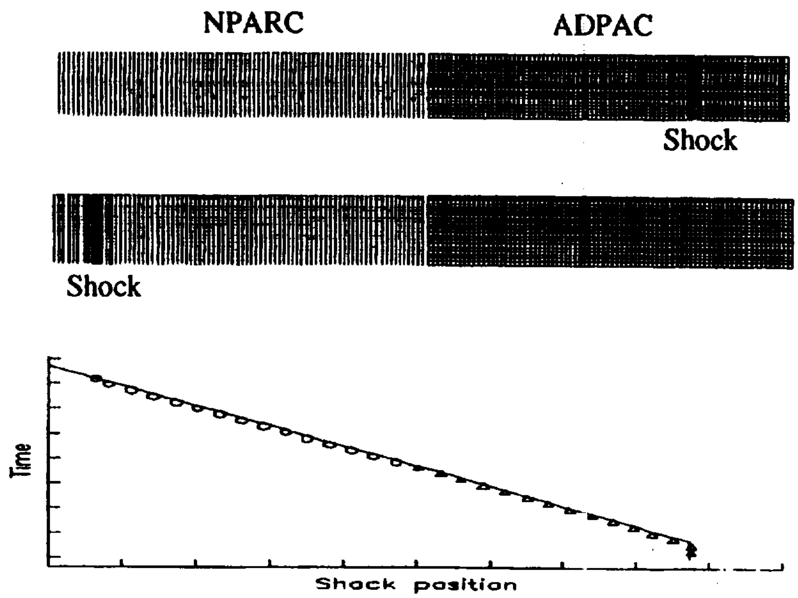


$\rho, \rho u, \rho v, \rho w, E, \mu_t$

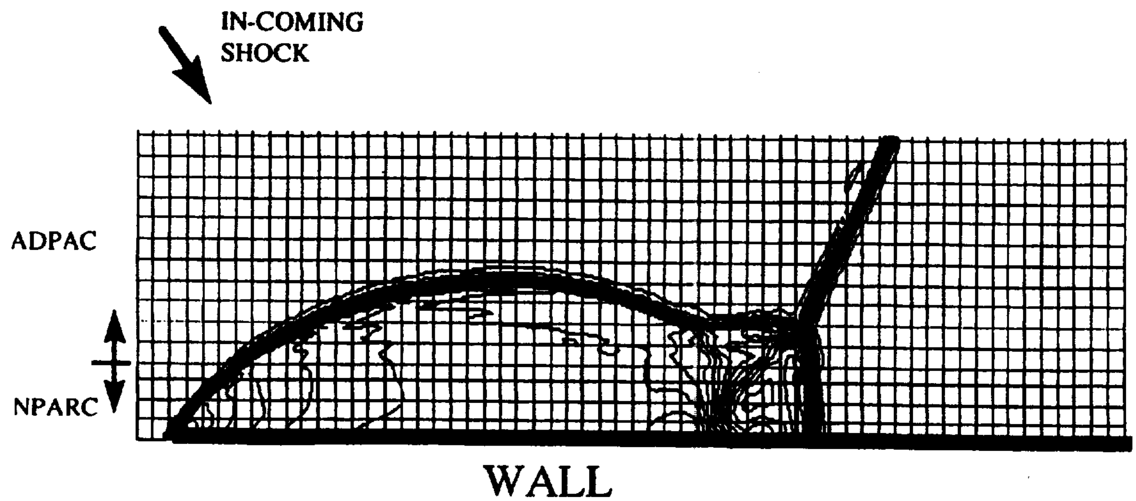
Code-Coupling-Specific Integration Issues (Cont'd)

- NPARC blocks exchange data once per time step - ADPAC blocks exchange data every Runge-Kutta stage
- Dissipation at coupled boundaries calculated differently from other ADPAC internal block boundaries
- Compatibility of ADPAC multi-grid algorithm may be an issue (not investigated)

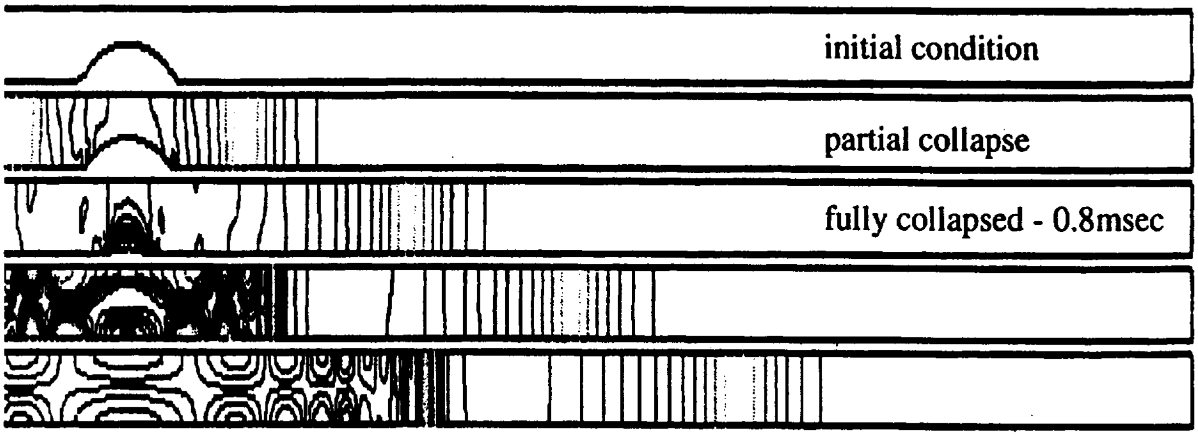
Coupled ADPAC-NPARC Unsteady Simulation
(Unsteady Normal Shock Validation Case)



Coupled ADPAC-NPARC Unsteady Simulation
(Double Mach Reflection Validation Case)

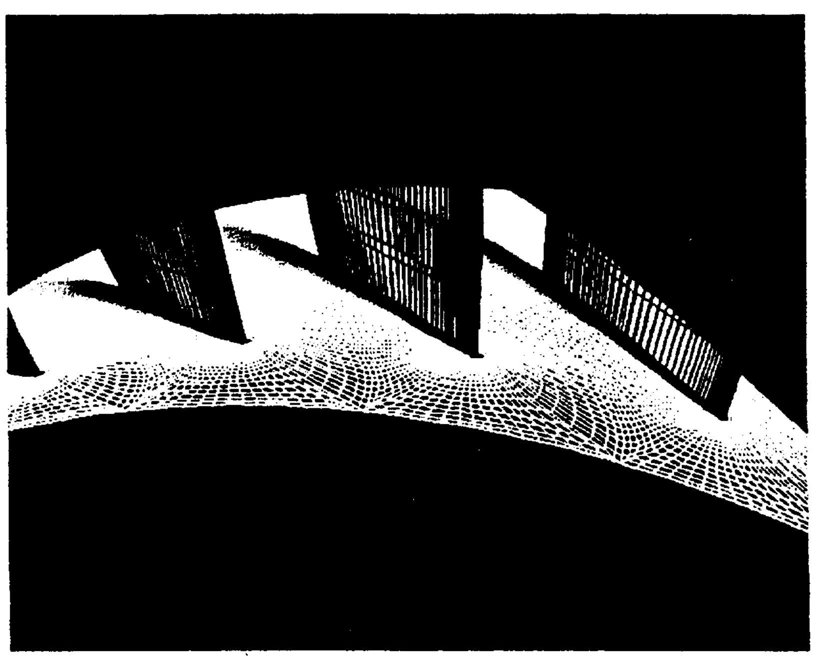


NPARC Simulation of Duct with Collapsing Bump



ADPAC Steady Simulation - T58 First Stage Rotor

(calculated mass flow matches Sajben experiment)



Status

Completed -

• T58 flow path and first stage rotor geometry defined (GE drawings)

• 3D grid generated for T58

• NPARC & ADPAC incorporated into VCE

• NPARC to ADPAC coupling validated by test cases
• Steady ADPAC solution obtained for T58 first stage rotor

• Axisymmetric grid and NPARC unsteady solution obtained for duct with collapsing bump

Remaining -
• Obtain 3D grid and NPARC unsteady solution for duct with collapsing bump
• Obtain 3D NPARC to ADPAC coupled solution for duct with bump / T58
• Update NPARC and ADPAC for latest release of VCE
• Obtain Axi-NPARC to ADPAC coupled solution (2D<->3D I/F object req'd)
• Obtain Parallel NPARC to Parallel ADPAC coupled solution
• Get back to investigating the physics

Conclusions

- VCE offers a convenient way to couple NPARC and ADPAC while allowing for sufficient control of the individual codes
- Although incorporation of codes into VCE is painful the first time, once completed, a robust coupled simulation capability is possible
- The newest release of VCE looks promising in terms of improvements (to be evaluated)
- The full potential of VCE as a coupling mechanism is unrealizable without well defined interface standards

VCE APPLICATION TO NATIONAL COMBUSTION CODE (NCC)

Nan-Suey Liu
NASA Lewis Research Center
Cleveland, Ohio

Role Of VCE IN NCC

To provide on-the-flight data communications among the modules during the simulation

For example: Co-processing the flow solver and mesh adaptor

Co-processing certain modules to facilitate steady-state calculation

Co-processing results visulization

To provide coupling between NCC and other NPSS component codes

Current Status

A prototype of the integrated system consisting of VCE 3.4, CFD-DTF, CFD-GUI, CFD-VIEW, CFD-GEOM, and the baseline flow solver of the NCC, i.e., CORSAIR, has been demonstrated.

Further review and evaluation of this prototype system, taking into considerations of (i) the latest versions of all components; and (ii) the customization for NCC requirements, will be conducted.

Question

Any plan of hanging DTF, GUI, pV3_GOLD into VCE 3.6 ?

VCE APPLICATIONS AT ALLIEDSIGNAL ENGINES

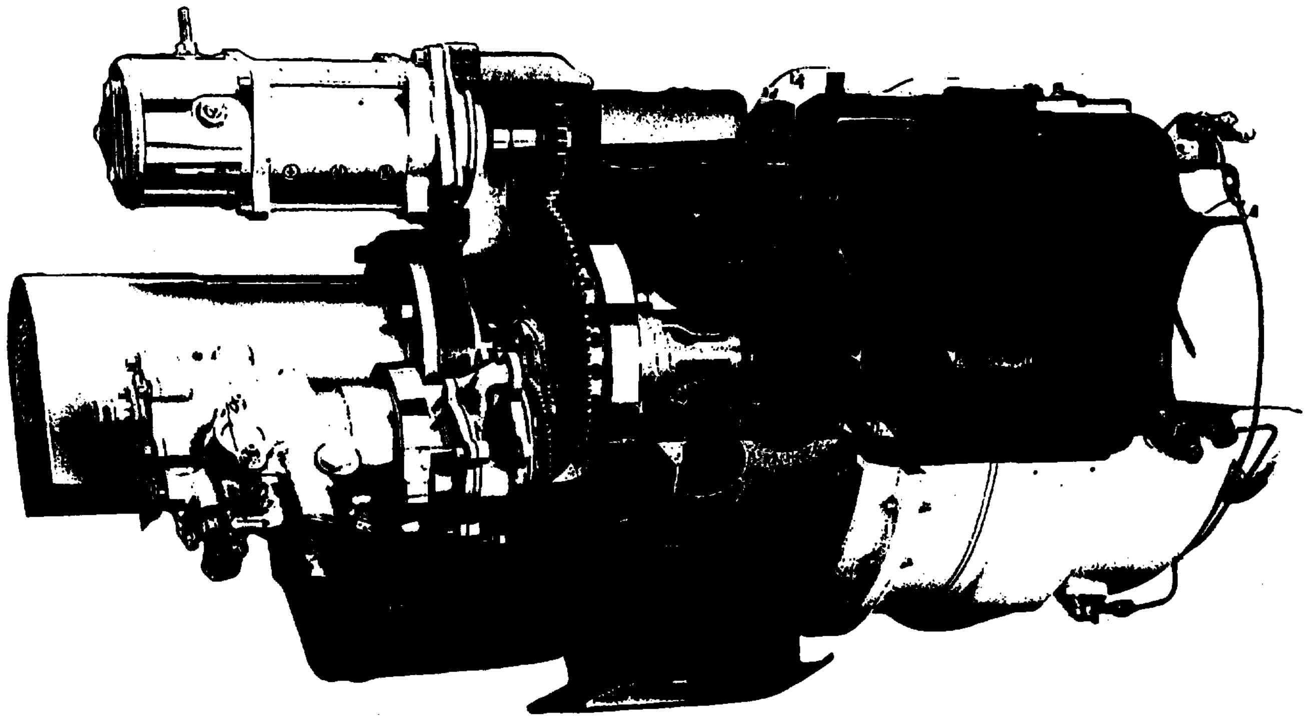
Wolfgang Sandel
AlliedSignal Engines
Phoenix, Arizona

Multi-Disciplinary Analysis

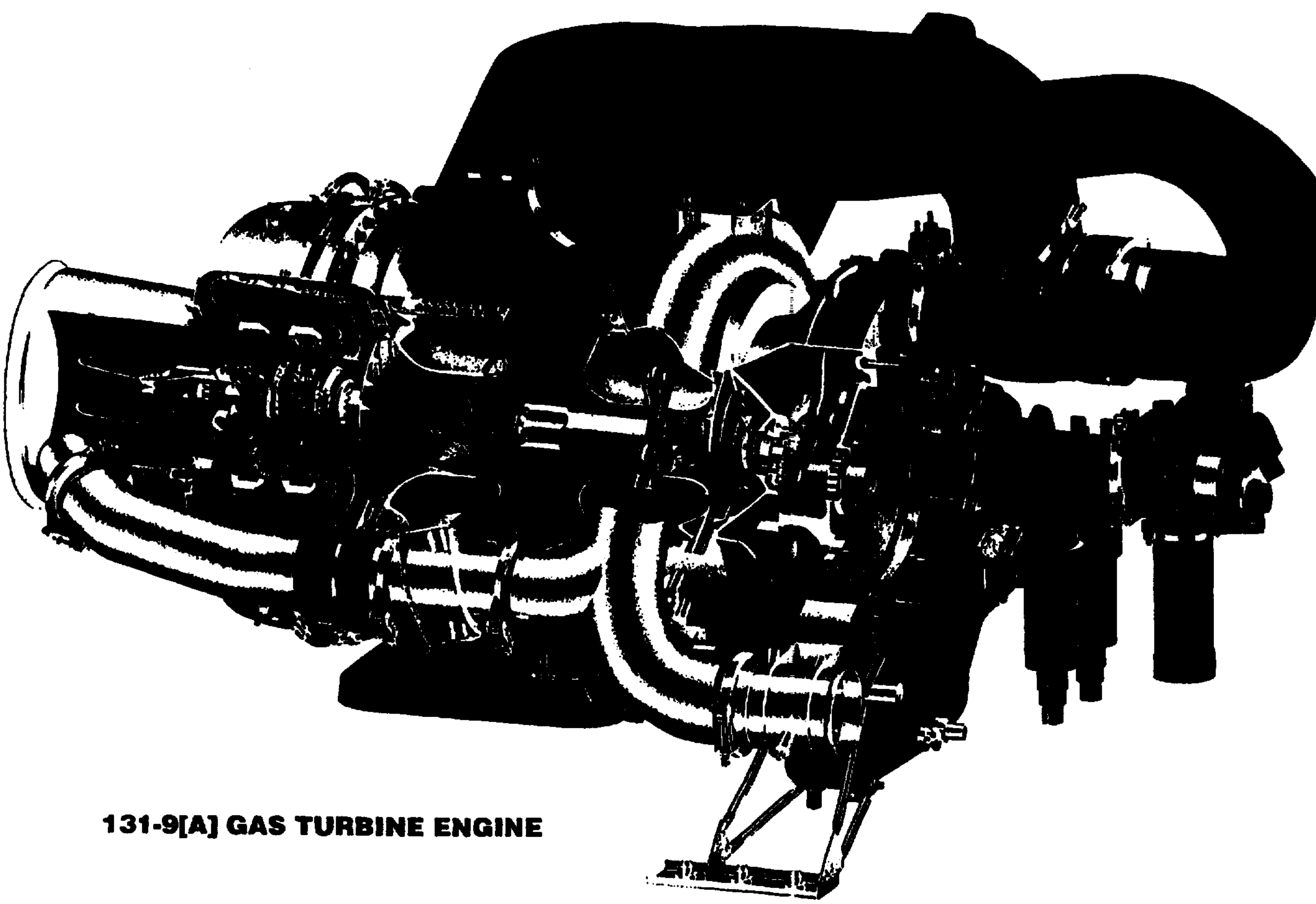
- Sequential analysis process (fluid-structural or fluid-thermal) meets needs in most cases.
- Barriers to VCE use for strongly coupled simulations:
  - commercial codes are involved
  - other simulation aspects take precedence
  - complexity/computational constraints force model simplifications incompatible with 3-D coupling
- Setup of 3-D data transfer for sequentially connected analyses is tedious across mesh types, VCE has built-in solution

Inter-Component Simulations

- Multiple opportunities exist:
  - Flow simulation with coupled inlet and fan analysis
  - Flow simulation of radial turbine stage that couples vane, rotor and exit diffusor in a mixing plane approach
  - Flow simulation with coupled APU inlet plenum, load and gas generator compressor and cooling flow (involves interfaces across mesh types and dimensions)



36-150 GAS TURBINE ENGINE



131-9[A] GAS TURBINE ENGINE

Design/Analysis Process Streamlining

- VCE features useful to enhance design/analysis processes
  - Scripting and control facilities allow to bundle tools, automate existing process steps
  - Mesh interfacing feature simplifies data transfer tasks
  - New framework-integrated tools (like interfaces between levels of dimensionality and plotting capabilities) would simplify the building of "design workbenches"
  - In some cases simple design steps may be included and a monitored and controlled design/analysis loop be constructed, potentially coupled with optimization tools

VCE benefits turbomachinery design/analysis process in multiple ways

Supports design-by-analysis goal

- multi-disciplinary simulations
- inter-component simulations
- mesh-interfacing tool

Streamlines processes/tools

- scripting facility may be used to automate tasks and bundle tools
- supports data exchange across mesh types
- levels of dimensionality interfacing and graphing would complement existing set of integrated tools

Speed through parallel processing

- once analysis codes are hooked up to VCE, they are also prepared for parallel processing

Faster response for new processes

- software support through framework based features improves flexibility, reduces reaction time to support new processes

Challenges that VCE may face to become accepted standard

Commercial software vendors need to participate in effort

- Industry's design/analysis systems contain mix of commercial codes and codes with source access - VCE simulations should include both.

Ease of use helps acceptance rate

- Simulations and execution suites will be specified by design engineers, not software engineers - interface and script syntax acceptance is important.

Avoid narrow focus

- Recognize VCE's contributions to sequentially coupled analysis approaches and design processes.

Completeness of tool set influences acceptance

- It is a good idea to enhance the built-in tool support with graphics, level of dimensionality interfaces.

Summary

- AE expects VCE to provide improved analysis fidelity through component coupling in specific cases.
- AE finds fewer multi-disciplinary coupling opportunities than initially expected.
- For VCE to become a standard tool in the design/analysis environment, commercial software vendors need to participate.
- VCE's integrated mesh interfacing tools benefit existing sequential multi-disciplinary analysis processes as well.
- AE sees a potential for VCE to support streamlining of existing design/analysis processes and tools.
- VCE enables speed improvements through parallelization.
- AE welcomes the addition of more tools built into the VCE framework as this increases VCE's usefulness.

VCE APPLICATIONS AT PRATT & WHITNEY

David Edwards
Pratt & Whitney
East Hartford, Connecticut

Background

- NASA Projects
  - Affordable High Performance Computing Project
  - NCC

- Objective - utilize VCE for coupling a set of separate CFD simulation solvers to perform one integrated simulation of an engine. Specific modules to integrated include:
  - pV3/pV3-Gold visualization system
  - Pratt & Whitney CFD solvers (NISTAR,NASTAR,CORSAIR)
  - NASA Turbo CFD solver

Integration of pV3-Gold

- Two approaches
  - Integration of pV3 directly into the solver, then integrate solver into VCE
    - Positive - reduction of data transferred over network and memory requirements for system
    - Negative- requires each new applications to be integrated with pV3 before insertion into VCE
  - Develop VCE- pV3 object where information is passed from solvers to VCE-pV3 object. Object is a pV3 client that passes information to pV3 visualization server
    - Positive - once object is developed it can be applied to other VCE modules
    - Negative - data transfer over network increased significantly for transient data, memory requirements increases
- Decision - develop both approaches

pV3-VCE Status

- pV3-VCE object developed for VCE 2.4
  - Tested with multi-block, structured solver (NISTAR)
  - In process of being tested with unstructured solver (CORSAIR)
- VCE 2.4 Limitations
  - Solver time can not be passed to VCE-pV3 object (required for particle tracing)
  - Scripts must be modified for different cases
  - VCED can adsorb all the CPU of a workstation
  - scripting communication is one-way
  - No batch capabilility
  - Double precision floating-point
- pV3-VCE object will be ported to VCE 2.6

CFD Solver Status

- Work initiated on integrating Pratt & Whitney Solvers into VCE 2.4
  - Limitations on bounday objects identified
    - Circumferential Averaging
    - Interpolation between rotating grid/stationary grid
    - Interpolation in cylindrical coordinates versus cartesian cordinates
  - Limitations on running CFD solver in a parallel mode
    - Combining results of parallel solver before passing information along boundary is not efficient
  - VCE 2.4 not acceptable for code coupling
- Decision - wait for VCE version 2.6

Concluding Remarks

- Overall Status - Waiting for VCE 2.6
- Observations
  - VCE is a good concept
  - CFDRC
    - Working relation with NASA/industry should be team approach instead of a customer-vendor relation
  - NASA/Industry needs to be more pro-active with VCE development
    - Design reviews are critical before development. NASA/industry needs to be more involved.

APPLICATIONS OF THE VCE ENVIRONMENT

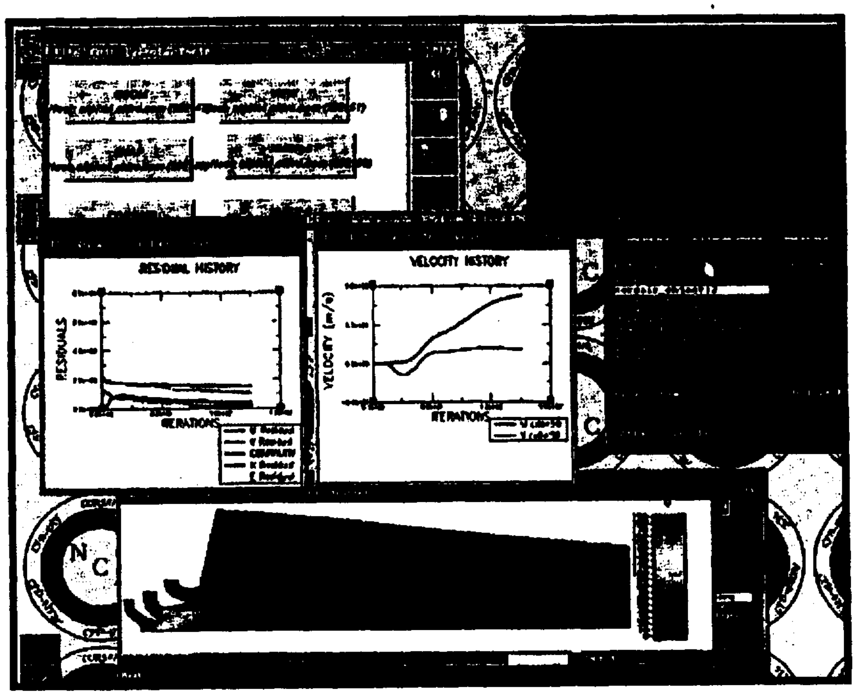
John M. Siegel, Jr
CFD Research Corporation
Huntsville, Alabama

VCE INTERFACING TECHNOLOGY

A variety of vce-enabled utility codes are provided with the integrated VCE system:

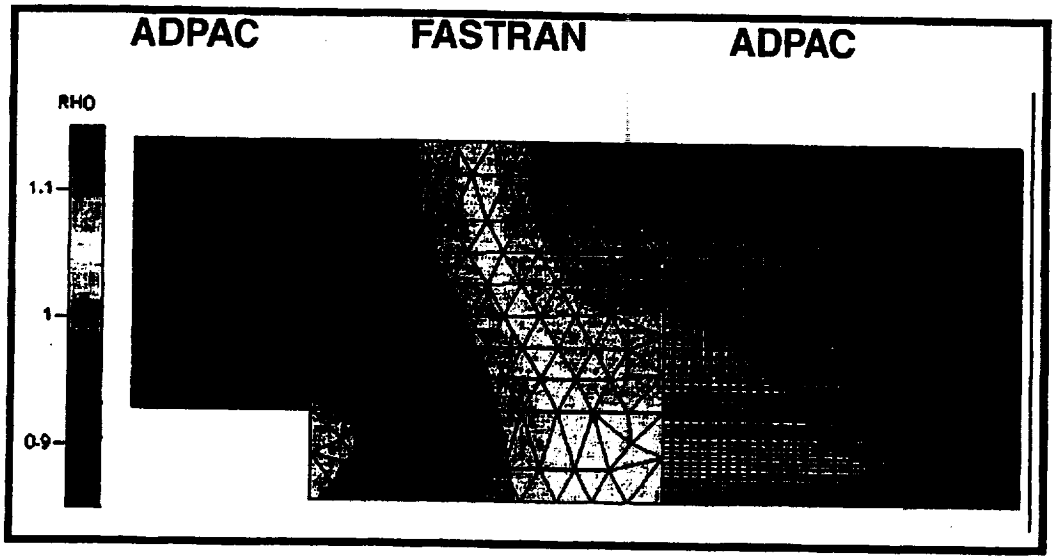
- **CFD-GEOM (geometry modeling & grid generation)**
- **grid-post (a simple routine for posting structured grids)**
- **CFD-VIEW (post-processing/visualization package)**
- **xmgr (xy data plotting software)**
- **grab (an ImageMagick based screen grabber -- used for creating movies)**

VCE INTERFACING TECHNOLOGY



Real time visualization

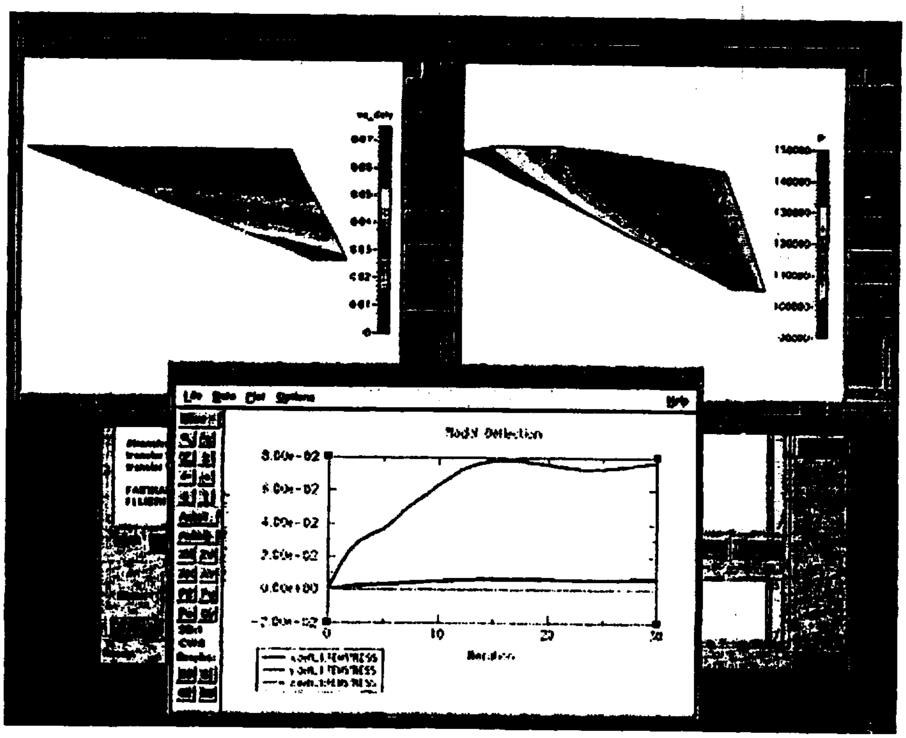
VCE INTERFACING TECHNOLOGY



Code to code coupling, arbitrary grid matching, distributed computing

VCE INTERFACING TECHNOLOGY

Applications



Two-sided fem - fluid interface

Fluid:

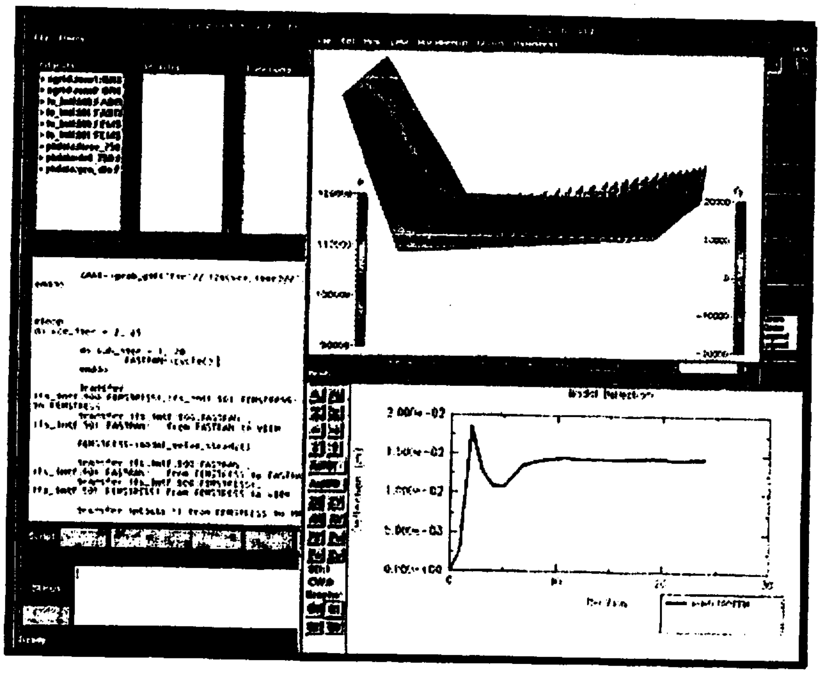
$$M_\infty = 0.58$$
$$AOA = 5 \text{ degrees}$$
$$P = 1.0\text{E5 Pa}$$
$$\rho_\infty = 0.5\frac{kg}{m^3}$$
$$T = 300 \text{ K}$$

Solid:

$$\nu = 0.3$$
$$E = 2\text{E11}\frac{N}{m^3}$$
$$\rho = 500\frac{kg}{m^3}$$
$$: = 0.04 \text{ m}$$

VCE INTERFACING TECHNOLOGY

Applications

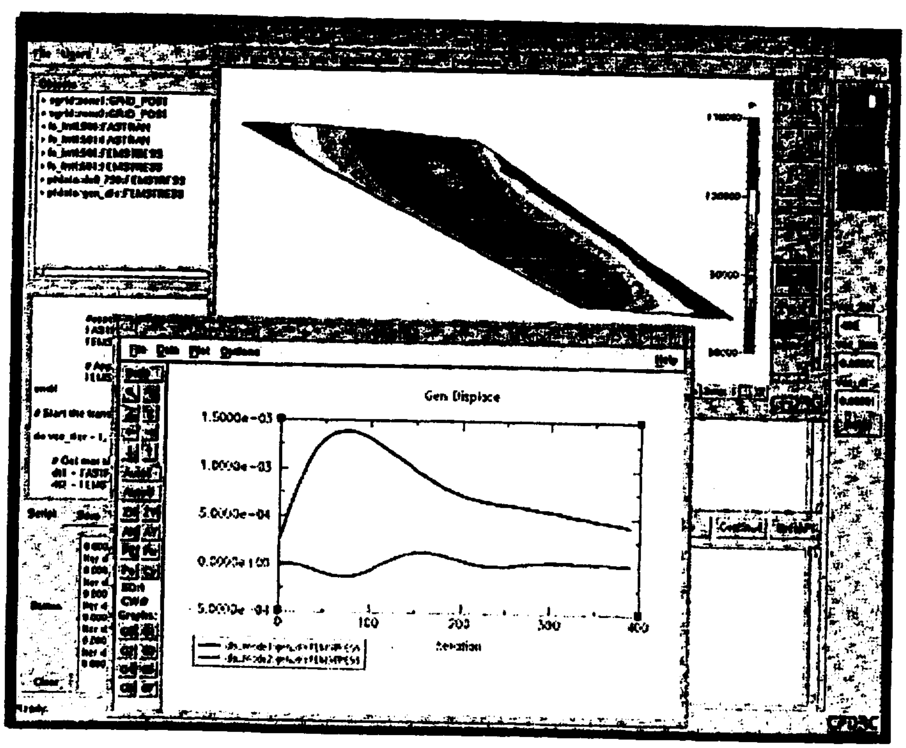


Steady one-to-one modal analysis

$$M_{\infty} = 0.8$$
$$AOA = 1.0 \text{ degrees}$$
$$P = 1.0E5 \text{ Pa}$$
$$\rho_{\infty} = 0.5\frac{kg}{m^3}$$
$$T = 300 \text{ K}$$

VCE INTERFACING TECHNOLOGY

Applications

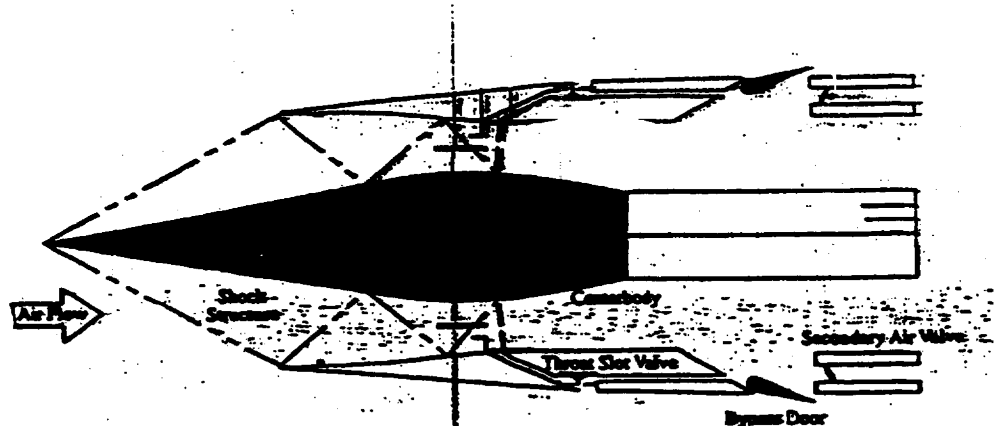


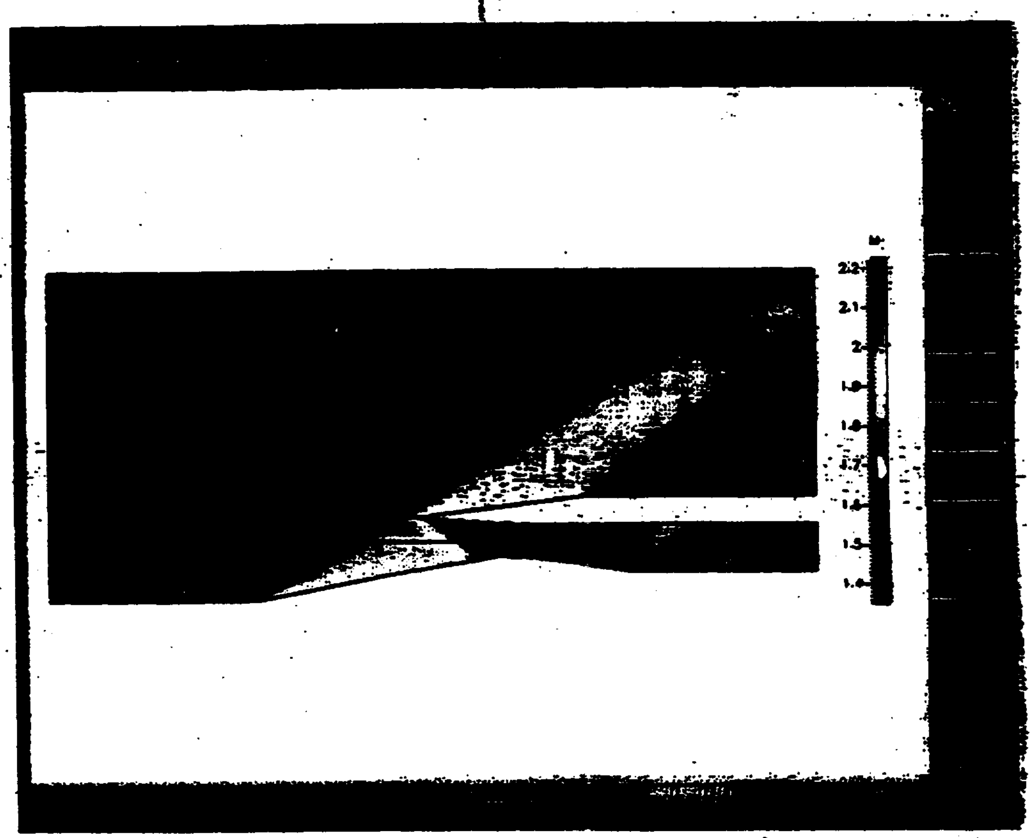
Transient one-to-one modal analysis

$$M_{\infty} = 0.9$$
$$AOA = 0.0 \text{ degrees}$$
$$P = 1.0E5 \text{ Pa}$$
$$\rho_{\infty} = 0.5\frac{kg}{m^3}$$
$$T = 300 \text{ K}$$

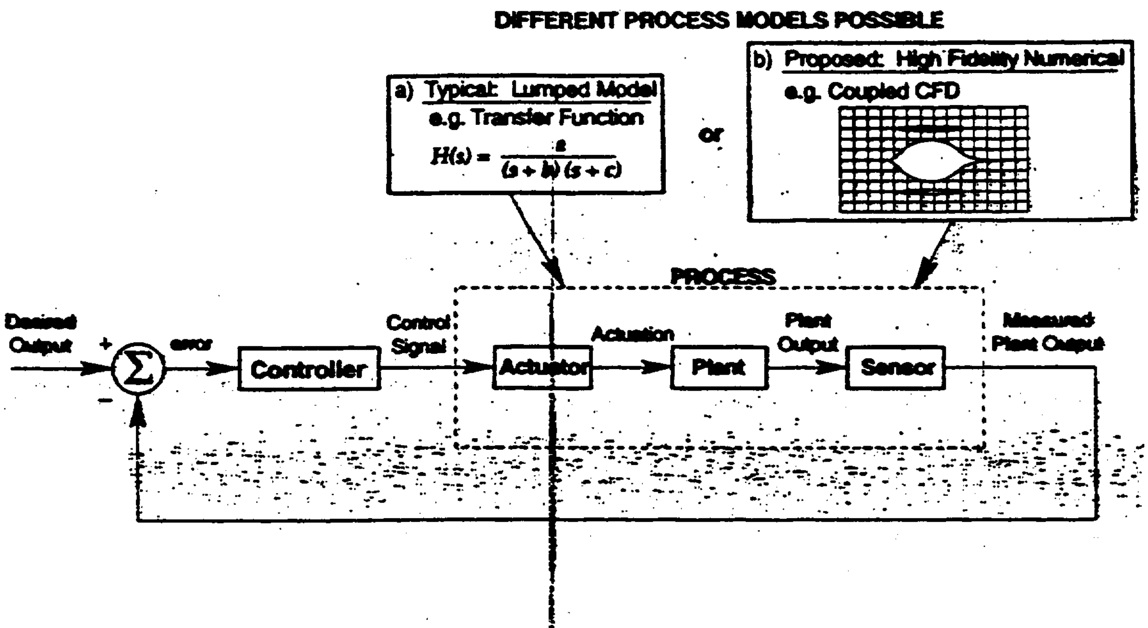
MDICE: CONTROLS INTEGRATION EXAMPLE

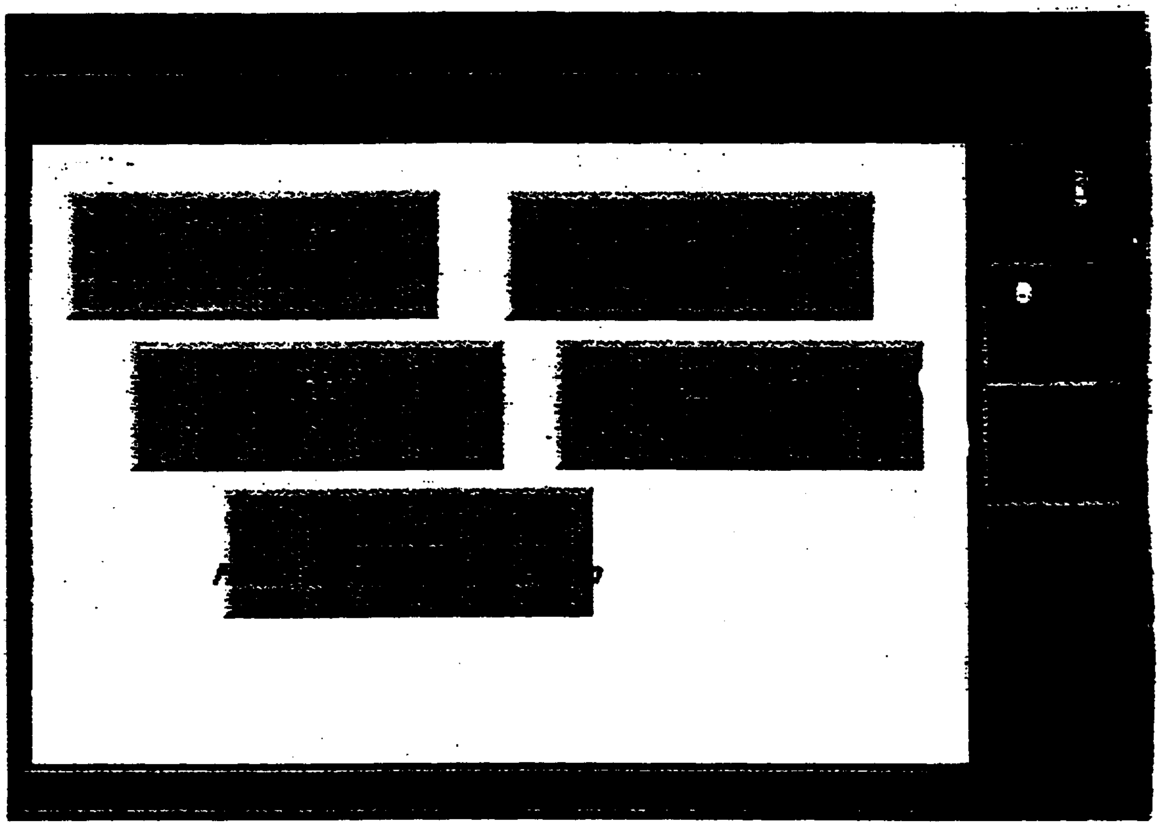
Mixed Compression Translating Centerbody (MCTCB) Supersonic Intel



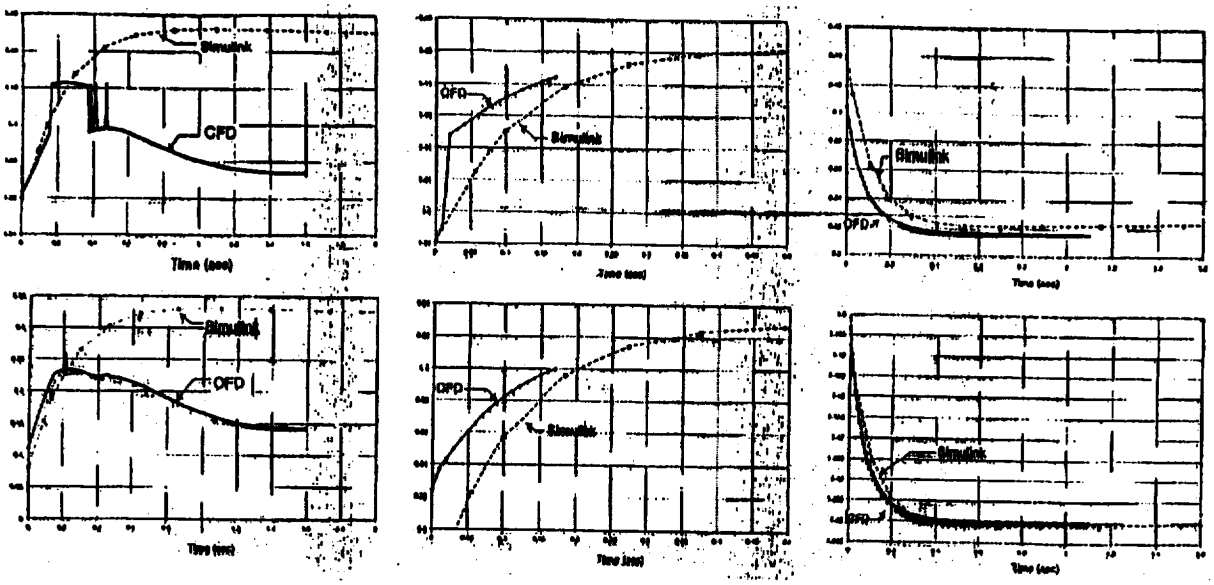


MDICE: CONTROLS INTEGRATION APPROACH





MDICE: CONTROLS RESULT



REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 1998	Conference Publication

4. TITLE AND SUBTITLE

Visual Computing Environment Workshop

5. FUNDING NUMBERS

WU-509-10-31-00

6. AUTHOR(S)

Charles Lawrence, compiler

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E-11279

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA CP—1998-208525

11. SUPPLEMENTARY NOTES

Responsible person, Charles Lawrence, organization code 5900, (216) 433-6048.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category: 61 Distribution: Nonstandard

This publication is available from the NASA Center for AeroSpace Information, (301) 621-0390.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Visual Computing Environment (VCE) is a framework for intercomponent and multidisciplinary computational simulations. Many current engineering analysis codes simulate various aspects of aircraft engine operation. For example, existing computational fluid dynamics (CFD) codes can model the airflow through individual engine components such as the inlet, compressor, combustor, turbine, or nozzle. Currently, these codes are run in isolation, making intercomponent and complete system simulations very difficult to perform. In addition, management and utilization of these engineering codes for coupled component simulations is a complex, laborious task, requiring substantial experience and effort. To facilitate multicomponent aircraft engine analysis, the CFD Research Corporation (CFDRC) is developing the VCE system. This system, which is part of NASA's Numerical Propulsion Simulation System (NPSS) program, can couple various engineering disciplines, such as CFD, structural analysis, and thermal analysis.

14. SUBJECT TERMS

Air breathing engines; Simulation; Software tools

15. NUMBER OF PAGES

70

16. PRICE CODE

A04

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102