NASA / TM—2001-210610

Overview of ICE Project: Integration of Computational Fluid Dynamics and Experiments

James D. Stegeman, Richard A. Blech, Theresa L. Babrauckas, and William H. Jones
Glenn Research Center, Cleveland, Ohio

March 2001

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the Lead Center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA's counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized data bases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA Access Help Desk at 301–621–0134

- Telephone the NASA Access Help Desk at 301–621–0390

- Write to:
  NASA Access Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076

NASA / TM—2001-210610

Overview of ICE Project: Integration of Computational Fluid Dynamics and Experiments

James D. Stegeman, Richard A. Blech, Theresa L. Babrauckas, and William H. Jones
Glenn Research Center, Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center

March 2001

Available from

NASA Center for Aerospace Information
7121 Standard Drive
Hanover, MD 21076
Price Code: A05

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22100
Price Code: A05

Available electronically at http://gltrs.grc.nasa.gov/GLTRS

OVERVIEW OF ICE PROJECT: INTEGRATION OF COMPUTATIONAL FLUID DYNAMICS AND EXPERIMENTS

James D. Stegeman, Richard A. Blech, Theresa L. Babrauckas, and William H. Jones
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

SUMMARY

Experimental research and numerical simulation are frequently performed independently. Generally, an experimentalist gathers data in a test cell, views them on-line as they are being collected, and then reduces and analyzes the bulk of the data off-line. Similarly, the analyst models the test article by using a numerical simulation code and then analyzes the results. Significant benefits can be achieved by integrating information from both experimental and simulation processes. For example, data from simulations can quickly be compared with those from experiments so that a simulation code can be validated or anomalies accounted for. Validated simulations can be useful in preparing future experiments. In addition, a searchable database containing integrated information from both experiments and simulations can be a valuable design tool.

In light of these benefits, researchers at the NASA Glenn Research Center have developed a prototype integrated environment for interactively exploring, analyzing, and validating information from computational fluid dynamics (CFD) computations and experiments. The Integrated CFD and Experiments (ICE) project is a first attempt at providing a researcher with a common user interface for control, manipulation, analysis, and data storage for both experiments and simulation. ICE can be used as a live, on-line system that displays and archives data as they are gathered; as a postprocessing system for dataset manipulation and analysis; and as a control interface or "steering mechanism" for simulation codes while visualizing the results. Although the full capabilities of ICE have not been completely demonstrated, this report documents the current system. Various applications of ICE are discussed: dynamic simulation of a supersonic inlet, real-time data visualization and display, low-speed compressor testing, and a parallel-processing simulation code interface for multistage turbomachinergy. A detailed data model for the compressor application is included in the appendix.

INTRODUCTION

Traditionally, experimental research and numerical simulations have been conducted in completely isolated environments. This isolation applies especially in the computer environments used to acquire experimental data and run numerical simulations. This situation makes little sense, especially when the synergism that should exist between simulation and experiments is considered. For example, simulation developers rely heavily on experimental data to validate numerical models. However, validation can often be hampered by the manual process of transferring results from data acquisition computers, reducing the data to a usable engineering form, and finally interpreting the results from the data through a visualization process. Experimental information can be useful for providing initial or boundary conditions for numerical simulations. Also, experimentally derived models are often incorporated in numerical simulations.

Just as experimental data are invaluable for validating numerical simulations, sufficiently validated simulations can be useful in planning experiments. For example, simulations can be used to estimate the effect of the experimental facility on the measured results. The simulation information can also be useful for determining the location and amount of instrumentation.

All these situations require significant movement and manipulation of information among several computing environments. Each environment can have different operating systems, data formats, user interfaces, and capacities. In addition, users frequently are forced into repetitious procedures for reducing and manipulating data from both experiments and simulations. Valuable data can also be lost if inadequate facilities are provided for archiving and documentation. Finally, long experimental turnaround time and lack of interactive tools for analyzing results can further hinder progress. In summary, significant time, energy, and money are wasted owing to insufficient integration of information from both the experimental and simulation worlds.

The Integrated CFD and Experiments (ICE) project was begun to address these issues. The initial ICE concept is presented in reference 1, and details on the ICE experimental subsystem are given in reference 2. This report documents four specific applications of ICE: the dynamic simulation of a supersonic inlet, real-time data acquisition and display, low-speed compressor testing, and a parallel-processing simulation code interface for multistage turbomachinery.

ICE DESIGN

ICE can be thought of as an environment that allows a researcher (or researchers) to coordinate the integration of information-generating processes and thus to conduct a comprehensive investigation of any project from a single computer (fig. 1). The solid lines in figure 1 indicate direct lines of communication built into the ICE system. The dashed lines indicate that some type of interface routine is required for this level of process control. The term "knowledge base" (KB) is used to denote that the information managed by ICE is more than just raw data. It includes images and users' notes that aid in interpreting the data. It also maintains a postprocessing "history" of the data that may be useful at a later time if questions regarding validity should arise. At this stage in the ICE development "knowledge base" is not intended to imply use of expert systems or artificial intelligence technology. The use of such technology is, however, an eventual goal of the ICE project.

Brainstorming sessions were held to determine what types of information and functionality would be necessary for an analyst and an experimentalist to use this system. These initial ideas were presented to researchers from both experimental and computational backgrounds. These ideas were iterated upon, since a computer scientist's idea of necessary information is markedly different from that of someone performing the actual research.

The resultant information was then distilled down into the most basic categories. These categories form the foundation for a knowledge base (table I) and a set of reusable libraries and processes (fig. 2) upon which the system is built.[1] The libraries provide basic services, such as graphical user interface generation, process management, and knowledge-base access. These libraries are used to develop standard ICE processes that support the user in generating, displaying, analyzing, and documenting data. The libraries can also be used to develop custom processes tailored to a specific application. The standard processes provided with ICE are the editor, visual, knowledge-base management, and Interactive Data Display System (IDDS, ref. 3) processes. The editor process is fairly generic in its operation. Once the ICE system receives an EDIT request, the editor process is spawned with a handle to the object to edit. The editor process checks the validity of the request by interrogating the object's EditEnable flag

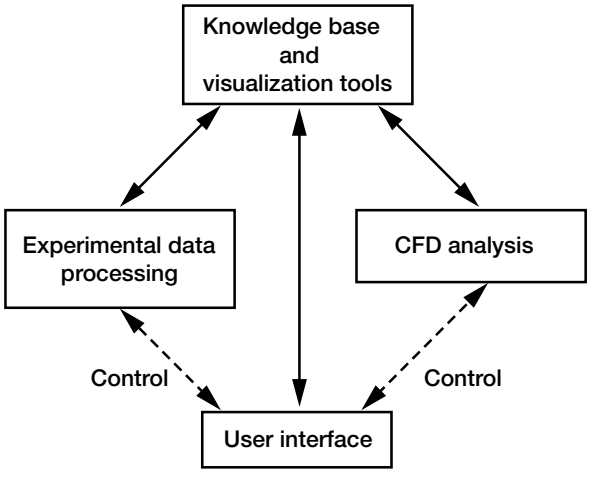


Figure 1.—Initial concept of ICE system.

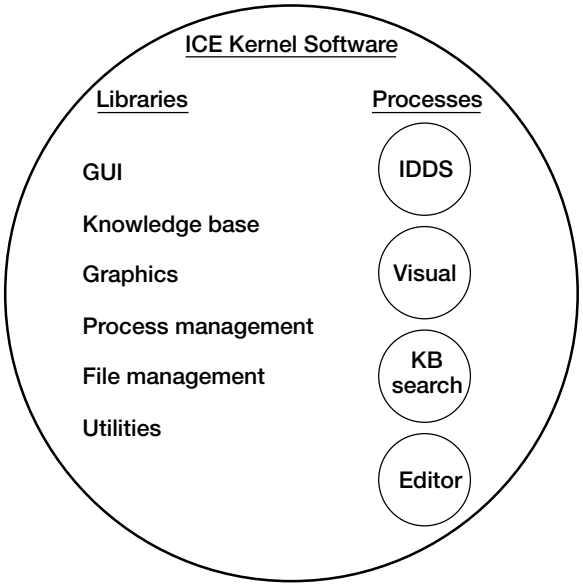


Figure 2.—Basic libraries and processes.

[1]See the appendix for a complete description of table I.

TABLE I.—BASIC KNOWLEDGE-BASE TYPES AND ATTRIBUTES

Basic types		
Class	Members	Data files
Attributes		
ID	Type	ID
Type	Description	Type
Depth	EditEnable	Depth
Genealogy	Display	Genealogy
FileSpec	Precision	FileSpec
Process	StringSize	InitialProcess
	CharSets	UpdateProcess
	WordSet	AnalysisProcess
	Genealogy	
	MinVal	
	MaxVal	
	MinAlarm	
	MaxAlarm	
	Implicit	

(table I). If the request is valid, the appropriate field (or fields) is presented to the user for manipulation. If the request is not valid, the user is informed and the editor process exits. Once a valid edit session is complete, type and range checking is performed.

All ICE data types have a default visualization technique associated with them. The visual process provides the user access to these techniques along with the data interrogation capabilities of ICE. The default method for array visualization is to generate a plot. A tabular display of the plot data is also available. Data interrogation is accomplished with a set of "probing lines," as shown in the upper right quadrant of figure 7 (see page 7). These lines can be vertical, as shown, or they can be horizontal. The user controls their orientation according to the information desired.

The knowledge base upon which ICE operates captures the entire sphere of data associated with a testing program in a logical, well-ordered manner. Each piece of information either relates to or is dependent on another piece of information, thus forming a structured hierarchy of information. The information can refer either to a test cell and experimental data or to a parametric study of some phenomenon that uses a simulation tool. All data can be included in an ICE knowledge base.

IDDS is an interactive graphics package that provides two- and three-dimensional analysis capability for the ICE system. IDDS and ICE have been designed to operate together through file sharing or through shared-memory communication. File sharing has proved best for "slicing and dicing" the data, and shared memory has been most efficient for visualizing real-time data.

All in all, the elements of figure 2 combine to provide the following default functionality: a graphical user interface (GUI) for process and knowledge-base management, process run time and process control panels, two- and three-dimensional plotting capabilities, static or dynamic table displays, knowledge-base search panels, knowledge-base data display or editor services, and finally, on-line documentation services.

Installing and Using ICE

The initial ICE system by itself knows nothing about an application such as a multistage compressor. ICE must be configured to understand the data and processes specific to an application. The configuration process is referred to as "installing" the ICE system. For example, a test facility for a class of aerospace components, such as compressors, would be a prime candidate for an ICE installation. An ICE installation is therefore referred to as "a facility." To develop a facility, one must first identify the structure of the facility (e.g., compressor) and the processes that will be generating the facility information (i.e., the result-argument relationships). Once this identification has been accomplished, the results and arguments can be further organized into a knowledge-base tree (fig. 3). By using the tree information along with the basic knowledge-base types (i.e., class, members, and/or data files of table I), the desired structure of the knowledge base for ICE can be defined.[2] This definition is stored in a text file (see the appendix) that is read by ICE so that its internals can be targeted to meet application demands. By targeting its internal structure ICE links the application-specific processes to the knowledge base and the appropriate GUI

[2]The actual method used for determining the knowledge-base structure is beyond the scope of this report. The interested reader is referred to any text on object-oriented modeling and design.

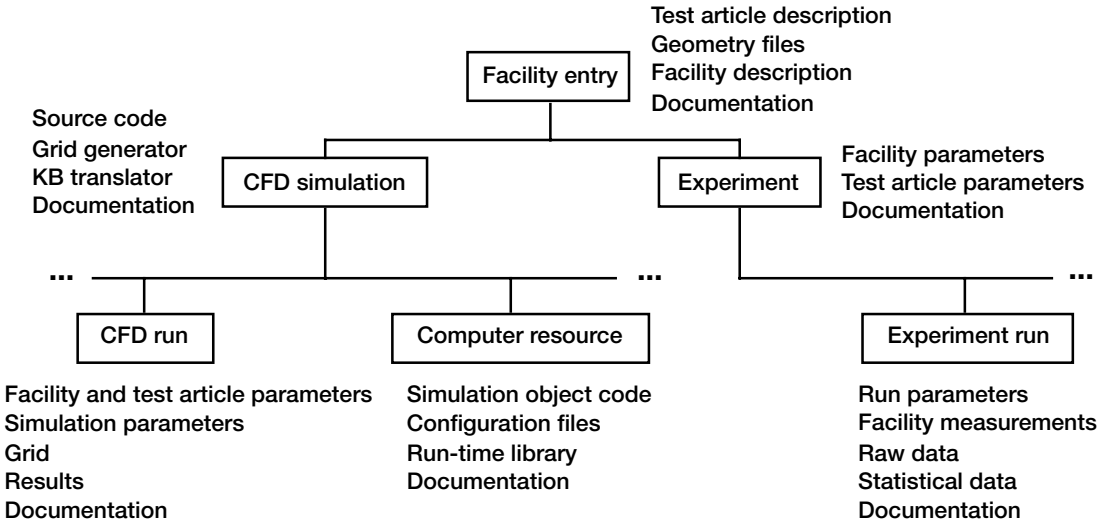


Figure 3.—Example of ICE knowledge-base structure.

elements. Thus, the ICE knowledge base consists of records that define process parameters, the associated process result files, and any descriptions, notes, or pictures generated during initialization or run time or while performing postrun analysis.

Once the ICE facility and its associated knowledge base have been defined, the user then uses the GUI to navigate through the knowledge base performing the necessary operations. Examples would be process manipulation (start, stop, reset, etc.), file manipulation (save, delete, etc.), visualization (two and three dimensions, tables, etc.), documentation (image saving, on-line notes, etc.), and basic parameter editing. Parameter editing allows the user to change the input to a process, execute the process, and visualize the results. In the case of an experiment, the parameters may directly control some aspect of setting facility conditions, controlling the test article, or triggering data acquisition. In the case of a simulation, parameter editing can be used to modify boundary conditions or otherwise perturb the simulation. This process is sometimes called computational steering. Of course, the simulation being steered must be modified to accept and acknowledge the parameter changes. This type of operation is illustrated in the next main section.

ICE Templates

To assist the users of ICE, we developed three default templates by using the ICE libraries:

1. CFD (simulation)
2. EXP (experimental)
3. Analysis (analytical)

A template can be thought of as an example ICE application that can easily be customized for a specific facility. The default templates form three subsystems, shown on the right side of figure 4. Each subsystem's main interface routine must execute on the host ICE machine. However, both the CFD and EXP subsystems can have interface processes that allow the simulation or the data gathering to take place on distributed computers. Communication specifics are left to the developer. Members of the ICE project have used Application Portable Parallel Library (APPL, ref. 4), Parallel Virtual Machine (PVM, ref. 5), ScramNet (private communication), and the local network to transfer data. APPL is an in-house-developed library of routines used for transferring data between computers. PVM is a public domain software package with similar capabilities. ScramNet is a commercial product that uses a fiber optic network and control software to transfer data between computers. Complete discussions of these products can be found in the references and company literature. The information exchange buses shown in figure 4 are abstract concepts used by the ICE design team. The information exchange buses refer to the internal movement and sharing of data among ICE processes.

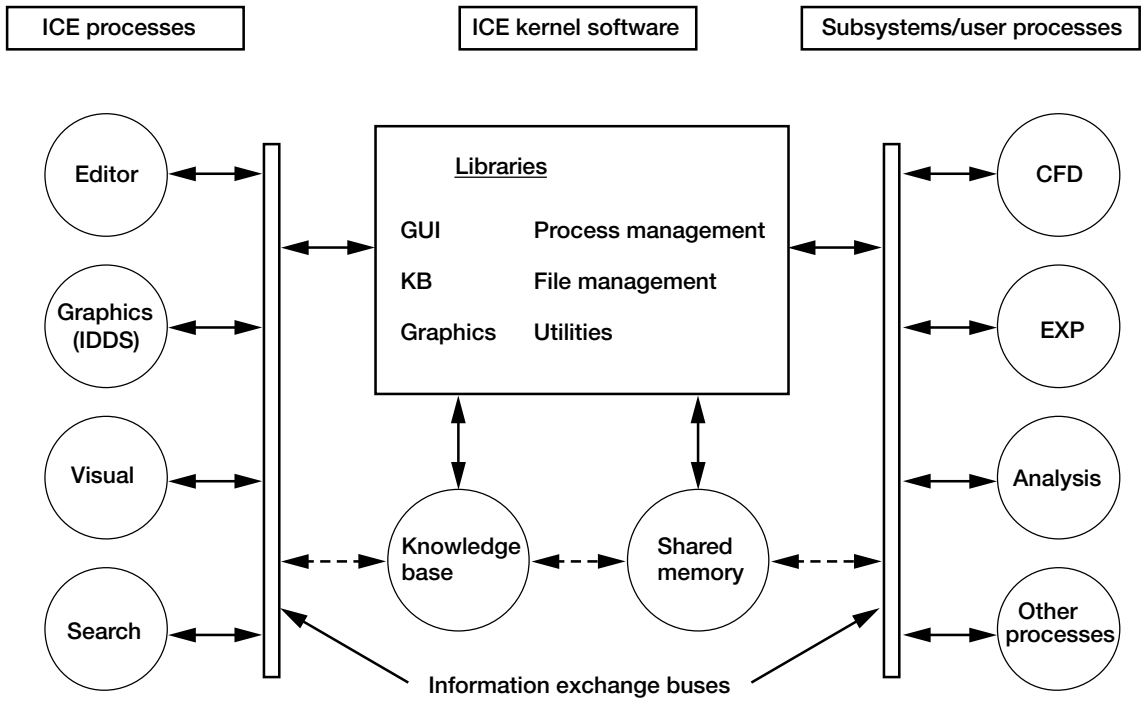


Figure 4.—Conceptual ICE architecture.

Also shown in figure 4 is a circle labeled “other processes.” When installing ICE the user can develop other subsystems (from blank templates) for use alongside of or in place of those provided by the default configuration of ICE. The ability to customize an environment provides the installer (or user) with the flexibility to take into account specifics that might have been missed by the ICE design team.

Examples of specific CFD, EXP, and Analysis subsystems will be presented in the following sections. Note that there is a standard template for each subsystem but not a standard configuration for each subsystem.

APPLICATIONS OF ICE

This section presents examples of how ICE has been configured for four specific applications.

Supersonic Inlet

The NASA Glenn Research Center is heavily engaged in jet engine research, including high-speed inlet design and performance assessment. NASA Glenn has collaborated with Boeing Aircraft Corporation to investigate the commercial feasibility of a supersonic transport. Under a Space Act Agreement between Boeing and NASA the ICE system was applied as an aid for understanding supersonic inlet performance. The inlet research program included wind tunnel testing and numerical simulation. The Large Perturbation Flow Field Analysis and Simulation for Supersonic Inlets (LAPIN, ref. 6) computer simulation code was identified as having the required modeling capability for this project. The LAPIN code performs one-dimensional, time-accurate simulations of various supersonic inlet configurations.

In beginning this project the first item of business was to identify the most useful way in which to use ICE for the inlet application. Because the ICE system was not available in time to be used for the wind tunnel test program, we decided that the main ICE interface would be used to control the operation of the LAPIN code (i.e., we wanted to steer the code). Figure 5 shows the basic inlet configuration used for this project (note that both the cowl and the translating centerbody should be rotated about the centerline for a true three-dimensional inlet). To accomplish the steering, we examined the LAPIN variables to determine their functions. Examples of these variables include

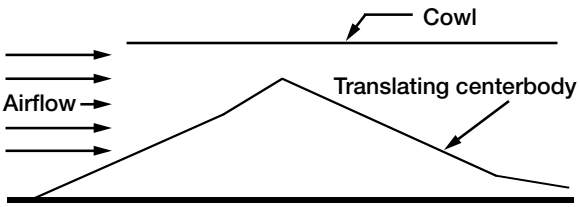


Figure 5.—Basic inlet configuration used by ICE/LAPIN.

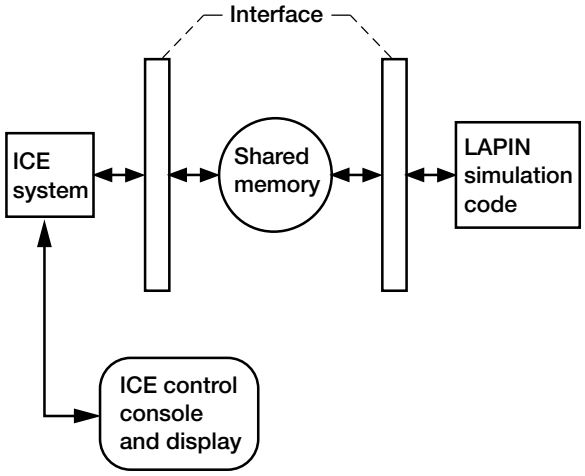


Figure 6.—Conceptual layout of ICE/LAPIN configuration.

centerbody position and inlet and exit boundary conditions. Once these have been determined, they are made available to the user to perform the actual steering. An ICE knowledge base was then constructed for the inlet application. The name and type (i.e., integer or float) of each variable were then defined in the ICE knowledge-base configuration file. For ICE and LAPIN to access these variables we had to develop interface routines for LAPIN that connected to ICE in the appropriate manner. We chose to use shared memory for transferring the information between the interface processes, as shown in figure 6. Note that all processes in figure 6, except for LAPIN, execute on the same machine. LAPIN could also run on the same machine as the other ICE processes if desired, but for this example it was run on an external workstation or supercomputer.

Figure 7 depicts a typical control screen for the ICE/LAPIN project. Shown in the upper left quadrant is a real-time monitor panel. The title bar contains the date and time this work was performed and the knowledge-base node with which the user is interacting. Following that are items that the user decided were of interest when this port was being developed: the number of time steps the code has completed, the accumulated simulation time, the wall clock time when the simulation started, the mode of operation for LAPIN, a slider bar for variable geometry positioning, and free-stream and exit boundary condition information.

The lower left quadrant shows seven steering variables and their values: perturbation type, perturbation amplitude, perturbation width, perturbation boundary condition, perturbation start time, boundary condition setpoint, and boundary condition measured value. Other steering variables are available to the user, but they are not shown here. The variables shown allow the user to investigate what happens to inlet performance when the airflow is disrupted at various inlet locations by varying both perturbation levels and boundary conditions.

The upper right quadrant is typical of an on-line plot with ICE. The lower curve is a plot of the perturbation to the inlet. The upper curve is a plot of the inlet's response to the perturbation. Both plots are of data taken downstream of the shock location. The two vertical lines in this plot are user controlled. Currently, they are indicating the time of a peak input perturbation (lower curve) and the resultant response to that perturbation (upper curve). By obtaining the time difference between input and response the user can calculate the phase difference and thus see how responsive the current inlet is to changing flow conditions.

The lower right quadrant is also typical of an on-line plot with ICE. Shown here along the y axis is a calculated pressure ratio against the corresponding axial coordinate. The separation between the two lines indicates the shock wave location. A display of this type is useful for seeing how changing conditions affect shock wave location. The position of shock wave location is vital for supersonic inlet performance.

As the code is running, the upper and lower right quadrant displays will automatically update their values. By using the slider bar control (upper left quadrant) or by changing one or more variables (lower left quadrant), the user can then see how the changes will affect the items of interest in the remaining quadrants.

As previously mentioned, the LAPIN simulation code could be executed on the same machine as the ICE software. Because the simulation is a relatively simple one (only one dimensional), ICE is able to perform its functions and the user is still able to visualize the results in a near-real-time mode. If the simulation were more complex (two or three dimensional), this would not be possible. Most of the computer's processing power would be required to

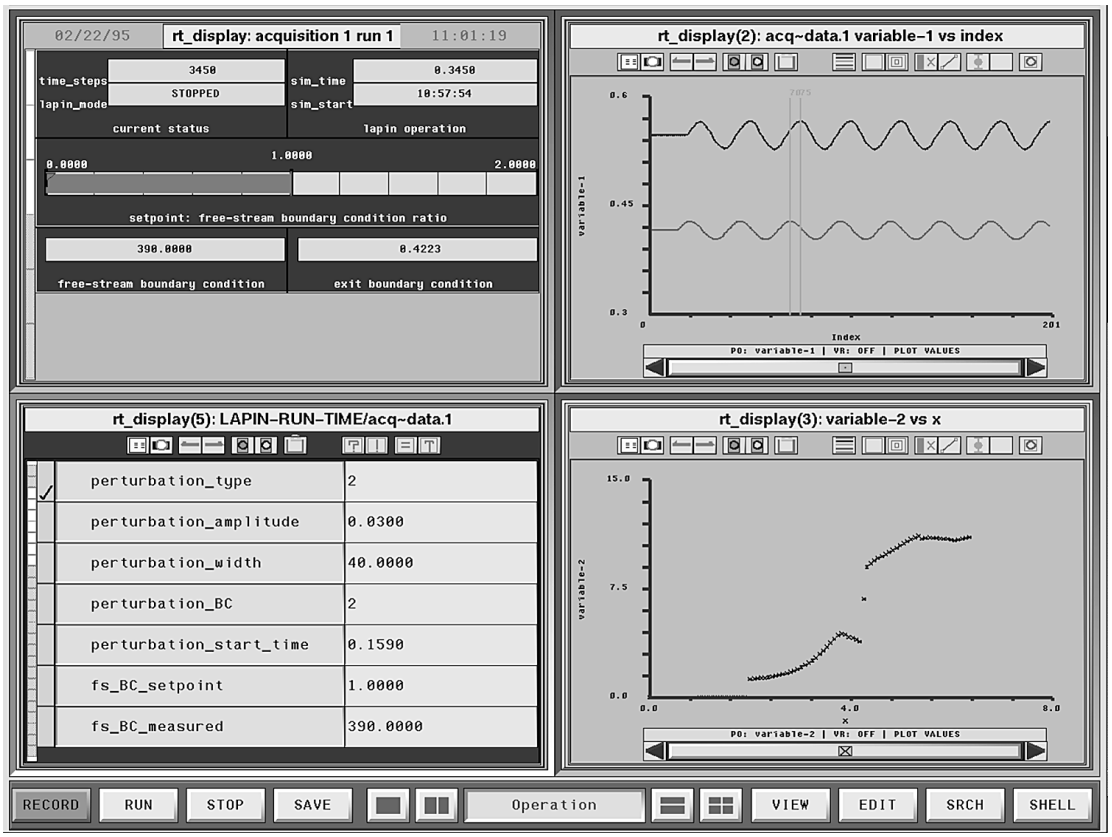


Figure 7.—ICE/LAPIN control screen.

solve the simulation with little power left over for the ICE system. To address this situation, the interface routine for the LAPIN code (fig. 6) was redesigned such that communication with LAPIN would take place through APPL or PVM. By doing this, the simulation code being used could now be run on a different computer, thus taking advantage of any special processing power afforded to that computer. For example, if the simulation code is designed to run with optimal performance on a parallel computer, the ICE software could be accommodating. As an extreme example of this, the ICE/LAPIN software was delivered to Boeing in Seattle and installed on one of their computers. The LAPIN code was then run on the CRAY Y–MP computer located at NASA Glenn in Cleveland. Communications were accomplished between the two sites through satellite and ground terminals.

Real-Time Data Acquisition and Display

NASA Glenn is a large campus with numerous test cells connected through the local area network (LAN). Frequently, it is desirable to remotely access and visualize data from these test cells. Toward that end, procedures and routines have been developed to transport the collected data and store them in a central repository for future reduction. ICE team members collaborated with members of the Scientific Engineering Branch of the Computer Services Division at NASA Glenn to integrate the ICE software in this environment. Data, which had already been collected, were replayed over the network and collected by an ICE interface routine. The interface routine then placed the data into shared memory for retrieval by the Analysis subsystem. Figure 8 is a schematic layout of this integration configuration. Figure 9 shows an example of the Analysis subsystem output. The left side of figure 9 consists of simulation data, and the right side depicts the replayed experimental data. Both types of data are displayed by using the IDDS (developed at Glenn). Once the data have been input into IDDS (and the Analysis subsystem), the user can apply all the available manipulation tools supplied by IDDS to the data.

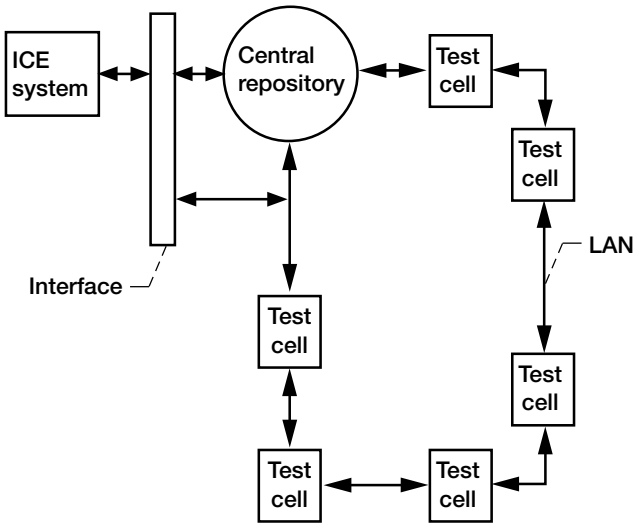
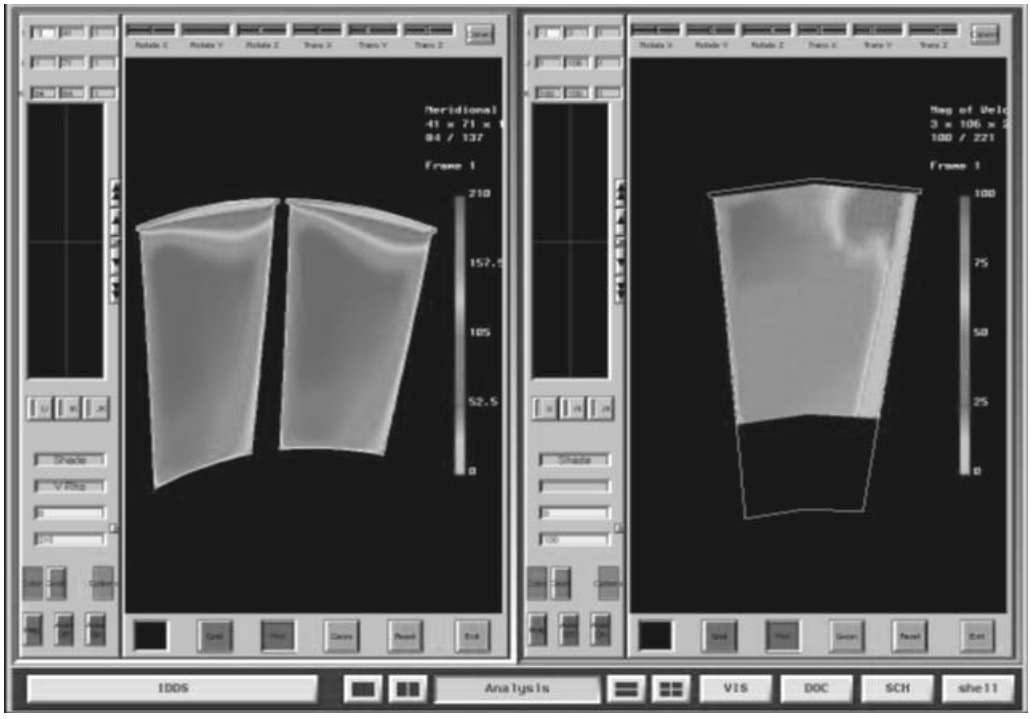


Figure 8.—ICE/LAN integration configuration.



CFD data Experimental data

Figure 9.—ICE/LAN visualization screen.

Large, Low-Speed Compressor Facility

NASA Glenn performs research on many different aspects of propulsion. In particular, an ICE installation was developed to support compressor research. Two types of compressor are used in an aircraft engine: axial and centrifugal. An axial compressor takes the flow entering from the left and simply compresses it while passing the flow to the right (fig. 10(a)). A centrifugal compressor "turns" the flow 90° from entry. That is, if the flow enters from the top, it will exit to the left and/or right (fig. 10(b)). The Large, Low-Speed Compressor Facility (LSCF) at Glenn

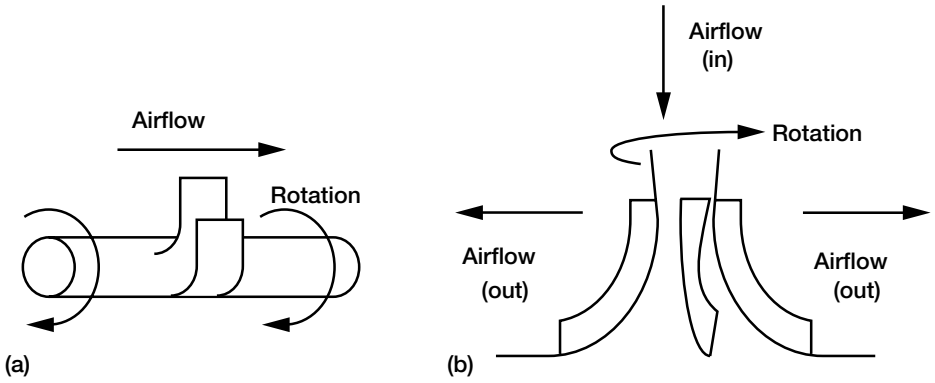


Figure 10.—Compressor types. (a) Axial compressor configuration. (b) Centrifugal compressor configuration.

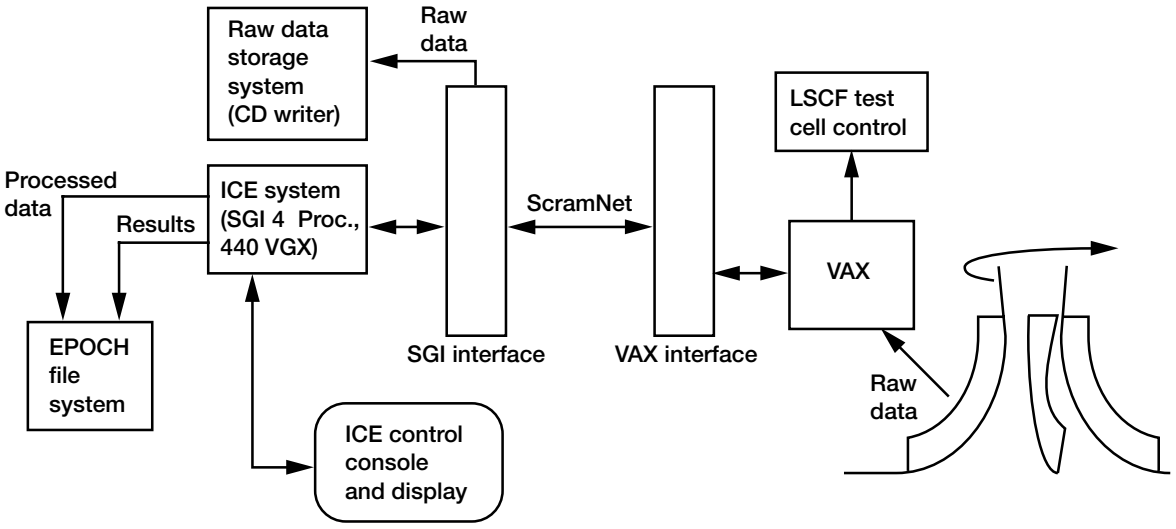


Figure 11.—ICE/LSCF proposed system configuration.

focuses on the centrifugal compressor. This facility derives its name from the fact that the machines tested are scaled up in size, making it easier to measure the flow field within a blade passage. Typically, the machines are scaled to 10 to 15 feet in diameter. A debate on the usage of scaled data is beyond the scope of this report.

Significant effort was expended in tailoring ICE to the experimental part of this application. The initial concept is documented in reference 2. One requirement was to have run-time displays of various flow velocity averages in a blade passage. The averaging procedures used for these displays were computationally intensive. A data acquisition process was developed that used parallel processing to achieve the speed required to meet the run-time display requirement. This process was then integrated into the ICE system. Another requirement was that the ICE system interface nonintrusively with an existing VAX-based data acquisition system. Finally, mass storage had to be provided to support extremely large (several megabytes) datasets. Figure 11 shows the architecture of the hardware environment within which the ICE software operates.

The system in figure 11 operates in real time, acquiring and displaying data as follows: Raw data are gathered from the test article under consideration by a VAX computer. The data are then stored on the VAX and then sent to the ICE system through a ScramNet link. Once acquired by the ICE system, the data are routed directly to the compact disk writer for permanent storage and then handed off to one of the processors for reduction (ensemble averaging, flow vector computation, etc.). The reduced dataset is now available for display on the ICE console.

The control console data are refreshed whenever the most recently acquired raw data have been reduced. The user need do nothing to get a refreshed screen. When something appears of interest to the researcher, he/she need only request it “saved” and the current data making up the state of the console display will be routed to an Epoch file

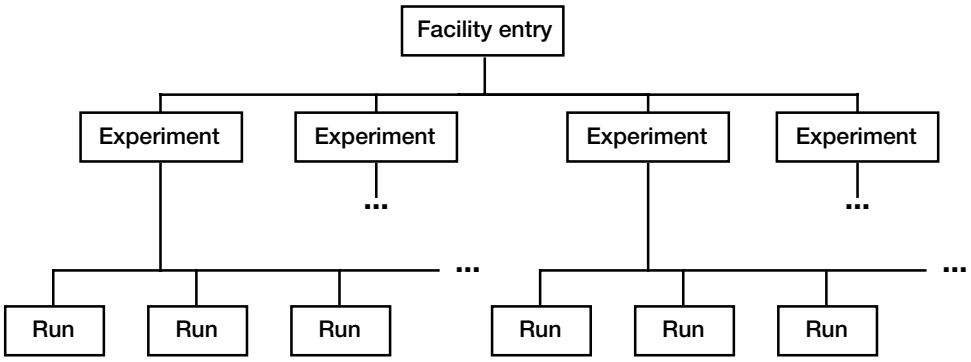


Figure 12.—LSCF knowledge-base structure.

TABLE II.—ATTRIBUTES OF LSCF KNOWLEDGE-BASE STRUCTURE

Node of figure 12		
Facility entry	Experiment	Run
Attributes		
Test article description Geometry files Facility description Documentation	Facility parameters Test article parameters Documentation	Run parameters Facility measurements Raw data Statistical data Documentation

system for long-term storage and retrieval. Any notes or computations made by the user will also be attached to the saved data.

When the processed data are saved to the Epoch system, they are saved according to the format (organization) specified when constructing the knowledge-base layout. Figure 12 shows the layout for this test. Each of the nodes shown contains one or more attributes used for description or data containment. Table II lists these attributes, which the reader will note are the same there as in figure 3. The detailed definition of the knowledge base for the LSCF was a significant effort. The complete definition can be found in the appendix. It is replicated herein as it may be useful to other turbomachinery data-modeling activities.

Numerous timing tests and data reduction simulations were performed. Results indicated that the ICE system developed for the LSCF would have met expectations. Unfortunately, because of time and schedule constraints ICE was never utilized during an actual test.

MSTAGE: CFD Simulation Integration

The multistage turbomachinery CFD simulation MSTAGE (ref. 7), configured to simulate compressors, was integrated into the ICE system. A CFD simulation graphical user interface was developed to accomplish this task in ICE. The GUI simplifies the use of CFD codes by in-house experts. The MSTAGE code can be used to help design experiments in the LSCF. In this way ICE is very useful. The user can archive information pertaining to a particular CFD code by using the ICE knowledge base. The user can also specify the values of the input variables associated with the CFD code. After execution the code's results can be archived along with the values of the input variables for future study and record-keeping purposes. The archived information from a series of MSTAGE runs is used to guide experiments and offers the most judicial use of the LSCF. This guidance reduces the time and money spent on testing compressor designs.

A walkthrough of the MSTAGE code integrated into the ICE system is as follows: The interface is spawned from the ICE main screen by clicking on the RUN OBJECT PROCESS button. The interface then asks the user whether or not a new input file for the MSTAGE code needs to be created from the values in the knowledge base. After the user selects an answer, the CFD simulation monitor and architecture selection screen are drawn as illustrated in figure 13. The left side of figure 13 depicts the simulation run time and the clock time, current modes,

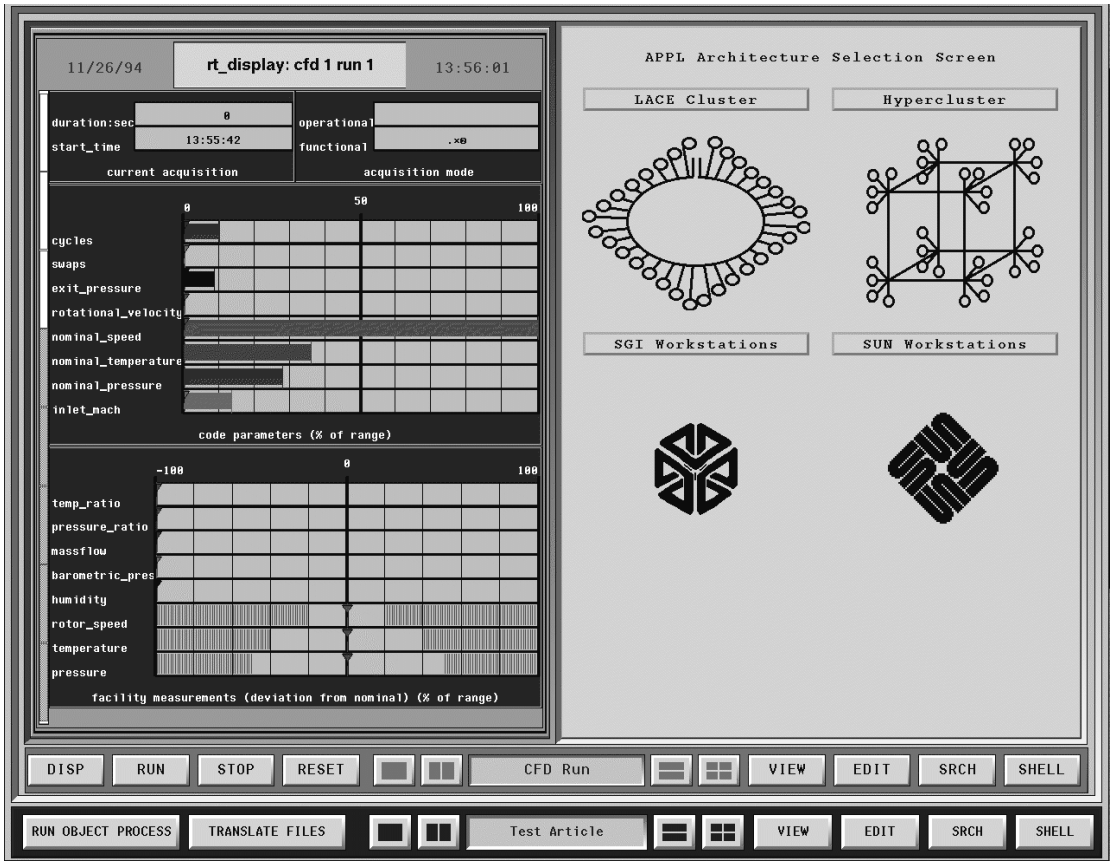


Figure 13.—Parallel-processing user interface.

various MSTAGE code parameters, and a processor load monitor. The right side of figure 13 depicts the architecture selection screen. The architectures shown for a particular user are determined from a text file that lists all the parallel-processing environments for which the user has an account.

The user then chooses the architecture and the individual processors on which to run a CFD code. Parallel architectures are represented graphically with icons. Any unavailable machines or processors are “grayed out.” The user clicks with a mouse on the desired available architecture from the choices shown. The software then shows the available processors on that architecture. Figure 14 illustrates the resulting screen if the user selects the LACE cluster. The LACE cluster is composed of 32 IBM RS6000 workstations interconnected by a high-speed network. The user has several options at this point. The user may return to the main screen by clicking on the Main Screen button (lower left of fig. 14). Alternatively, the user may wish to view the processor usage of the selected architecture. The left side of figure 14 shows an example output after clicking on the Processor Usage button (upper left of fig. 14). From this information the user can determine which processors to avoid or select. The user then selects which processors will be used to execute the parallel CFD code. After the processors are selected, the user clicks on the Create procdef button (on the lower right of fig. 14).

Once the parallel architecture is selected, the user designates which field (or fields) of MSTAGE’s run-time record will be visualized in a x-y plot. A click on the desired data in the knowledge-base screen will create the resulting x-y plot of the data. Execution of the code begins with a click on the RUN button. Once the execution starts, a timer informs the user of the execution start time and length as shown on the left side of figure 15. While the CFD code is running, the data outputs selected previously are displayed and updated on a x-y plot format by using the ICE visualization function. Figure 15 shows an example of visualization of the intermediate data from the CFD code. After the CFD code is finished executing, the user will be able to view and archive the results in the ICE knowledge base.

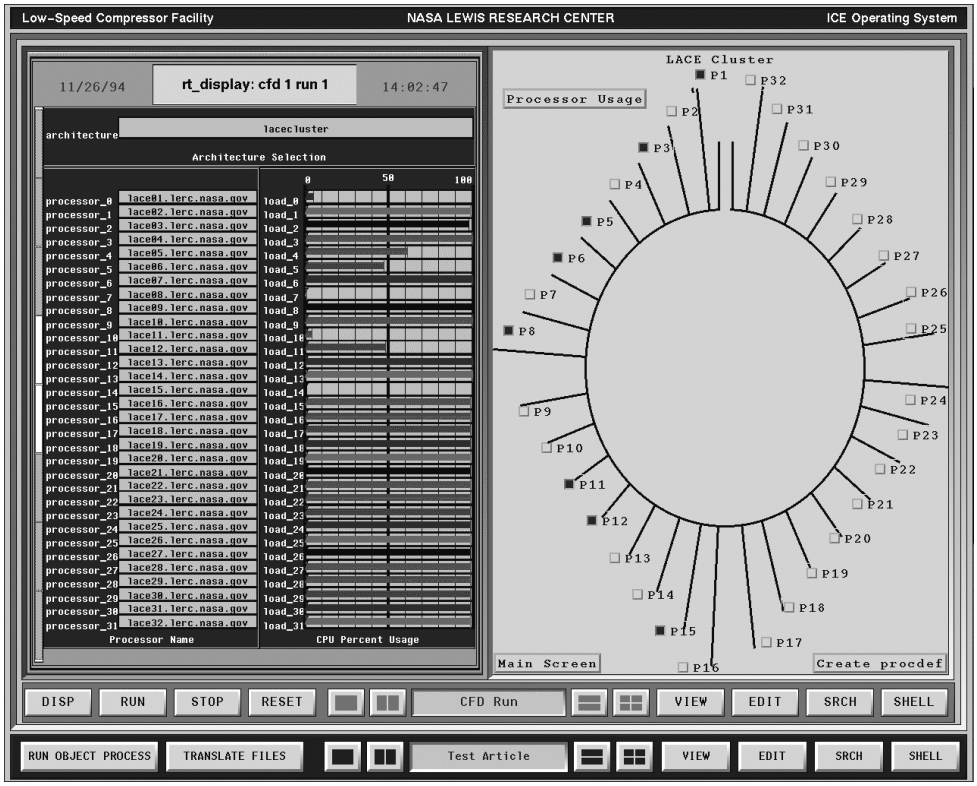


Figure 14.—Processor assignment and usage interface.

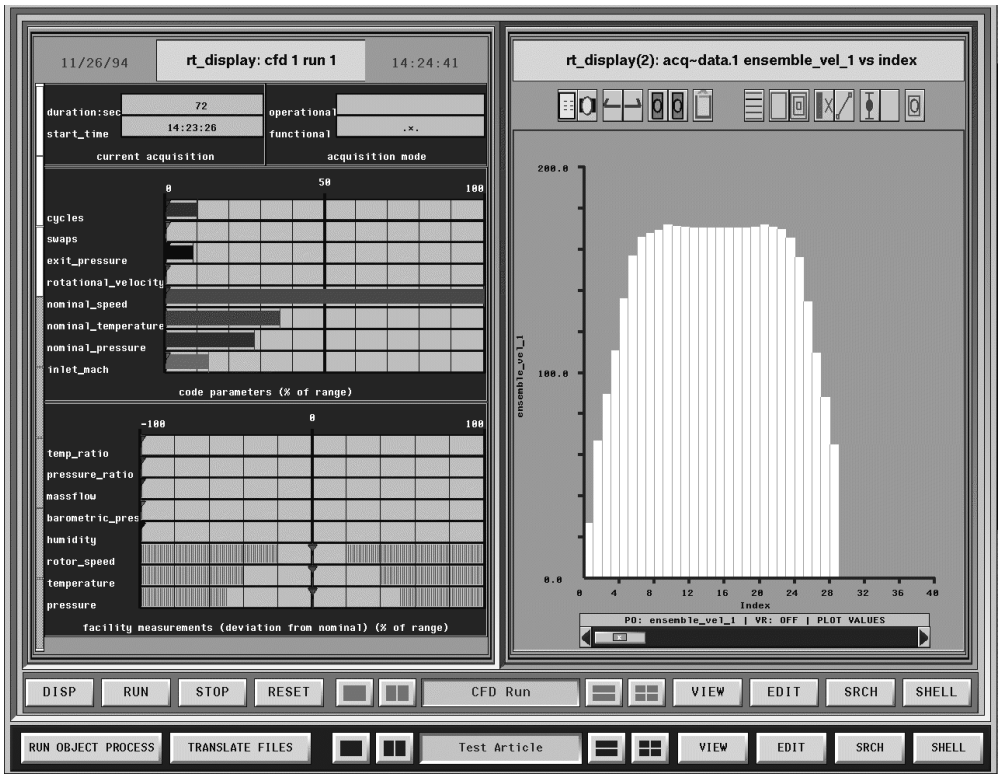


Figure 15.—Run-time display of simulation results.

CONCLUDING REMARKS

The Integrated CFD and Experiments (ICE) system as designed is operational. By using shared memory to attach processes, ICE can be configured and expanded to meet system requirements. It is capable of accepting and displaying both experimental and simulation data. The user can save these data and recall them at a later time for further analysis or search the data for specific values or trends. Everything we started out to do has been accomplished. However, ICE, as it is written, is operating system dependent. When we started out on this endeavor (~1990), personal computers (PC's) were not very powerful, C++ was still in its infancy, and Java had not been invented. Decisions had to be made regarding programming language (C, Fortran, C++), graphics for windowing (X, Motif), graphics for plotting (gl), databases (Oracle, Sybase). There was a limited budget and we used machines (SGI) that would fulfill our most pressing requirement: graphics. It was decided that a user would not like to wait very long for a picture to be drawn, and so we opted for a machine that pushed pixels the fastest.

Integrating an application into ICE is not a trivial task. An ICE team member and the future user must sit down and, together, they will design the desired system configuration. A template has been developed to assist in integrating new simulations into the ICE environment. Significant expertise is still required in data modeling and software engineering. However, ICE is an excellent first attempt at incorporating the worlds of simulation, experimentation, and analysis.

FUTURE WORK

ICE, or a concept like it, should be made portable to all system platforms: PC and Unix. ICE is, essentially, a single-facility data manipulator. Under the ICE system comparing data from two or more facility entries (see fig. 3) can be rather arduous, at best. If the user interface is not explicitly told where to find the data, ICE will not let the user search the other facilities and import data into the current facility session for comparison purposes. Toward that end, work has begun on a new concept called Portable Redesigned Integrated CFD and Experiments (PRICE).

PRICE uses object-oriented technologies to produce well-known, generic objects that can be easily mixed and matched to meet the specific requirements of individual projects. The basic principle of PRICE is to split the conceptual aspects of the software system in half: one half manages the data and all interrelationships therein; the other half presents information to the user uncovered during a traversal of that underlying managed data. The consuming object can be a GUI or other forms of code: for example, a search engine or even another PRICE-wrapped application. This feature helps eliminate two problem areas: information propagation and code interface development (GUI's).

Figure 16 shows a high-level conceptual model of the PRICE environment. As is suggested in the figure, specific application wrappers are built upon well-known, closely defined, object-encapsulated components from the foundation application architecture and infrastructure. The infrastructure imposes (and supplies) a relatively small number of mandatory capacities. Most application elements are defined in an optional, and thus discoverable, manner.

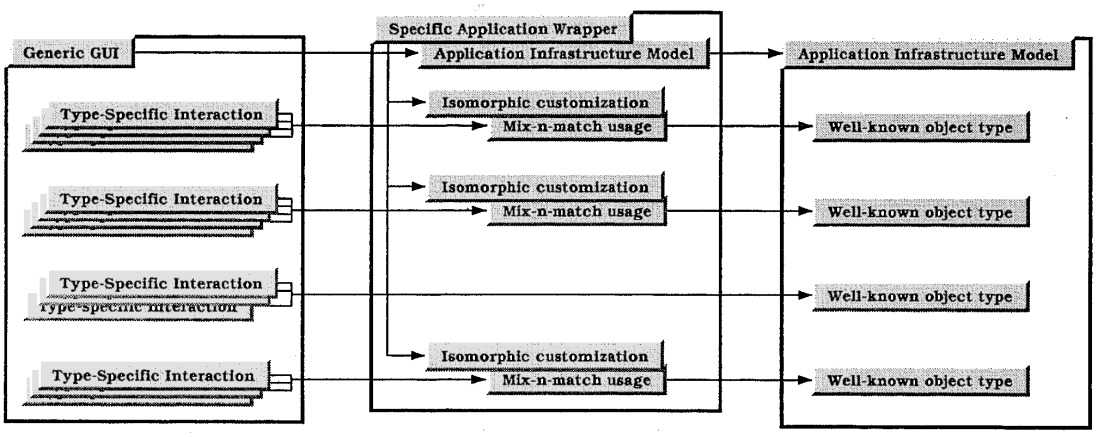


Figure 16.—High-level conceptual model for PRICE.

An application typically has operations that it will perform, each encapsulated in an operation object. However, the set of operations presented by a specific object may be empty and, thus, it must be scanned (by a known method) to discover which operations do, in fact, exist. Similarly, the parameters presented by the application are defined in a discoverable manner.

Each encapsulated element comes with functionality appropriate to its type. For example, virtually all objects of the architecture "know" how to save and restore themselves to and from persistent storage. Customizing types to the specific needs of an application is allowed through object derivation mechanisms. However, the architecture imposes the restriction that such customizations not alter the basic nature of the encapsulated information. Thus, a Mach number must stay a Mach number for all external purposes but may be customized with, for example, code that converts experimental pressure tap readings to that Mach number for some set of circumstances peculiar to the experimental application choosing to use that well-known form.

With this architectural formulation of applications in hand, the task of defining and implementing a consuming GUI, although perhaps not easy, becomes one that is no longer specific to a particular application. A GUI would be constructed with one or more interaction modules for each well-known encapsulated form of the application architecture. Each module would deal with that information in a manner appropriate to the corresponding information. Upon discovering the nature of a particular piece of an application, the GUI would then dispatch control to the module appropriate to that element. When multiple interactions exist for a particular form of information, the user would be allowed to switch between the different choices.

Because the nature of information is a well-known trait of the encapsulating objects, it is also possible to code other consuming forms to automatically consume or exchange information between applications. Graphs of applications may be arranged where, upon an appropriate signal, successor applications may interrogate their predecessors to obtain particular needed forms of information. Because of the fine-grained encapsulation of information, this act of propagation becomes content focused rather than application focused. That is, an application hoping to acquire flow-field enthalpies looks for flow-field enthalpy objects rather than for applications known to produce flow-field enthalpy results.

The PRICE infrastructure is currently under development. Further information is available on the World Wide Web at http://www.grc.nasa.gov/WWW/price00.

REFERENCES

1. Szuch, John R.; Arpasi, Dale J.; and Strazisar, Anthony J.: Enhancing Aeropropulsion Research With High-Speed Interactive Computing. NASA TM–104374, 1991.
2. Babrauckas, Theresa L.; and Arpasi, Dale J.: Integrated CFD and Experiments: Real-Time Data Acquisition Development. ASME Paper 93–GT–97, 1993.
3. Stegeman, James D.: User's Manual for Interactive Data Display System (IDDS). NASA TM–105572, 1992.
4. Quealy, Angela; Cole, Gary L.; and Blech, Richard A.: Portable Programming on Parallel/Networked Computers Using the Application Portable Parallel Library (APPL). NASA TM–106238, 1993.
5. Sunderam, V.S.: PVM: A Framework for Parallel Distributed Computing. Concurrency: Practice and Experience, vol. 2, no. 4, Dec. 1990.
6. LAPIN: Large Perturbation Flow Field Analysis and Simulation for Supersonic Inlets. Version 2.0b, I/O Guide, NASA Lewis Research Center, Aug. 1995.
7. Mulac, Richard A.; and Adamczyk, John J.: The Numerical Simulation of a High-Speed Axial Flow Compressor. ASME Paper 91–GT–272, 1991.

APPENDIX—ICE CLASSES AND TYPES AND AN EXAMPLE

ICE software is designed to look and act object oriented without actually using object-oriented technologies. This design was accomplished by using C programming language to define key words and structures. Once ICE is started, a parsing routine takes over and reads the prescribed file. Tokens (key words) are found and combined with the token type to build a hierarchy of these relationships in memory. Please note that classes define complex data structures. The relationship between classes is implied through a tree structure where each branch of the tree defines a new class. As such, it becomes a rather straightforward classification structure (see table I) as follows:

1. Classes define levels in a tree structure.
2. Classes can have members.
3. Classes can have data files.
4. Classes are defined until another class token is found.
5. Classes prescribe their own depth in the tree structure.
6. Members are elements of a class.
7. Members can have attributes.
8. Data files provide initialization information to a class. That is, data files contain the raw image of experimental or simulation data. These files are then read by ICE to initialize the knowledge base.

To aid in understanding, we depicted the hierarchy in figure 17. In order to “input” this hierarchy to ICE, the following key words and attributes (class, members, and data files) would have to be entered into a text file for parsing. Note that the actual knowledge base defined for the LSCF is presented in this appendix.

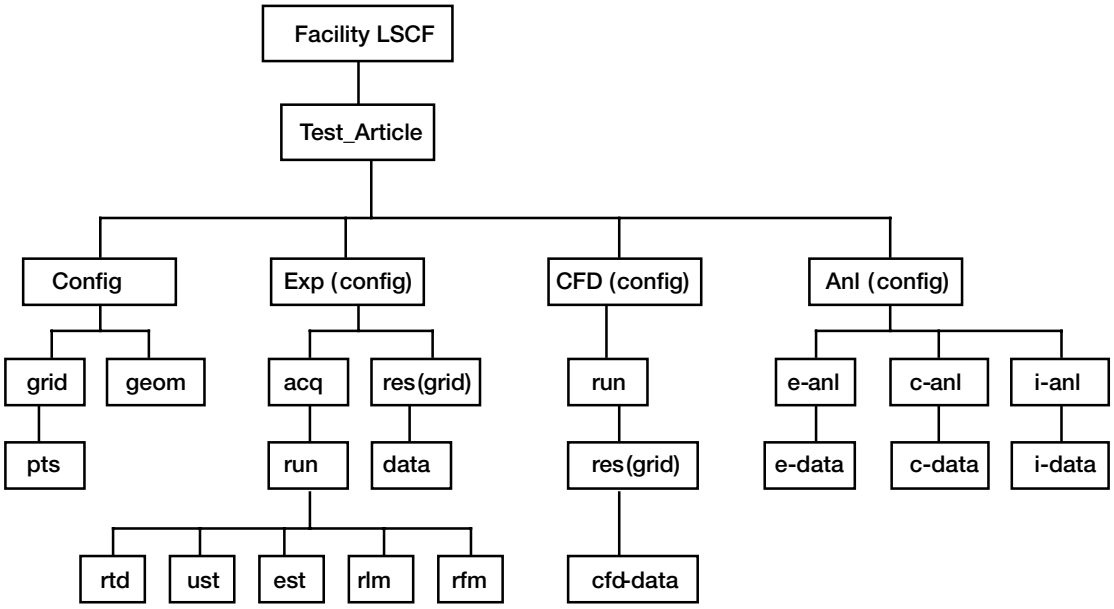


Figure 17.—Example of LSCF heirarchy.

Class

ID	name of entity being defined
Type	kind of class. Value can only be Unix at this point. The original intent was to provide support for relational databases serving as the knowledge-base keeper as well. Unix simply outperformed the relational database tested and became our standard.
Depth	hierarchical level of current class in knowledge-base tree
Genealogy	path taken through tree to reach node being defined
Process	computer process to be executed when current class is encountered

Members

Type	kind of member. Can be STRING_OF_CHARACTERS, INTEGER, FLOATING_POINT_SINGLE, KB_REFERENCE, ENUMERATED_STRING, DATA_FILE, DATE_TIME. Arrays are specified as follows: FLOATING_POINT_SINGLE(xxx) and INTEGER(xxx), where xxx is the required dimensionality.
Description	textual information about what entity represents
EditEnable	flag indicating whether or not user can edit this parameter
Display	number of digits to show user
Precision	number of decimal places to show
StringSize	number of characters in string
CharSets	allowable set of characters for this member
WordSet	allowable set of words for this member
Genealogy	path taken through tree to reach node being defined
MinVal	minimum value
MaxVal	maximum value
MinAlarm	minimum value this member can attain before alarm is raised
MaxAlarm	maximum value this member can attain before alarm is raised
Implicit	polynomial coefficients for implicit data generation

Data Files

ID	Name
Type	Unix
Depth	not applicable
Genealogy	not applicable
FileSpec	shows generic path through knowledge base associated with an instance of this file
InitialProcess	identification of process to execute if data file does not exist. Will generate data.
UpdateProcess	identification of process to execute prior to each data file read
AnalysisProcess	identification of process to execute for ICE_VISUAL analysis

Example

```
rem /****** define root level ******/

class LSCF
type UNIX
depth 1
geneology LSCF
file_spec LSCF.kb

member TA
type KB_RECORD

rem /****** define second level of tree (hence depth=2) ******/

class TA
type UNIX
depth 2
geneology LSCF.ta
file_spec ta_~1.kb

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description entry name

member config
type KB_RECORD
description test article configuration kb records

member cfd
type KB_RECORD
description cfd kb records

member exp
type KB_RECORD
description experiment kb records

member anl
type KB_RECORD
description analysis kb records

rem /****** Let ICE know what the KB_RECORDs can contain ******/
rem /****** That is, what do the classes look like?   ******/

class config
type UNIX
depth 3
genealogy LSCF.ta.config
file_spec config_~1_~2.kb
```

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of this test article configuration
would have to be entered into a text file for parsing.

member rotor_id
type STRING_OF_CHARACTERS
string_size 64
display 64
description name of the rotor

member rotor_blades
type INTEGER
display 4
description number of rotor blades

member rotor_tip
type FLOATING_POINT_SINGLE
display 6
precision 2
description rotor tip clearance

member stator_id
type STRING_OF_CHARACTERS
string_size 64
display 64
description name of the stator

member stator_blades
type INTEGER
display 4
description number of stator blades

member stator_tip
type FLOATING_POINT_SINGLE
display 6
precision 2
description stator tip clearance

member stages
type INTEGER
display 4
description number of stages

member geom
type DATA_FILE
description file of blade surface geometry

rem /****** a sub-class contained within this class ******/
member grid
type KB_RECORD
description grids which support this configuration

```
rem /****** define the new class now ******/
class grid
type UNIX
depth 4
genealogy LSCF.ta.config.grid
file_spec grid_~1_~2_~3.kb

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of the grid

member generator
type STRING_OF_CHARACTERS
string_size 64
display 64
description path to the grid generator

member type
type EMNUMERATED_STRING
word_set grid_type_def
description (cartesian, cylindrical)

member num_points
type INTEGER(3)
display 4
description number of points along axis 1, 2, 3 respectively

member points
type DATA_FILE
description file of grid points

rem /****** sub-classes ******/
class points
type UNIX
depth 5
analysis_process ice_idds
geneology LSCF.ta.config.grid.points
file_spec pts_~1_~2_~3_~4.kb

member points:coords
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description number of points along asix 1, 2, 3, respectively

class geom
type UNIX
depth 4
geneology LSCF.ta.config.geom
file_spec geom_~1_~2_~3.kb
```

```
member geom:axial_coord
type FLOATING_POINT_SINGLE
display 6
precision 2
description axial coordinate of blade surface location

member geom:radial_coord
type FLOATING_POINT_SINGLE
display 6
precision 2
description radial coordiante of blade surface location

member geom:angular_1_coord
type FLOATING_POINT_SINGLE
display 6
precision 2
description angular coordinate of blade surface 1 location

member geom:angular_2_coord
type FLOATING_POINT_SINGLE
display 6
precision 2
description angular coordinate of blade surface 2 location

member geom:thickness
type FLOATING_POINT_SINGLE
display 6
precision 2
description thickness of the blade at this location

rem /****** the above statements define the left most tree ******/
rem /****** structure. Must now define the next structure ******/
rem
rem /******      EXPERIMENT          ******/

class exp
type UNIX
depth 3
genealogy LSCF.ta.exp
file_spec exp_~1_~2.kb

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of this experiment

member engineer
type STRING_OF_CHARACTERS
string_size 64
display 64
description responsible person
```

```
member ta_config
type KB_REFERENCE
genealogy LSCF.ta.config
description configurations which this experiment supports

member exp:acq
type KB_RECORD
description acquisition sub-structure

member exp:res
type KB_RECORD
description calculation results kb records

class exp:res
type UNIX
depth 4
genealogy LSCF.ta.exp.exp:res
file_spec exp:res_~1_~2_~3.kb

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of this result set

member run_time
type FLOATING_POINT_SINGLE
display 6
precision 2
description total facility run time used (sec)

member grid
type KB_REFERENCE
genealogy LSCF.ta.config.grid
description grid used for these results

member exp:data
type DATA_FILE
description results data file

class exp:data
type UNIX
depth 5
genealogy LSCF.ta.exp.exp:res.exp:data
file_spec exp:res_~1_~2_~3_~4.kb

member exp:data:time
type FLOATING_POINT_SINGLE
display 6
precision 4
description simulation time of result (sec)
```

member exp:data:density
type FLOATING_POINT_SINGLE
display 6
precision 3
description density at this grid point

member exp:data:velocity
type FLOATING_POINT_SINGLE(3)
display 6
precision 3
description velocity on axis 1, 2, 3 respectively

member exp:data:energy
type FLOATING_POINT_SINGLE
display 6
precision 3
description energy at this grid point

class exp:acq
type UNIX
depth 4
genealogy LSCF.ta.exp.exp:acq
file_spec exp:res_~1_~2_~3.kb
member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of this acquisition

member passages
type INTEGER
display 4
min_val 1
max_val 128
description number of flow passages

member windows/passage
type INTEGER
display 4
min_val 1
max_val 256
description psg=win/win_per_passage

member lfa_number_channels
type INTEGER
display 2
description number of axial directions

member save_dt
type INTEGER
display 2
description number of DT channels to record

member save_aux
type INTEGER
display 2
description number of AUX channels to record

member processor
type ENUMERATED_STRING
word_set processor_type_def
description LFA proc (macrodyne, MACRODYNE, Macrodyne, m,M, TSI,tsi,t,T)

member exp:run
type KB_RECORD
description run setup record

class exp:run
type UNIX
depth 5
genealogy LSCF.ta.exp.exp:acq.exp:run
file_spec exp:res_~1_~2_~3_~4.kb
process LSCF_acq

member run_number
type INTEGER
display 8
description number of this run

member engineer
type STRING_OF_CHARACTERS
string_size 64
display 64
description responsible person

member stat_seed
type STRING_OF_CHARACTERS
string_size 64
display 64
description kb_path of initial ust, or ‘NONE’

member raw_data_seed
type STRING_OF_CHARACTERS
string_size 64
display 64
description kb_path of raw data, or ‘ON_LINE’

member sample_limit
type INTEGER
display 4
description maximum samples per window

member counts/revolution
type INTEGER
display 5
description win=cnts/ents_per_rev

member cpr_tolerance
type INTEGER
display 6
description end-of-rev encoder count tolerance

member cpr_divider
type INTEGER
display 5
description encoder counts scalar

member velocity_limits_1
type FLOATING_POINT_SINGLE(2)
display 6
precision 4
description min and max velocity permissible for channel 1

member velocity_limits_2
type FLOATING_POINT_SINGLE(2)
display 6
precision 4
description min and max velocity permissible for channel 2

member velocity_limits_3
type FLOATING_POINT_SINGLE(2)
display 6
precision 4
description min and max velocity permissible for channel 3

member inlet_mach
type FLOATING_POINT_SINGLE
display 6
precision 2
description free-stream mach number

member angle_of_attack
display 64
description responsible person

member stat_seed
type STRING_OF_CHARACTERS
string_size 64
display 64
description kb_path of initial ust, or ‘NONE’

member raw_data_seed
type STRING_OF_CHARACTERS
string_size 64
display 64
description kb_path of raw data, or ‘ON_LINE’

member sample_limit
type INTEGER
display 4
description maximum samples per window

member counts/revolution
type INTEGER
display 5
description win=cnts/ents_per_rev

member cpr_tolerance
type INTEGER
display 6
description end-of-rev encoder count tolerance

member cpr_divider
type INTEGER
display 5
description encoder counts scalar

member velocity_limits_1
type FLOATING_POINT_SINGLE(2)
display 6
precision 4
description min and max velocity permissible for channel 1

member velocity_limits_2
type FLOATING_POINT_SINGLE(2)
display 6
precision 4
description min and max velocity permissible for channel 2

member velocity_limits_3
type FLOATING_POINT_SINGLE(2)
display 6
precision 4
description min and max velocity permissible for channel 3

member inlet_mach
type FLOATING_POINT_SINGLE
display 6
precision 2
description free-stream mach number

member angle_of_attack
type FLOATING_POINT_SINGLE
display 6
precision 2
description inlet angle of attack (deg)

member inlet_pressure
type FLOATING_POINT_SINGLE
display 6
precision 4
description inlet pressure setpoint (psi)

member barometric_pressure
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal barometric pressure (inches)

member inlet_temperature
type FLOATING_POINT_SINGLE
display 6
precision 4
description inlet temperature setpoint (degR)

member pressure_ratio
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal pressure ratio

member temperature_ratio
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal temperature ratio

member speed
type FLOATING_POINT_SINGLE
display 8
precision 6
description TA rotor speed setpoint (RPM)

member massflow
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal massflow

member humidity
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal humidity (%)

member total_temp
type FLOATING_POINT_SINGLE
display 6
precision 4
description inlet total temperature (deg R)

member probe
type FLOATING_POINT_SINGLE
display 6
precision 2
description inlet angle of attack (deg)

member inlet_pressure
type FLOATING_POINT_SINGLE
display 6
precision 4
description inlet pressure setpoint (psi)

member barometric_pressure
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal barometric pressure (inches)

member inlet_temperature
type FLOATING_POINT_SINGLE
display 6
precision 4
description inlet temperature setpoint (degR)

member pressure_ratio
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal pressure ratio

member temperature_ratio
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal temperature ratio

member speed
type FLOATING_POINT_SINGLE
display 8
precision 6
description TA rotor speed setpoint (RPM)

member massflow
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal massflow

member humidity
type FLOATING_POINT_SINGLE
display 6
precision 4
description nominal humidity (%)

member total_temp
type FLOATING_POINT_SINGLE
display 6
precision 4
description inlet total temperature (deg R)

member probe
type FLOATING_POINT_SINGLE(3)
display 6
precision 4
description probe location (R, Z, Theta)

member span
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent span

member chord
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent chord

member exp:data:velocity
type FLOATING_POINT_SINGLE(3)
display 6
precision 3
description velocity on axis 1, 2, 3 respectively

member pitch
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent pitch

member gap
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent gap

member beam_1
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description orientation of beam 1 (a, b, c)

member beam_2
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description orientation of beam 2 (a, b, c)

member beam_3
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description orientation of beam 3 (a, b, c)

member pmt_voltage
type FLOATING_POINT_SINGLE(3)
display 8
precision 6
description PMT supply voltage

member laser_power
type FLOATING_POINT_SINGLE(3)
display 6
precision 4
description probe location (R, Z, Theta)

member span
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent span

member chord
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent chord

member exp:data:velocity
type FLOATING_POINT_SINGLE(3)
display 6
precision 3
description velocity on axis 1, 2, 3 respectively

member pitch
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent pitch

member gap
type FLOATING_POINT_SINGLE
display 6
precision 4
description percent gap

member beam_1
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description orientation of beam 1 (a, b, c)

member beam_2
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description orientation of beam 2 (a, b, c)

member beam_3
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description orientation of beam 3 (a, b, c)

member pmt_voltage
type FLOATING_POINT_SINGLE(3)
display 8
precision 6
description PMT supply voltage

member laser_power
type FLOATING_POINT_SINGLE(3)
display 8
precision 3
description power of the laser

member hub
type FLOATING_POINT_SINGLE
display 6
precision 3
description hub location

member tip
type FLOATING_POINT_SINGLE
display 6
precision 3
description tip location

member trailing_edge
type FLOATING_POINT_SINGLE
display 6
precision 4
description trailing edge location

member leading_edge
type FLOATING_POINT_SINGLE
display 6
precision 3
description leading edge location

member rotor_suction_surface
type FLOATING_POINT_SINGLE
display 8
precision 3
description rotor suction surface location

member rotor_pressure_surface
type FLOATING_POINT_SINGLE
display 6
precision 3
description rotor pressure surface location

member stator_suction_surface
type FLOATING_POINT_SINGLE
display 8
precision 3
description stator suction surface location

member stator_pressure_surface
type FLOATING_POINT_SINGLE
display 6
precision 3
description stator pressure surface location

member start_win
type INTEGER
display 4
description 1st window in 1st passage (wrapping)

member first_bl
type FLOATING_POINT_SINGLE(3)
display 8
precision 3
description power of the laser

member hub
type FLOATING_POINT_SINGLE
display 6
precision 3
description hub location

member tip
type FLOATING_POINT_SINGLE
display 6
precision 3
description tip location

member trailing_edge
type FLOATING_POINT_SINGLE
display 6
precision 4
description trailing edge location

member leading_edge
type FLOATING_POINT_SINGLE
display 6
precision 3
description leading edge location

member rotor_suction_surface
type FLOATING_POINT_SINGLE
display 8
precision 3
description rotor suction surface location

member rotor_pressure_surface
type FLOATING_POINT_SINGLE
display 6
precision 3
description rotor pressure surface location

member stator_suction_surface
type FLOATING_POINT_SINGLE
display 8
precision 3
description stator suction surface location

member stator_pressure_surface
type FLOATING_POINT_SINGLE
display 6
precision 3
description stator pressure surface location

member start_win
type INTEGER
display 4
description 1st window in 1st passage (wrapping)

member first_blank_win
type INTEGER
display 4
description first window for blanking data

member last_blank_win
type INTEGER
display 4
description last window for blanking data

member interrupt_marker
type STRING_OF_CHARACTERS
string_size 1
display 1
description null=normal,P=paused then continued,R=restarted,T=terminated

member lfa_gain
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description signal amplification

member lfa_high_amplitude
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description high-amplification cutoff (hertz)

member lfa_low_amplitude
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description low-amplification cutoff (hertz)

member lfa_high_pass
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description high-pass filter bandwidth (hertz)

member lfa_low_pass
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description low-pass filter bandwidth (hertz)

member lfa_wavelength
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description wavelength (1, 2, 3) (nanometers)

member lfa_crossing_angle
type FLOATING_POINT_SINGLE(3)
display 8
precision 3
description crpssomg-angle (degrees)

member lfa_Bragg_frequency
type FLOATING_POINT_SINGLE(3)
display 8
precision 2
display 4
description first window for blanking data

member last_blank_win
type INTEGER
display 4
description last window for blanking data

member interrupt_marker
type STRING_OF_CHARACTERS
string_size 1
display 1
description null=normal,P=paused then continued,R=restarted,T=terminated

member lfa_gain
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description signal amplification

member lfa_high_amplitude
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description high-amplification cutoff (hertz)

member lfa_low_amplitude
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description low-amplification cutoff (hertz)

member lfa_high_pass
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description high-pass filter bandwidth (hertz)

member lfa_low_pass
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
description low-pass filter bandwidth (hertz)

member lfa_wavelength
type FLOATING_POINT_SINGLE(3)
display 8
precision 4
description wavelength (1, 2, 3) (nanometers)

member lfa_crossing_angle
type FLOATING_POINT_SINGLE(3)
display 8
precision 3
description crpssomg-angle (degrees)

member lfa_Bragg_frequency
type FLOATING_POINT_SINGLE(3)
display 8
precision 2
description Bragg frequency

member lfa_number_fringes
type INTEGER(3)
display 6
description number of fringes counted for LA channels 1, 2, 3

member lfa_comp_tolerance
type FLOATING_POINT_SINGLE(3)
display 8
precision 2
description comparison tolerance

member lfa_coin_width
type FLOATING_POINT_SINGLE
display 8
precision 2
description coincident window width

member survey_mode
type STRING_OF_CHARACTERS
string_size 1
display 1
description A=Automated, M=Manual

member survey_type
type STRING_OF_CHARACTERS
string_size 4
display 4
description survey type (NIJK, INJK, etc.)

member survey_file
type STRING_OF_CHARACTERS
string_size 64
display 64
description survey file name

member number_stations
type INTEGER
display 5
description number of survey stations

member current_station
type INTEGER
display 5
description current survey station

member num_beam_paths
type INTEGER
display 5
description number of beam paths at each survey station

member current_beam_path
type INTEGER
display 5
description current beam path number

member i_survey
type INTEGER
display 5
description Bragg frequency

member lfa_number_fringes
type INTEGER(3)
display 6
description number of fringes counted for LA channels 1, 2, 3

member lfa_comp_tolerance
type FLOATING_POINT_SINGLE(3)
display 8
precision 2
description comparison tolerance

member lfa_coin_width
type FLOATING_POINT_SINGLE
display 8
precision 2
description coincident window width

member survey_mode
type STRING_OF_CHARACTERS
string_size 1
display 1
description A=Automated, M=Manual

member survey_type
type STRING_OF_CHARACTERS
string_size 4
display 4
description survey type (NIJK, INJK, etc.)

member survey_file
type STRING_OF_CHARACTERS
string_size 64
display 64
description survey file name

member number_stations
type INTEGER
display 5
description number of survey stations

member current_station
type INTEGER
display 5
description current survey station

member num_beam_paths
type INTEGER
display 5
description number of beam paths at each survey station

member current_beam_path
type INTEGER
display 5
description current beam path number

member i_survey
type INTEGER
display 5
description I-index of survey station

```
member j_survey
type INTEGER
display 5
description J-index of survey station

member k_survey
type INTEGER
display 5
description K-index of survey station

member rtd
type DATA_FILE
description file of run time data

member ust
type DATA_FILE
description file of unaveraged statistics

member est
type DATA_FILE
description file of ensemble averaged statistics

member rlm
type DATA_FILE
description file of raw laser measurements

member rfm
type DATA_FILE
description file of raw facility measurements

rem /****** another sub-class ******/

class rtd
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.rtd
file_spec rtd_~1_~2_~3_~4_~5_~6.kb

member rtd:run_start
type DATE_TIME
display 20
description run_mode start date and time

member rtd:run_duration
type INTEGER
display 6
min_val 0
max_val 3600
description duration of run mode (seconds)

member rtd:data_rate
type INTEGER(3)
display 5
description rate of data for each LA channel
```

member rtd:interrupt_start
type DATE_TIME
display 20
description time of a data acquisition run interrupt

member rtd:interrupt_duration
type INTEGER
display 6
description duration of an interrupt (seconds)

class ust
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.ust
file_spec ust_~1_~2_~3_~4_~5_~6.kb

member ust:samples
type FLOATING_POINT_SINGLE
display 4
precision 0
min_val 0.0
max_val 128.0
description number of samples in this statistic

member ust:velocity_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 5000.0
description mean velocity on channel 1 (EU)

member ust:velocity_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 5000.0
description mean velocity on channel 2 (EU)

member ust:velocity_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 5000.0
description mean velocity on channel 3 (EU)

member ust:std_dev_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 1

member ust:std_dev_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
display 20
description time of a data acquisition run interrupt

member rtd:interrupt_duration
type INTEGER
display 6
description duration of an interrupt (seconds)

class ust
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.ust
file_spec ust_~1_~2_~3_~4_~5_~6.kb

member ust:samples
type FLOATING_POINT_SINGLE
display 4
precision 0
min_val 0.0
max_val 128.0
description number of samples in this statistic

member ust:velocity_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 5000.0
description mean velocity on channel 1 (EU)

member ust:velocity_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 5000.0
description mean velocity on channel 2 (EU)

member ust:velocity_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 5000.0
description mean velocity on channel 3 (EU)

```
member ust:std_dev_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 1

member ust:std_dev_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 2

member ust:std_dev_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 3

member ust:window
type INTEGER
implicit 0,1
display 3
rem min_val 0
rem max_val 32768
description window index (implicit)

class est
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.est
file_spec est_~1_~2_~3_~4_~5_~6.kb

member est:samples
type FLOATING_POINT_SINGLE
display 6
precision 5
min_val 0.0
max_val 32768.0
description number of samples in this statistic

member est:velocity_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val -100.0
max_val 400.0
description mean velocity on channel 1 (EU)
```

```
member est:velocity_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val -100.0
max_val 400.0
description mean velocity on channel 2 (EU)

member est:velocity_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val -100.0
max_val 400.0
description mean velocity on channel 3 (EU)

member est:std_dev_1
type FLOATING_POINT_SINGLE
display 8
precision 4
description standard deviation on channel 2

member ust:std_dev_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 3

member ust:window
type INTEGER
implicit 0,1
display 3
rem min_val 0
rem max_val 32768
description window index (implicit)

class est
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.est
file_spec est_~1_~2_~3_~4_~5_~6.kb

member est:samples
type FLOATING_POINT_SINGLE
display 6
precision 5
min_val 0.0
max_val 32768.0
description number of samples in this statistic
```

member est:velocity_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val -100.0
max_val 400.0
description mean velocity on channel 1 (EU)

member est:velocity_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val -100.0
max_val 400.0
description mean velocity on channel 2 (EU)

member est:velocity_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val -100.0
max_val 400.0
description mean velocity on channel 3 (EU)

member est:std_dev_1
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 1

member est:std_dev_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 2

member est:std_dev_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 3

member est:window
type INTEGER
implicit 0,1
display 3
rem min_val 0
rem max_val 256
description window index (implicit)

class rlm
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.rlm
file_spec rlm_~1_~2_~3_~4_~5_~6.kb

member rlm:delta_t
type INTEGER
display 5
min_val 0
max_val 100
description elapsed time from last sample (10 microsec)

member rlm:window
type INTEGER
display 5
min_val 0
max_val 16384
description window (counts)

member rlm:velocity_1
type INTEGER
display 5
min_val 0
max_val 16384
description velocity on channel 1 (counts)

member rlm:velocity_2
type INTEGER
display 5
min_val 0
max_val 16384
min_val 0.0
max_val 200.0
description standard deviation on channel 1

member est:std_dev_2
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 2

member est:std_dev_3
type FLOATING_POINT_SINGLE
display 8
precision 4
min_val 0.0
max_val 200.0
description standard deviation on channel 3

member est:window
type INTEGER
implicit 0,1
display 3
rem min_val 0
rem max_val 256
description window index (implicit)

class rlm
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.rlm
file_spec rlm_~1_~2_~3_~4_~5_~6.kb

member rlm:delta_t
type INTEGER
display 5
min_val 0
max_val 100
description elapsed time from last sample (10 microsec)

member rlm:window
type INTEGER
display 5
min_val 0
max_val 16384
description window (counts)

member rlm:velocity_1
type INTEGER
display 5
min_val 0
max_val 16384
description velocity on channel 1 (counts)

member rlm:velocity_2
type INTEGER
display 5
min_val 0
max_val 16384
description velocity on channel 2 (counts)

member rlm:velocity_3
type INTEGER
display 5
min_val 0
max_val 16384
description velocity on channel 3 (counts)

member rlm:dt_1
type INTEGER
display 5
min_val 0
max_val 16384
description data for DT channel 1 (counts)

member rlm:dt_2
type INTEGER
display 5
min_val 0
max_val 16384
description data for DT channel 2 (counts)

member rlm:dt_3
type INTEGER
display 5
min_val 0
max_val 16384
description data for DT channel 3 (counts)

member rlm:aux_1
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 1 (counts)

member rlm:aux_2
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 2 (counts)

member rlm:aux_3
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 3 (counts)

member rlm:aux_4
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 4 (counts)

member rlm:aux_5
type INTEGER
description velocity on channel 2 (counts)

member rlm:velocity_3
type INTEGER
display 5
min_val 0
max_val 16384
description velocity on channel 3 (counts)

member rlm:dt_1
type INTEGER
display 5
min_val 0
max_val 16384
description data for DT channel 1 (counts)

member rlm:dt_2
type INTEGER
display 5
min_val 0
max_val 16384
description data for DT channel 2 (counts)

member rlm:dt_3
type INTEGER
display 5
min_val 0
max_val 16384
description data for DT channel 3 (counts)

member rlm:aux_1
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 1 (counts)

member rlm:aux_2
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 2 (counts)

member rlm:aux_3
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 3 (counts)

member rlm:aux_4
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 4 (counts)

member rlm:aux_5
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 5 (counts)

member rlm:aux_6
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 6 (counts)

member rlm:aux_7
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 7 (counts)

member rlm:aux_8
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 8 (counts)

member rlm:aux_9
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 9 (counts)

class rfm
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.rfm
file_spec rfm_~1_~2_~3_~4_~5_~6.kb

member delta_t
type INTEGER
display 4
min_val 0
max_val 600
description elapsed time from last sample (sec)

member inlet_pressure
type INTEGER
display 5
min_val 0
max_val 100000
description inlet total pressure (counts)

member inlet_temp
type INTEGER
display 5
min_val 0
max_val 100000
display 5

min_val 0
max_val 16384
description data for auxiliary channel 5 (counts)

member rlm:aux_6
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 6 (counts)

member rlm:aux_7
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 7 (counts)

member rlm:aux_8
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 8 (counts)

member rlm:aux_9
type INTEGER
display 5
min_val 0
max_val 16384
description data for auxiliary channel 9 (counts)

class rfm
type UNIX
depth 6
genealogy LSCF.ta.exp.exp:acq.exp:run.rfm
file_spec rfm_~1_~2_~3_~4_~5_~6.kb

member delta_t
type INTEGER
display 4
min_val 0
max_val 600
description elapsed time from last sample (sec)

member inlet_pressure
type INTEGER
display 5
min_val 0
max_val 100000
description inlet total pressure (counts)

```
member inlet_temp
type INTEGER
display 5
min_val 0
max_val 100000
description inlet total temperature (counts)

member speed
type INTEGER
display 5
min_val 0
max_val 20000
description TA rotational speed (counts)

member humidity
type INTEGER
display 5
min_val 0
max_val 16384
description facility humidity (counts)

member barometric_pressure
type INTEGER
display 5
min_val 0
max_val 100000
description barometric pressure (counts)

member massflow
type INTEGER
display 5
min_val 0
max_val 100000
description massflow (counts)

member pressure_ratio
type INTEGER
display 5
min_val 0
max_val 100000
description ratio of inlet/outlet pressure (counts)

member temp_ratio
type INTEGER
display 5
min_val 0
max_val 100000
description ratio of inlet/outlet temperature (counts)

rem /******        CFD Definitions       ******/
class cfd
type UNIX
depth 3
genealogy LSCF.ta.cfd
file_spec cfd_~1_~2.kb
```

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of this calculation

description inlet total temperature (counts)

member speed
type INTEGER
display 5
min_val 0
max_val 20000
description TA rotational speed (counts)

member humidity
type INTEGER
display 5
min_val 0
max_val 16384
description facility humidity (counts)

member barometric_pressure
type INTEGER
display 5
min_val 0
max_val 100000
description barometric pressure (counts)

member massflow
type INTEGER
display 5
min_val 0
max_val 100000
description massflow (counts)

member pressure_ratio
type INTEGER
display 5
min_val 0
max_val 100000
description ratio of inlet/outlet pressure (counts)

member temp_ratio
type INTEGER
display 5
min_val 0
max_val 100000
description ratio of inlet/outlet temperature (counts)

```
rem /******        CFD Definitions       ******/
class cfd
type UNIX
depth 3
genealogy LSCF.ta.cfd
file_spec cfd_~1_~2.kb

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of this calculation

member engineer
type STRING_OF_CHARACTERS
string_size 64
display 64
description responsible person

member code
type STRING_OF_CHARACTERS
string_size 64
display 64
description path to cfd code

member ta_config
type KB_REFERENCE
genealogy LSCF.ta.config
rem tells us the reference configuration....

member cfd:run
type KB_RECORD
description calculation results kb records

class cfd:run
type UNIX
depth 4
genealogy LSCF.ta.cfd.cfd:run
file_spec cfd:run_~1_~2_~3.kb
process LSCF_cfd

member inlet_mach
type FLOATING_POINT_SINGLE
display 6
precision 2
description free stream mach number

member angle_of_attack
type FLOATING_POINT_SINGLE
display 6
precision 2
description inlet angle of attack (deg)
```

```
member nominal_pressure
type FLOATING_POINT_SINGLE
display 8
precision 2
description inlet pressure setpoint (psi)

member nominal_temperature
type FLOATING_POINT_SINGLE
display 8
precision 2
description inlet temperature setpoint (degF)

member nominal_speed
type FLOATING_POINT_SINGLE
display 8
precision 2
description TA rotor speed setpoint (rpm)

member engineer
type STRING_OF_CHARACTERS
string_size 64
display 64
description responsible person

member code
type STRING_OF_CHARACTERS
string_size 64
display 64
description path to cfd code

member ta_config
type KB_REFERENCE
genealogy LSCF.ta.config
rem tells us the reference configuration....

member cfd:run
type KB_RECORD
description calculation results kb records

class cfd:run
type UNIX
depth 4
genealogy LSCF.ta.cfd.cfd:run
file_spec cfd:run_~1_~2_~3.kb
process LSCF_cfd

member inlet_mach
type FLOATING_POINT_SINGLE
display 6
precision 2
description free stream mach number
```

member angle_of_attack
type FLOATING_POINT_SINGLE
display 6
precision 2
description inlet angle of attack (deg)

member nominal_pressure
type FLOATING_POINT_SINGLE
display 8
precision 2
description inlet pressure setpoint (psi)

member nominal_temperature
type FLOATING_POINT_SINGLE
display 8
precision 2
description inlet temperature setpoint (degF)

member nominal_speed
type FLOATING_POINT_SINGLE
display 8
precision 2
description TA rotor speed setpoint (rpm)

member rotational_vel
type FLOATING_POINT_SINGLE(4)
display 8
precision 5
description rotational velocity parameters

member exit_pressure
type FLOATING_POINT_SINGLE(4)
display 8
precision 5
description exit pressure parameters

member machine
type ENUMERATED_STRING
word_set cfd_machine_def
description machine used for these results

member result_file
type STRING_OF_CHARACTERS
string_size 64
display 64
description file containing code results

member input_file
type STRING_OF_CHARACTERS
string_size 64
display 64
description code initialization file

member speed
type INTEGER
display 5
min_val 0
max_val 20000
description TA rotational speed (counts)

member tasks
type INTEGER
display 3
description number of concurrent blade row flow calculations

member swaps
type INTEGER
display 3
min_val 1
max_val 100
description number of data swaps between blade row solutions

member cycles
type INTEGER
display 5
min_val 1
max_val 100
description number of iterations per blade row solution

member inlet_boundary_condition
type INTEGER
display 1
min_val 1
max_val 2
member rotational_vel
type FLOATING_POINT_SINGLE(4)
display 8
precision 5
description rotational velocity parameters

member exit_pressure
type FLOATING_POINT_SINGLE(4)
display 8
precision 5
description exit pressure parameters

member machine
type ENUMERATED_STRING
word_set cfd_machine_def
description machine used for these results

member result_file
type STRING_OF_CHARACTERS
string_size 64
display 64
description file containing code results

member input_file
type STRING_OF_CHARACTERS
string_size 64
display 64
description code initialization file

member speed
type INTEGER
display 5
min_val 0
max_val 20000
description TA rotational speed (counts)

member tasks
type INTEGER
display 3
description number of concurrent blade row flow calculations

member swaps
type INTEGER
display 3
min_val 1
max_val 100
description number of data swaps between blade row solutions

member cycles
type INTEGER
display 5
min_val 1
max_val 100
description number of iterations per blade row solution

member inlet_boundary_condition
type INTEGER
display 1
min_val 1
max_val 2
description 1=total conditions, 2=mass flow conditions

member exit_boundary_condition
type INTEGER
display 1
min_val 1
max_val 2
description 1=impose radial equilibrium

member hub_boundary_condition
type INTEGER
display 1
min_val 1
max_val 2
description 1=extrapolate flow vars based on normal pressure gradient

member shroud_boundary_condition
type INTEGER
display 1
min_val -1
max_val 1
description -1=unshrouded geometry, 1=shrouded geometry

member periodic_boundary_condition
type INTEGER
display 1
min_val 1
max_val 2
description 1=periodic

member blade_boundary_condition
type INTEGER
display 1
min_val 1
max_val 2
description 1-extrapolate flow vars based on normal pressure gradient

member rotational_vel_flag
type INTEGER
display 1
min_val 0
max_val 1
description 0=omega, 1=advance ratio

member exit_pressure_flag
type INTEGER
display 1
min_val 0
max_val 1
description 0=(exit:static/inlet:static), 1=(exit:static/inlet:total)

member viscosity_flag
type INTEGER
display 4
description 0=inviscid flag, 1=viscous alg, >1=inviscid/viscous split

member beginning_i
type INTEGER
display 4
description beginning i location of turbulent viscosity calculation
member iterative_frequency
type INTEGER
display 4
description iterative frequency of turbulent viscosity calculation

member cfl_number
type FLOATING_POINT_SINGLE
display 8
precision 5
description cfl number

member gamma
type FLOATING_POINT_SINGLE
display 8
precision 5
description ratio of specific heats

member gas_constant
type FLOATING_POINT_SINGLE
display 9
precision 5
description gas constant

member reference_diameter
type FLOATING_POINT_SINGLE
display 8
precision 5
description reference diameter

member hub_constant
type FLOATING_POINT_SINGLE
display 8
precision 5
description constant defining inlet boundary condition layer at hub

member hub_exponent
type FLOATING_POINT_SINGLE
display 8
precision 5
description exponent defining inlet boundary condition layer at hub

member shroud_constant
type FLOATING_POINT_SINGLE
display 8
precision 5
description constant defining inlet boundary condition layer at shroud

member shroud_exponent
type FLOATING_POINT_SINGLE
display 8
precision 5
description exponent defining inlet boundary condition layer at shroud

member transition_location
type FLOATING_POINT_SINGLE
display 8
precision 5
description transition location on blade as percent of chord

member iterative_frequency
type INTEGER
display 4
description iterative frequency of turbulent viscosity calculation

member cfl_number
type FLOATING_POINT_SINGLE
display 8
precision 5
description cfl number

member gamma
type FLOATING_POINT_SINGLE
display 8
precision 5
description ratio of specific heats

member gas_constant
type FLOATING_POINT_SINGLE
display 9
precision 5
description gas constant

member reference_diameter
type FLOATING_POINT_SINGLE
display 8
precision 5
description reference diameter

member hub_constant
type FLOATING_POINT_SINGLE
display 8
precision 5
description constant defining inlet boundary condition layer at hub

member hub_exponent
type FLOATING_POINT_SINGLE
display 8
precision 5
description exponent defining inlet boundary condition layer at hub

member shroud_constant
type FLOATING_POINT_SINGLE
display 8
precision 5
description constant defining inlet boundary condition layer at shroud

member shroud_exponent
type FLOATING_POINT_SINGLE
display 8
precision 5
description exponent defining inlet boundary condition layer at shroud

member transition_location
type FLOATING_POINT_SINGLE
display 8
precision 5
description transition location on blade as percent of chord

member viscosity_2
type FLOATING_POINT_SINGLE
display 8
precision 5
description second order artificial viscosity parameter

member viscosity_4
type FLOATING_POINT_SINGLE
display 8
precision 5
description forth order artificial viscosity parameter

member axial_coefficient
type FLOATING_POINT_SINGLE
display 8
precision 5
description axial residual averaging coefficient

member radial_coefficient
type FLOATING_POINT_SINGLE
display 8
precision 5
description radial residual averaging coefficient

member tangential_coefficient
type FLOATING_POINT_SINGLE
display 8
precision 5
description tangential residual averaging coefficient

member cpu_time
type FLOATING_POINT_SINGLE
display 6
precision 3
description cpu time used (sec)

member grid
type KB_REFERENCE
genealogy LSCF.ta.config.grid
description grid used for these results

member cfd:p3d
type DATA_FILE
description results Plot3D data file

class cfd:p3d
type UNIX
depth 6
analysis_process ice_idds
genealogy LSCF.ta.cfd.cfd:run.cfd:p3d
file_spec cfd:data_~1_~2_~3_~4_~5.kb

member cfd:data:time
type FLOATING_POINT_SINGLE
display 6
precision 3
description simulation time of result (sec)

member cfd:data:density
type FLOATING_POINT_SINGLE
member viscosity_2
type FLOATING_POINT_SINGLE
display 8
precision 5
description second order artificial viscosity parameter

member viscosity_4
type FLOATING_POINT_SINGLE
display 8
precision 5
description forth order artificial viscosity parameter

member axial_coefficient
type FLOATING_POINT_SINGLE
display 8
precision 5
description axial residual averaging coefficient

member radial_coefficient
type FLOATING_POINT_SINGLE
display 8
precision 5
description radial residual averaging coefficient

member tangential_coefficient
type FLOATING_POINT_SINGLE
display 8
precision 5
description tangential residual averaging coefficient

member cpu_time
type FLOATING_POINT_SINGLE
display 6
precision 3
description cpu time used (sec)

member grid
type KB_REFERENCE
genealogy LSCF.ta.config.grid
description grid used for these results

member cfd:p3d
type DATA_FILE
description results Plot3D data file

```
class cfd:p3d
type UNIX
depth 6
analysis_process ice_idds
genealogy LSCF.ta.cfd.cfd:run.cfd:p3d
file_spec cfd:data_~1_~2_~3_~4_~5.kb

member cfd:data:time
type FLOATING_POINT_SINGLE
display 6
precision 3
description simulation time of result (sec)

member cfd:data:density
type FLOATING_POINT_SINGLE
display 6
precision 3
description density at this grid point

member cfd:data:velocity
type FLOATING_POINT_SINGLE(3)
display 6
precision 3
description velocity on axis 1, 2, 3 respectively

member cfd:data:energy
type FLOATING_POINT_SINGLE
display 6
precision 3
description energy at this point

rem /******    ANALYSIS Definition   ******/
class anl
type UNIX
depth 3
genealogy LSCF.ta.anl
file_spec anl_~1_~2.kb
process LSCF_anl

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of the analysis

member engineer
type STRING_OF_CHARACTERS
string_size 64
display 64
description responsible person

member ta_config
type KB_REFERENCE
genealogy LSCF.ta.config
```

```
member e-anl
type KB_RECORD
description analysis of ta configuration from experimental results

member c-anl
type KB_RECORD
description analysis of ta configuration from cfd results

member i-anl
type KB_RECORD
description analysis of ta configuration from integrated cfd and exp

class e-anl
type UNIX
depth 4
genealogy LSCF.ta.anl.e-anl
display 6
precision 3
description density at this grid point

member cfd:data:velocity
type FLOATING_POINT_SINGLE(3)
display 6
precision 3
description velocity on axis 1, 2, 3 respectively

member cfd:data:energy
type FLOATING_POINT_SINGLE
display 6
precision 3
description energy at this point

rem /******    ANALYSIS Definition   ******/
class anl
type UNIX
depth 3
genealogy LSCF.ta.anl
file_spec anl_~1_~2.kb
process LSCF_anl

member id
type STRING_OF_CHARACTERS
string_size 64
display 64
description descriptive name of the analysis

member engineer
type STRING_OF_CHARACTERS
string_size 64
display 64
description responsible person

member ta_config
type KB_REFERENCE
genealogy LSCF.ta.config
```

```
member e-anl
type KB_RECORD
description analysis of ta configuration from experimental results

member c-anl
type KB_RECORD
description analysis of ta configuration from cfd results

member i-anl
type KB_RECORD
description analysis of ta configuration from integrated cfd and exp

class e-anl
type UNIX
depth 4
genealogy LSCF.ta.anl.e-anl
file_spec e-anl_~!_~2_~3.kb

rem members.... representing arguments and parameters used for data
rem generation should be defined here and filled in by the
rem exp_data_generator

member e-data
type DATA_FILE
description exp. data formed by filtering the results of multiple exps.

class e-data
type UNIX
depth 5
genealogy LSCF.ta.anl.e-anl.e-data
file_spec e-data_~1_~2_~3_~4.kb

member e-data:dum
type FLOATING_POINT_SINGLE
description dummy variable

class c-anl
type UNIX
depth 4
genealogy LSCF.ta.anl.c-anl
file_spec c-anl_~1_~2_~3.kb

rem members.... representing arguments and parameters used for data
rem generation should be defined here and filled in by the
rem cfd_data_generator

member c-data
type DATA_FILE
description cfd data formed by filtering the results of multiple cfd's.

class c-data
type UNIX
depth 5
genealogy LSCF.ta.anl.c-anl.c-data
file_spec c-data_~1_~2_~3_~4.kb
```

```
member c-data:dum
type FLOATING_POINT_SINGLE
description dummy variable

class i-anl
type UNIX
depth 4
genealogy LSCF.ta.anl.i-anl
file_spec i-anl_~1_~2_~3.kb

rem members.... representing arguments and parameters used for data
rem generation should be defined here and filled in by the
file_spec e-anl_~!_~2_~3.kb

rem members.... representing arguments and parameters used for data
rem generation should be defined here and filled in by the
rem exp_data_generator

member e-data
type DATA_FILE
description exp. data formed by filtering the results of multiple exps.

class e-data
type UNIX
depth 5
genealogy LSCF.ta.anl.e-anl.e-data
file_spec e-data_~1_~2_~3_~4.kb

member e-data:dum
type FLOATING_POINT_SINGLE
description dummy variable

class c-anl
type UNIX
depth 4
genealogy LSCF.ta.anl.c-anl
file_spec c-anl_~1_~2_~3.kb

rem members.... representing arguments and parameters used for data
rem generation should be defined here and filled in by the
rem cfd_data_generator

member c-data
type DATA_FILE
description cfd data formed by filtering the results of multiple cfd's.

class c-data
type UNIX
depth 5
genealogy LSCF.ta.anl.c-anl.c-data
file_spec c-data_~1_~2_~3_~4.kb
```

```
member c-data:dum
type FLOATING_POINT_SINGLE
description dummy variable

class i-anl
type UNIX
depth 4
genealogy LSCF.ta.anl.i-anl
file_spec i-anl_~1_~2_~3.kb

rem members.... representing arguments and parameters used for data
rem generation should be defined here and filled in by the
rem int_data_generator

member i-data
type DATA_FILE
description integrated cfd & exp data

class i-data
type UNIX
depth 5
genealogy LSCF.ta.anl.i-anl.i-data
file_spec i-data_~1_~2_~3_~4.kb

member i-data:dum
type FLOATING_POINT_SINGLE
description dummy variable

rem /************* LSCF Runtime process KB **********************/
rem /*****************************************************************/
rem /*                          */
rem  The following description will be used during an actual
rem  experiment. The data are captured according to this format
rem  and stored in the defined hierarchy.
rem /*                          */
rem /*****************************************************************/
rem /*****************************************************************/

class LSCF~RUN~TIME
type UNIX
depth 1

member acq~monitor
type KB_RECORD

member acq~data
type KB_RECORD

class acq~data
type UNIX
depth 2
```

```
member samp_taken
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
min_val 0.0
max_val 200000.0
description number of LFA samples taken

member samp_valid
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
min_val 0.0
max_val 200000.0
description number of LFA samples within velocity limits for current app
rem int_data_generator

member i-data
type DATA_FILE
description integrated cfd & exp data

class i-data
type UNIX
depth 5
genealogy LSCF.ta.anl.i-anl.i-data
file_spec i-data_~1_~2_~3_~4.kb

member i-data:dum
type FLOATING_POINT_SINGLE
description dummy variable

rem /************* LSCF Runtime process KB **********************/
rem /*****************************************************************/
rem /*                          */
rem  The following description will be used during an actual
rem  experiment. The data are captured according to this format
rem  and stored in the defined hierarchy.
rem /*                          */
rem /*****************************************************************/
rem /*****************************************************************/

class LSCF~RUN~TIME
type UNIX
depth 1

member acq~monitor
type KB_RECORD

member acq~data
type KB_RECORD

class acq~data
type UNIX
depth 2
```

member samp_taken
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
min_val 0.0
max_val 200000.0
description number of LFA samples taken

member samp_valid
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
min_val 0.0
max_val 200000.0
description number of LFA samples within velocity limits for current app

member samp_processed
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
min_val 0.0
max_val 200000.0
description number of LFA samples statistically processed

member windows_done
type FLOATING_POINT_SINGLE(3)
display 6
precision 0
min_val 0.0
max_val 16384.0
description number of windows statistically satisfied

member num_bad_revs
type INTEGER
display 4
description number of bad rotor revolutions

member num_zero_revs
type INTEGER
display 4
description number of zero-data rotor revolutions

member num_bad_vel
type INTEGER(3)
display 4
description number of velocity measurements during bad rotor revolutions

member window_done_list
type INTEGER(16384)
display 1
min_val 0
max_val 16384
description ordered list of completed windows

member acq_start_time_list
type DATE_TIME(128)
display 9
description latest acquisition start times

member acq_duration_list
type INTEGER(128)
display 8
min_val 0
max_val 28800
description latest acquisition durations (sec)

member pressure_samples
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility pressure samples

member samp_processed
type FLOATING_POINT_SINGLE(3)
display 6
precision 2
min_val 0.0
max_val 200000.0
description number of LFA samples statistically processed

member windows_done
type FLOATING_POINT_SINGLE(3)
display 6
precision 0
min_val 0.0
max_val 16384.0
description number of windows statistically satisfied

member num_bad_revs
type INTEGER
display 4
description number of bad rotor revolutions

member num_zero_revs
type INTEGER
display 4
description number of zero-data rotor revolutions

member num_bad_vel
type INTEGER(3)
display 4
description number of velocity measurements during bad rotor revolutions

member window_done_list
type INTEGER(16384)
display 1
min_val 0
max_val 16384
description ordered list of completed windows

member acq_start_time_list
type DATE_TIME(128)
display 9
description latest acquisition start times

member acq_duration_list
type INTEGER(128)
display 8
min_val 0
max_val 28800
description latest acquisition durations (sec)

member pressure_samples
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility pressure samples

member temperature_samples
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 500.0
description circular stack of facility temperature samples

member speed_samples
type FLOATING_POINT_SINGLE(128)
display 6
precision 2
min_val 0.0
max_val 20000.0
description circular stack of facility speed samples

member humidity
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility humidity samples

member bar_pressure
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 1000.0
description circular stack of facility barometric pressure samples

member massflow
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility massflow samples

member pressure_ratio
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility pressure ratios

member temperature_ratio
type FLOATING_POINT_SINGLE(128)
display 6
precision 2
min_val 400.0
max_val 1000.0
description circular stack of facility temperature ratios

member ensemble_samples
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
member temperature_samples
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 500.0
description circular stack of facility temperature samples

member speed_samples
type FLOATING_POINT_SINGLE(128)
display 6
precision 2
min_val 0.0
max_val 20000.0
description circular stack of facility speed samples

member humidity
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility humidity samples

member bar_pressure
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 1000.0
description circular stack of facility barometric pressure samples

member massflow
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility massflow samples

member pressure_ratio
type FLOATING_POINT_SINGLE(128)
display 8
precision 2
min_val 0.0
max_val 100.0
description circular stack of facility pressure ratios

member temperature_ratio
type FLOATING_POINT_SINGLE(128)
display 6
precision 2
min_val 400.0
max_val 1000.0
description circular stack of facility temperature ratios

member ensemble_samples
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 2000.0
description LFA samples indexed to ensemble win

member ensemble_vel_1
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 1 mean vel. indexed to ensemble win

member ensemble_vel_2
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 2 mean vel. indexed to ensemble win

member ensemble_vel_3
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 3 mean vel. indexed to ensemble win

member ensemble_std_1
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 200.0
description channel 1 std deviation indexed to ensemble win

member ensemble_std_2
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 200.0
description channel 2 std deviation indexed to ensemble win

member ensemble_std_3
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 200.0
description channel 3 std deviation indexed to ensemble win

member unaveraged_samples
type FLOATING_POINT_SINGLE(16384)
display 5
precision 2
min_val 0.0
max_val 200.0
description LFA samples linked to unaveraged window

min_val 0.0
max_val 2000.0
description LFA samples indexed to ensemble win

member ensemble_vel_1
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 1 mean vel. indexed to ensemble win

member ensemble_vel_2
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 2 mean vel. indexed to ensemble win

member ensemble_vel_3
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 3 mean vel. indexed to ensemble win

member ensemble_std_1
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 200.0
description channel 1 std deviation indexed to ensemble win

member ensemble_std_2
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 200.0
description channel 2 std deviation indexed to ensemble win

member ensemble_std_3
type FLOATING_POINT_SINGLE(256)
display 8
precision 2
min_val 0.0
max_val 200.0
description channel 3 std deviation indexed to ensemble win

member unaveraged_samples
type FLOATING_POINT_SINGLE(16384)
display 5
precision 2
min_val 0.0
max_val 200.0
description LFA samples linked to unaveraged window

member unaveraged_vel_1
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 1 mean vel indexed to unaveraged win

member unaveraged_vel_2
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 2 mean vel indexed to unaveraged win

member unaveraged_vel_3
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 3 mean vel indexed to unaveraged win

member unaveraged_std_1
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val 0.0
max_val 400.0
description channel 1 std deviation indexed to unaveraged win

member unaveraged_vel_2
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val 0.0
max_val 400.0
description channel 2 std deviation indexed to unaveraged win

member unaveraged_std_3
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val 0.0
max_val 400.0
description channel 3 std deviation indexed to unaveraged win

class acq~monitor
type UNIX
depth 2

member opr~time
type KB_RECORD
description 1:current acquisition

member opr~mode
type KB_RECORD
description 1:acquisition mode

member unaveraged_vel_1
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 1 mean vel indexed to unaveraged win

member unaveraged_vel_2
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 2 mean vel indexed to unaveraged win

member unaveraged_vel_3
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val -100.0
max_val 400.0
description channel 3 mean vel indexed to unaveraged win

member unaveraged_std_1
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val 0.0
max_val 400.0
description channel 1 std deviation indexed to unaveraged win

member unaveraged_vel_2
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val 0.0
max_val 400.0
description channel 2 std deviation indexed to unaveraged win

member unaveraged_std_3
type FLOATING_POINT_SINGLE(16384)
display 8
precision 2
min_val 0.0
max_val 400.0
description channel 3 std deviation indexed to unaveraged win

class acq~monitor
type UNIX
depth 2

member opr~time
type KB_RECORD
description 1:current acquisition

member opr~mode
type KB_RECORD
description 1:acquisition mode

member facility
type KB_RECORD
description 0:facility measurements (deviation from nominal)

member lfa~data
type KB_RECORD
description 0:acquisition status (% complete)

member lfa~update
type KB_RECORD
description 0:average update interval (milliseconds)

member lfa~samples
type KB_RECORD
description 0:laser measurements

class opr~time
type UNIX
depth 3
description 1:current acquisition

member start_time
type DATE_TIME
display 9

member duration(sec)
type INTEGER
display 5

class opr~mode
type UNIX
depth 3

member functional
type STRING_OF_CHARACTERS
string_size 15
display 15

member operational
type STRING_OF_CHARACTERS
string_size 15
display 15

```
class facility
type UNIX
depth 3

member pressure
type FLOATING_POINT_SINGLE
display 8
precision 2
min_val -10.0
max_val 10.0
min_alarm -5.0
max_alarm 5.0
description deviation of measured pressure from nominal

member temperature
type FLOATING_POINT_SINGLE
member facility
type KB_RECORD
description 0:facility measurements (deviation from nominal)

member lfa~data
type KB_RECORD
description 0:acquisition status (% complete)

member lfa~update
type KB_RECORD
description 0:average update interval (milliseconds)

member lfa~samples
type KB_RECORD
description 0:laser measurements

class opr~time
type UNIX
depth 3
description 1:current acquisition

member start_time
type DATE_TIME
display 9

member duration(sec)
type INTEGER
display 5

class opr~mode
type UNIX
depth 3

member functional
type STRING_OF_CHARACTERS
string_size 15
display 15
```

member operational
type STRING_OF_CHARACTERS
string_size 15
display 15

class facility
type UNIX
depth 3

member pressure
type FLOATING_POINT_SINGLE
display 8
precision 2
min_val -10.0
max_val 10.0
min_alarm -5.0
max_alarm 5.0
description deviation of measured pressure from nominal

member temperature
type FLOATING_POINT_SINGLE
display 8
precision 2
min_val -25.0
max_val 25.0
min_alarm -10.0
max_alarm 10.0
description deviation of measured temperature from nominal

member rotor_speed
type FLOATING_POINT_SINGLE
display 8
precision 2
min_val -500.0
max_val 500.0
min_alarm -100.0
max_alarm 100.0
description deviation of measured rotor speed from nominal

member humidity
type FLOATING_POINT_SINGLE
display 4
min_val 0
max_val 100
description facility humidity (counts)

member barometric_pres
type FLOATING_POINT_SINGLE
display 4
min_val 0
max_val 1000
description facility barometric pressure (counts)

member massflow
type FLOATING_POINT_SINGLE
display 5
min_val 0
max_val 100
description facility massflow (counts)

member pressure_ratio
type FLOATING_POINT_SINGLE
display 5
min_val 0
max_val 1000
description facility ratio of inlet/outlet pressure (counts)

member temp_ratio
type FLOATING_POINT_SINGLE
display 5
min_val 0
max_val 1000
description facility ratio of inlet/outlet temperature (counts)

class lfa~data
type UNIX
depth 3
display 8
precision 2
min_val -25.0
max_val 25.0
min_alarm -10.0
max_alarm 10.0
description deviation of measured temperature from nominal

member rotor_speed
type FLOATING_POINT_SINGLE
display 8
precision 2
min_val -500.0
max_val 500.0
min_alarm -100.0
max_alarm 100.0
description deviation of measured rotor speed from nominal

member humidity
type FLOATING_POINT_SINGLE
display 4
min_val 0
max_val 100
description facility humidity (counts)

member barometric_pres
type FLOATING_POINT_SINGLE
display 4
min_val 0
max_val 1000
description facility barometric pressure (counts)

member massflow
type FLOATING_POINT_SINGLE
display 5
min_val 0
max_val 100
description facility massflow (counts)

member pressure_ratio
type FLOATING_POINT_SINGLE
display 5
min_val 0
max_val 1000
description facility ratio of inlet/outlet pressure (counts)

member temp_ratio
type FLOATING_POINT_SINGLE
display 5
min_val 0
max_val 1000
description facility ratio of inlet/outlet temperature (counts)

class lfa~data
type UNIX
depth 3

member samples
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 150.0
min_alarm 0.0
max_alarm 100.0
description percent samples taken of max samples

member windows
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 150.0
min_alarm 0.0
max_alarm 100.0
description percent of windows completed of total windows

class lfa~update
type UNIX
depth 3

member samples
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 100.0
min_alarm 0.0
max_alarm 100.0
description sample update interval (msec)

member statistics
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 100.0
min_alarm 0.0
max_alarm 100.0
description statistics update interval (msec)

class lfa~samples
type UNIX
depth 3

member taken
type FLOATING_POINT_SINGLE(3)
display 8
precision 1
min_val 0.0
max_val 2000000.0

member valid
type FLOATING_POINT_SINGLE(3)
display 8
precision 1
member samples
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 150.0
min_alarm 0.0
max_alarm 100.0
description percent samples taken of max samples

member windows
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 150.0
min_alarm 0.0
max_alarm 100.0
description percent of windows completed of total windows

```
class lfa~update
type UNIX
depth 3

member samples
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 100.0
min_alarm 0.0
max_alarm 100.0
description sample update interval (msec)

member statistics
type FLOATING_POINT_SINGLE(3)
display 6
precision 1
min_val 0.0
max_val 100.0
min_alarm 0.0
max_alarm 100.0
description statistics update interval (msec)

class lfa~samples
type UNIX
depth 3

member taken
type FLOATING_POINT_SINGLE(3)
display 8
precision 1
min_val 0.0
max_val 2000000.0

member valid
type FLOATING_POINT_SINGLE(3)
display 8
precision 1
min_val 0.0
max_val 2000000.0

member processed
type FLOATING_POINT_SINGLE(3)
display 8
precision 1
min_val 0.0
max_val 2000000.0
```

```
rem /************ LSCF Runtime CFD process KB *********************/
rem /*************************************************************/
rem /*                          */
rem  The following description will be used during actual CFD
rem  code executions. The data are captured according to this
rem  format and stored in the defined hierarchy.
rem /*                          */
rem /*************************************************************/
rem /*************************************************************/

class CFD~RUN~TIME
genealogy CFD_RUN_TIME
type UNIX
depth 1

member cfd:monitor
type KB_RECORD

member cfd:variables
type KB_RECORD

class cfd:variables
type UNIX
genealogy CFD_RUN_TIME.cfd:variables
depth 2

member user_mode
type INTEGER
display 2
description code operating mode

member init_file
type STRING_OF_CHARACTERS
string_size 63
display 63
description code initialization file

member result_file
type STRING_OF_CHARACTERS
string_size 63
display 63
description code results file

member actual_mode
type INTEGER
display 2
description LAPIN operating mode

min_val 0.0
max_val 2000000.0
```

```
member processed
type FLOATING_POINT_SINGLE(3)
display 8
precision 1
min_val 0.0
max_val 2000000.0

rem /*********** LSCF Runtime CFD process KB *********************/
rem /*************************************************************/
rem /*                          */
rem  The following description will be used during actual CFD
rem  code executions. The data are captured according to this
rem  format and stored in the defined hierarchy.
rem /*                          */
rem /*************************************************************/
rem /*************************************************************/

class CFD~RUN~TIME
genealogy CFD_RUN_TIME
type UNIX
depth 1

member cfd:monitor
type KB_RECORD

member cfd:variables
type KB_RECORD

class cfd:variables
type UNIX
genealogy CFD_RUN_TIME.cfd:variables
depth 2

member user_mode
type INTEGER
display 2
description code operating mode

member init_file
type STRING_OF_CHARACTERS
string_size 63
display 63
description code initialization file

member result_file
type STRING_OF_CHARACTERS
string_size 63
display 63
description code results file
```

member actual_mode
type INTEGER
display 2
description LAPIN operating mode

member mode
type STRING_OF_CHARACTERS
string_size 15
display 15
description current operating mode

member time_steps
type INTEGER
display 5
min_val 0
max_val 10000
description number of time steps executed

member start_time
type DATE_TIME
display 9
description latest acquisition start time

member sim_time
type FLOATING_POINT_SINGLE
display 9
precision 4
min_val 0.0
max_val 3600.0
description simulation time in seconds

class cfd:monitor
type UNIX
genealogy CFD_RUN_TIME.cfd:monitor
depth 2

member cfd:monitor:status
type KB_RECORD
description 1:current status

member cfd:monitor:operation
type KB_RECORD
description 1:code operation

class cfd:monitor:status
type UNIX
genealogy CFD_RUN_TIME.cfd:monitor.cfd:monitor:status
depth 3

member cfd_mode
type STRING_OF_CHARACTERS
string_size 15
display 15
description current CFD operating mode

member time_steps
type INTEGER
display 5
min_val 0
max_val 10000
description number of time steps executed

member mode
type STRING_OF_CHARACTERS
string_size 15
display 15
description current operating mode

member time_steps
type INTEGER
display 5
min_val 0
max_val 10000
description number of time steps executed

member start_time
type DATE_TIME
display 9
description latest acquisition start time

member sim_time
type FLOATING_POINT_SINGLE
display 9
precision 4
min_val 0.0
max_val 3600.0
description simulation time in seconds

class cfd:monitor
type UNIX
genealogy CFD_RUN_TIME.cfd:monitor
depth 2

member cfd:monitor:status
type KB_RECORD
description 1:current status

member cfd:monitor:operation
type KB_RECORD
description 1:code operation

class cfd:monitor:status
type UNIX
genealogy CFD_RUN_TIME.cfd:monitor.cfd:monitor:status
depth 3

member cfd_mode
type STRING_OF_CHARACTERS
string_size 15
display 15
description current CFD operating mode

member time_steps
type INTEGER
display 5
min_val 0
max_val 10000
description number of time steps executed

class cfd:monitor:operation
type UNIX
genealogy CFD_RUN_TIME.cfd:monitor.cfd:monitor:operation
depth 3

member sim_start
type DATE_TIME
display 9
description latest acquisition start time

member sim_time
type FLOATING_POINT_SINGLE
display 9
precision 4
min_val 0.0
max_val 3600
description simulation time in seconds

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2001	Technical Memorandum

4. TITLE AND SUBTITLE

Overview of ICE Project: Integration of Computational Fluid Dynamics and Experiments

5. FUNDING NUMBERS

WU–704–10–23–00

6. AUTHOR(S)

James D. Stegeman, Richard A. Blech, Theresa L. Babrauckas, and William H. Jones

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

National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–12570

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 TM—2001-210610

11. SUPPLEMENTARY NOTES

Responsible person, James D. Stegeman, organization code 7180, 216–433–3389.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Categories: 05, 31, 07, and 33 Distribution: Nonstandard

Available electronically at http://gltrs.grc.nasa.gov/GLTRS

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

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Researchers at the NASA Glenn Research Center have developed a prototype integrated environment for interactively exploring, analyzing, and validating information from computational fluid dynamics (CFD) computations and experiments. The Integrated CFD and Experiments (ICE) project is a first attempt at providing a researcher with a common user interface for control, manipulation, analysis, and data storage for both experiments and simulation. ICE can be used as a live, on-line system that displays and archives data as they are gathered; as a postprocessing system for dataset manipulation and analysis; and as a control interface or "steering mechanism" for simulation codes while visualizing the results. Although the full capabilities of ICE have not been completely demonstrated, this report documents the current system. Various applications of ICE are discussed: a low-speed compressor, a supersonic inlet, real-time data visualization, and a parallel-processing simulation code interface. A detailed data model for the compressor application is included in the appendix.

14. SUBJECT TERMS

Simulation; Experiment; Knowledge base; Integration; CFD; Real time; Visualization; Data base

15. NUMBER OF PAGES

93

16. PRICE CODE

A05

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