TUNED GRID GENERATION WITH ICEM CFD

Armin Wulf and Vedat Akdag
ICEM CFD Engineering
Berkeley, CA 94704

ABSTRACT

ICEM CFD is a CAD based grid generation package that supports multiblock structured, unstructured tetrahedral and unstructured hexahedral grids. Major development efforts have been spent to extend ICEM CFD's multiblock structured and hexahedral unstructured grid generation capabilities. The modules added are: a parametric grid generation module and a semi-automatic hexahedral grid generation module. A fully automatic version of the hexahedral grid generation module for around a set of predefined objects in rectilinear enclosures has been developed. These modules will be presented and the procedures used will be described, and examples will be discussed.

INTRODUCTION

The ability to accurately create a computational grid about geometrically complex configurations is becoming increasingly important in the analysis world. With ICEM CFD computational grids can be employed to treat complex geometric topologies. ICEM CFD embodies full CAD tools for creating geometries or importing geometry from various CAD systems. Computational grids, including boundary conditions can be generated for over 25 different CFD flow solvers and structural analysis codes [1]. (Figure 1).

Existing methods for multiblock structured computational grid generation codes are generally very time consuming. Current codes also require high level of user expertise in order to achieve optimal usage. Since the rapid construction of suitable multiblock structured computational grids is still one of the pacing issues in CFD applications, additional functionality has been added to ICEM CFD. They are ICEM COMAK for parametric multi-block mesh generation and ICEM HEXA for semi-automatic hexahedral mesh generation. In addition, the fully automatic hexahedral grid generation module called ICEPAK will be presented. ICEPAK has been developed in coorporation with Fluid Dynamics International (fdi) [2] to support thermal management of electronic enclosures. For ICEPAK fully automatic grid generation is possible, because the geometry representation is restricted.

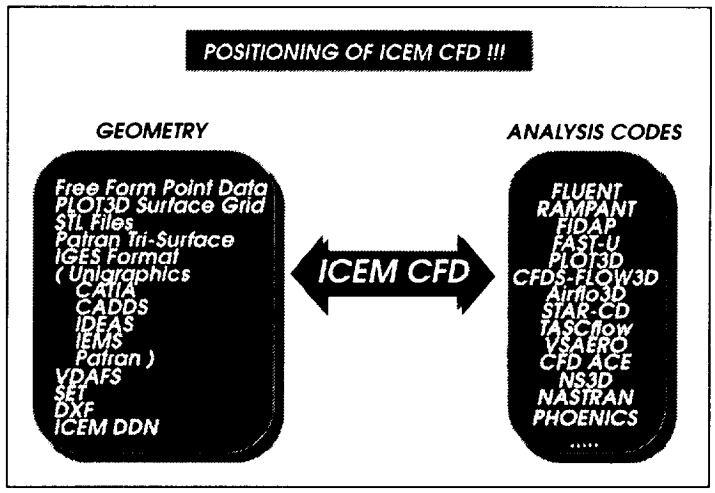

Figure 1: Positioning of ICEM CFD between the geometry and analysis codes.

PARAMETRIC GRID GENERATION USING ICEM COMAK

ICEM COMAK is a parametric multi-block grid generation tool which is an extended option of ICEM CFD's structured grid generation code. Once the mesh of a given configuration is created, using COMAK (COnfiguration MAKer) it is possible to replay the construction process in order to get a mesh of the same topology with geometric differences. It offers two modes; the Specifica-

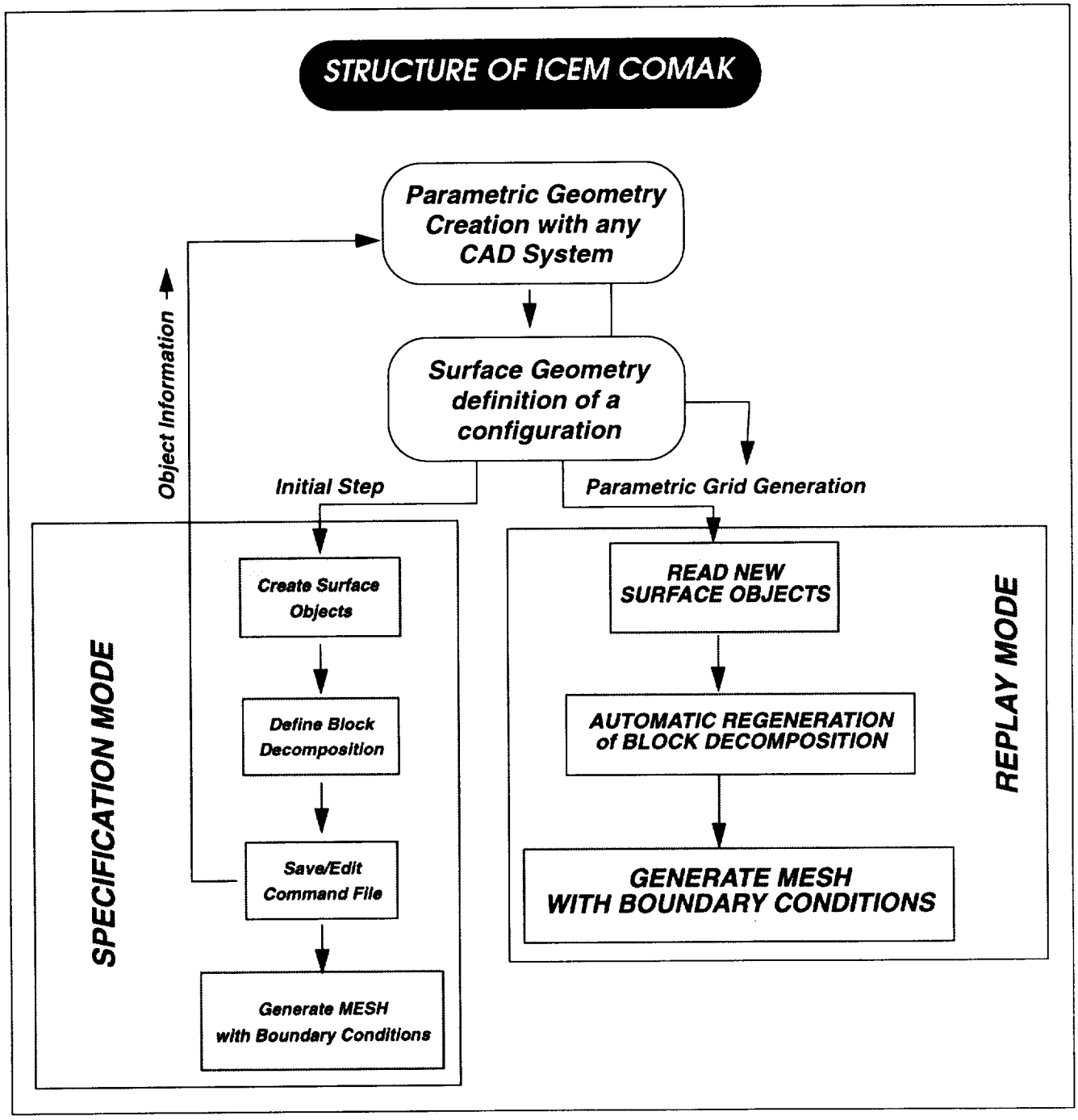

Figure 2: Structure of ICEM COMAK, the parametric grid generator

tion Mode and the Replay Mode. In the Specification Mode the user has a set of tools allowing him to interactively create, manipulate, group and manage geometric entities. The user's actions are stored into a command file. In the Replay Mode, when the user wants to create a mesh around a different object with the same topology, it is only necessary to break the object geometry into parts like the original. Then the user can replay the commands to create the new multi-block structured mesh associated to the new CAD geometry, where sets of geometrical entities can be different. The boundary conditions are set-up automatically. These features allow one to perform very efficient geometric trade studies.

In Figure 2, the structure of ICEM COMAK is shown. CAD systems such as I/EMS from Intergraph or Pro-Engineer from Parametric Technology Corporation are modelers that can create geometries using a parametric approach.

During the initial step or the specification mode, the geometry from the user's CAD system is input in IGES format to the ICEM COMAK environment. The first step is to put surfaces into object groups. The user then composes the computational block structure. During this operation the session commands are recorded in a command file. The file created is saved. After the points along block edges are distributed, the computational grid is created. The boundary conditions for the analysis are also created during this session. User then translates the mesh into the flow solver format of choice.

To perform the parametric mesh calculation for a similar geometry with the same topology but with geometric differences, the user goes back and names the objects in the new geometry with the same object names. After the new IGES file is created, the ICEM Manager grabs this IGES file and writes the CAD data into the object database and initiates ICEM COMAK. COMAK reads the new object files and automatically regenerates the computational grid by updating the computational topology with the new geometry and translates the mesh into the flow solver format. The replay mode can be treated as a continuous loop for computational grid generation in batch mode on modified geometries.

The ICEM Manager is written in TCL/TK [3] a programming system for developing and using graphical user interface applications. It is an easy to use scripting language for controlling and extending applications. The following sample script illustrates the execution of a series of commands in the ICEM Manager to support the automatic replay mode as it is described above. In this example it creates an input for the structural analysis code ANSYS from Swanson Analysis Systems, Inc.

```
# This script shows how to run a complex command from within the manager.
# The actions it performs are:
# 1. Translates two IGES files into DDN parts. The names of these parts
# and the IGES files are known beforehand.
# 2. Run a GPL program on each part, which extracts a set of Comak
# objects.
# 3. Run a Comak script to automatically generate the mesh.
# 4. Write an ANSYS output.

# To make this script available, you should put the following line in your
# ~/.iceman_init file:
#
# if {file exists user_app_script.tcl} { source user_app_script.tcl }
```

```
#
# The rest of this file is the contents of custom_script.tcl, which
# should be in the directory you run icemcfd from.

proc user_app_script {what} {
 global env ddn_path confname geoname partname s2u_path ansys_out-
put_path
 global partname sheetname

 if {($what == "all") | | ($what == "prepare")} {
# First remove all the existing DDN parts and the Comak objects.
set p (glob -nocomplain parts/* objects/$confname/$geoname/*)
if {$p != ""} {
 eval exec /bin/rm -f $p parts/.ddn_directory \
 objects/$confname/$geoname/.ddn_directory
}
update_partlist d 1

# Set up the IGES directive file.
exec /bin/echo "CONVERT,NAME > _directive_tmp
exec /bin/echo "define,CREATE_310_PART=1 >> _directive_tmp

# Convert all the parts in the iges directory to DDN parts.
# If any of them have "name" in their filename, run them through
# the gpl program.
set names ""
foreach if (glob iges/*) {
 # Convert the part to IGES.
 set dn (file tail (file rootname $if))
 set command "$env(ICEM_ACN)/iges/iges_post i=iges/$pf name=$dn \
 o=parts/$dn l=iges_list logit=yes d=_directive_tmp"
 runcom $command iges_list
 if (string match "name" $dn) { lappend names $dn }
}
update_partlist d 1

# Make a DDN command file that runs the GPL program and then exits.
set restart (open .restart.tmp w)
puts $restart "F.5.13.5.3.(pwd)"
puts $restart "F.5.13.3.kim1_gpl"
puts $restart "F.4.7.y"
close $restart

foreach pf $names {
 # Run the GPL program.
 set_part_partfile parts/$pf
 if {$partname == ""} { error "No part" }
 set command "$ddn_path db=parts pn=\"$partname\" sn=1 \
 i=.restart.tmp (get_ddn_defaults)"
 runxcom USER_APP_GPL $command "" 0 0
 update_partlist d 1
}

# Now move all the parts to the Comak directory.
foreach pf (glob parts/*) {
 if (string match "name" $pf) continue
 exec /bin/cp $pf objects/$confname/$geoname
}
update_partlist k 1

# Clear out the old domains.
set doms (glob -nocomplain mulcad/$confname/$geoname/domains/*)
if {$doms != ""} { eval exec /bin/rm -f $doms }
 }

 if 0 {
if {($what == "all") | | ($what == "comak")} {
 set extras ""
 # Now run the Comak job. These key sequences are the commands that
the
 # user would type in to Mulcad to perform the indicated actions.
{ 52))Run the key.cmd file. }
{ nnnDon't modify parameters. }
{ ))yExit comak. }
{ 03yy)Update topology. }
{ !253yMesh generation. }
{ )))Back to the top. }
{ 96yExit mulcad. }
 ) {
append extras (lindex $xx 0)
 }

 unselect_part
 catch {exec /bin/rm -f (glob mulcad/$confname/$geoname/parts/test1*)}
 update_partlist m 1
 set partname TEST1
 set sheetname 1

 set key (send_keyboard_events "ICEM 3.1 GRAPHICS" 5 $extras)
 app_mulcad
 exec /bin/rm -f $key
}
 }

 if {($what == "all") | | ($what == "comak")} {
unselect_part
catch {exec /bin/rm -f (glob mulcad/$confname/$geoname/parts/test1*)}
update_partlist m 1
set partname TEST1
set sheetname 1

#1.1.12.).)
set extras {16 5.2.).) ).) y 10.3 Y Y n ) 11.2.1.12.).5.3}
app_mulcad $extras
 }

 if {($what == "all") | | ($what == "output")} {
# Convert the structured domains into an unstructured one.
set doms (glob mulcad/$confname/$geoname/domains/*)
set topofile mulcad/$confname/topology/topo_mulcad_out
set outfile $place/domains/struct_merge
runcom "$s2u_path -t $topofile -o $outfile $doms"

# Now run the ansys converter.
set topofile mulcad/$confname/$geoname/boco
set ansys mulcad/$confname/$geoname/transfer/ANSYS
runcom "$ansys_output_path -dom $outfile -b $bocofile $ansys"

infomsg "Done with conversion."
 }
}

# These lines create a new menu and add the above command to the menu.

make_menu user_app "USER Application" ""
make_entry user_app "Whole script" "" "" "user_app_script all" ""
make_entry user_app "Prepare objects" "" "" "user_app_script prepare" ""
make_entry user_app "Run Comak" "" "" "user_app_script comak" ""
make_entry user_app "Write output" "" "" "user_app_script output" ""
```

Figure 3 illustrates the application of ICEM COMAK to full airplane volume grid configurations [4]. Using the specification mode the volume grid is calculated for the A320 aircraft. The command file is replayed to create the similar grid for the A330 aircraft. Notice the differences in the sizes of the engine and the fuselage also the distances of the pylon from the fuselage as well as the impact on the created grids.

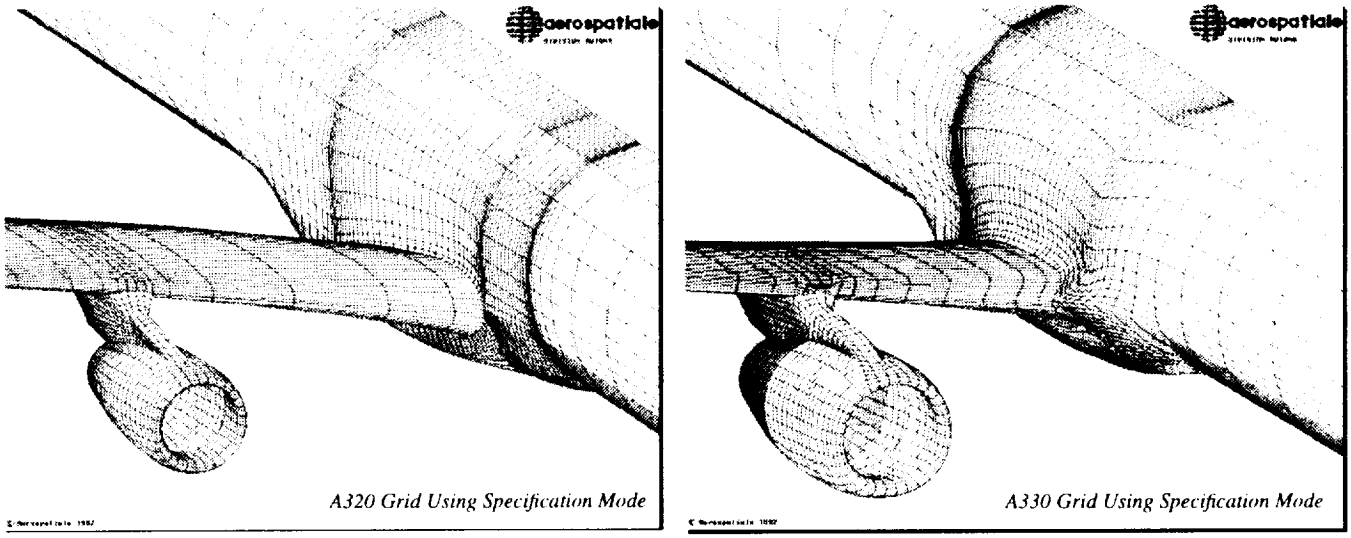

Figure 3: Using ICEM COMAK, computational volume grid for A320 and automatic generation of grid for the A330 (Courtesy of Aerospatiale).

SEMI-AUTOMATIC GRID GENERATION USING ICEM HEXA

ICEM HEXA is an object based semi-automated hexahedral volume mesher for creating multi-block structured meshes or unstructured hexahedral meshes.

The user can define the initial block structure or, alternatively, HEXA will automatically initialize the block structure around a given geometry. Blocks can be interactively adjusted to the underlying CAD geometry. Body fitted internal or external O-Grids can be generated by the system automatically. Mesh sizes can be defined on the object surfaces or individually on the edges using edge meshing options. The grid is projected onto the underlying CAD geometry with minimum user interaction, with complete independence from the orientation of the patches and patch boundaries of the underlying CAD geometry.

Input to ICEM HEXA is CAD geometry, in the form of NURB surfaces, trimmed NURB surface and NURB curves. CAD geometry is either created using ICEM CFD's CAD tool or translated from any other CAD system using IGES or other translator formats. Surface meshes in STL (Stereolithography) format or triangular surface meshes in PATRAN format can also be input as a surface representation for mesh generation.

The following is an example of the grid generation process for a generic chemical processing tank as shown in Figure 4. This configuration contains an inlet, an exhaust port, another cylindrical port for chemical control and two petruding cylinders to the tank. These cylinders are utilized for inspection and controlling the chemical process taking place inside the tank. Each of these cylinders are connected to a smaller cylinders which they connect to pressure regulators. Since this configuration contains many cylinder T-connections, it produces a moderately complex block decomposition strategy for any grid generation system.

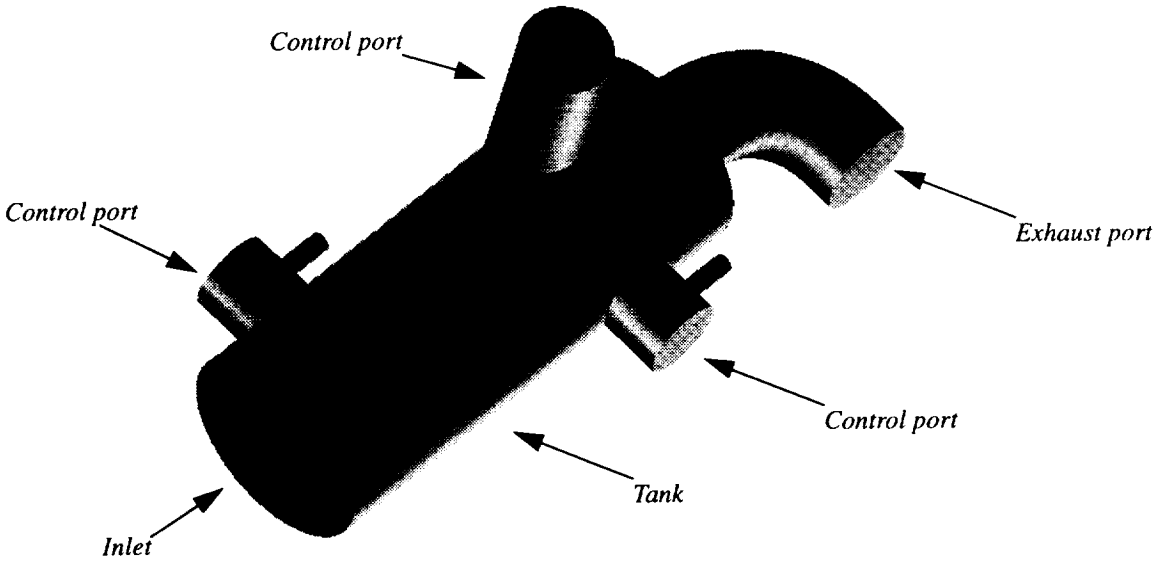

Figure 4: The chemical processing tank

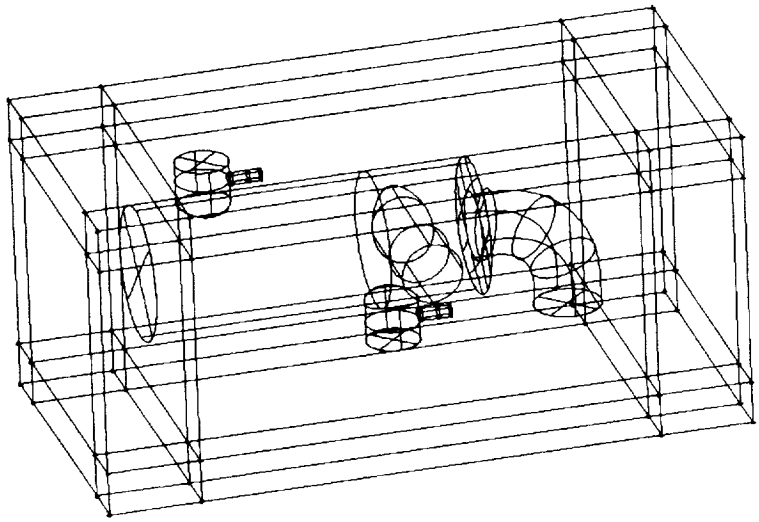

Figure 5: Block decomposition strategy

The system automatically initializes the blocking as seen in Figure 5.

Next the blocks are split interactively. During this process the blocks that will be used for grid generation are also selected. After splitting of blocks, the block edges and vertices are fit to the geometry with interactive manipulation. The edges of the blocks can be associated to the curves taken from the CAD model. After the association is done, vertices are moved onto the curves. The blocking structure is shown in Figure 6. After the initial fit, blocks are split to provide control over the critical areas such as the control ports.

After the initial fit, blocks are split further to provide control over the areas such as the pressure ports. The system starts to build internal index control for the blocks as the split operation takes place. Block indices can be used to control which part of the blocking are visible. Additionally, only those parts of the blocking which are currently visible are split during the splitting operations.

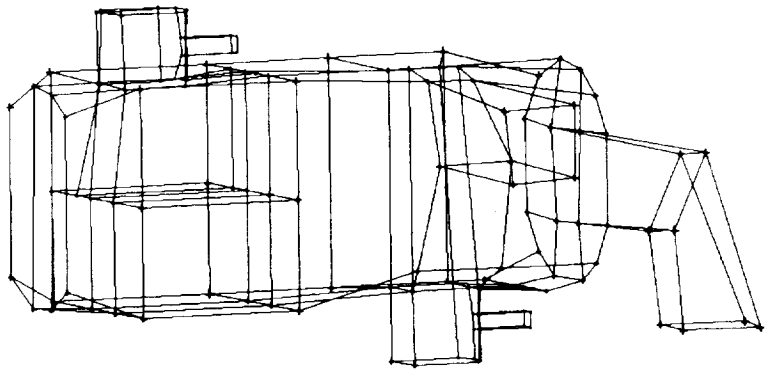

Figure 6: Block edges of the inner block is being fit to the curves.

Figure 7 illustrates the blocking structure created. This is only one of many possible blocking strategies. The user may select a strategy that is most appropriate for the type of analysis to be performed. Also seen in Figure 7 is the block edges are fit to the CAD surface geometry.

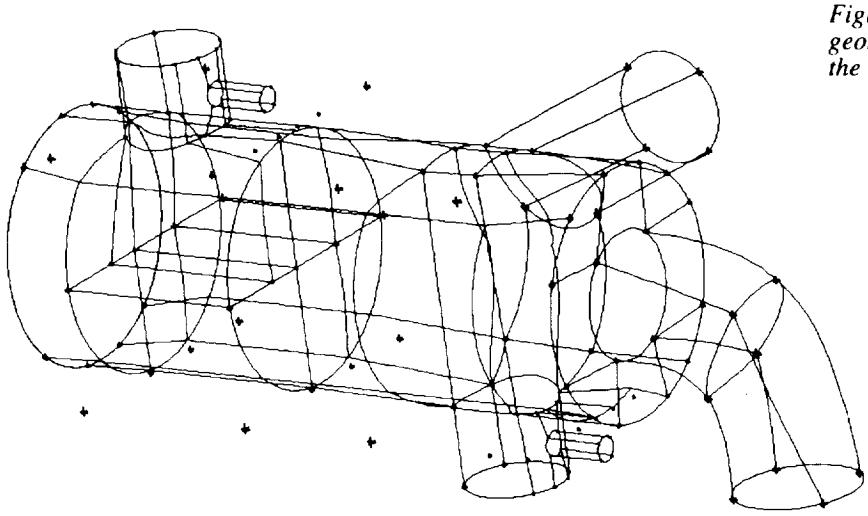

Figure 7: Blocking of the entire geometry. The block edges are fit to the surface geometry automatically.

The next step is to define the sizes of the grids on the surfaces of the geometry. The maximum length, initial height and the height ratio off the surface are defined on the object surfaces. If it is necessary, the sizes can later be adjusted on the edges individually.

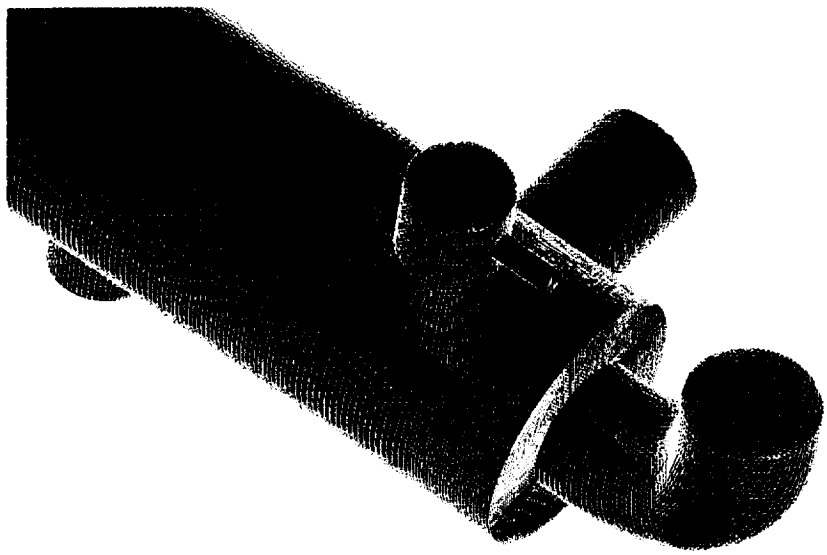

Figure 8: O-Grid generation around the sail and the rudders

Figure 8 shows the initial calculated grid. This grid contains many skewed cells since every cylinder is fitted with a single block. To improve the grid quality O-grid generation is needed. Built in tools allow the automatic O-Grid creation. First the blocks are selected and then the faces of these blocks that the O-Grid should pass through. Using the scale factor parameter the distance between the internal block to the walls of the external block is specified.

We will illustrate this feature on one of the control ports. Figure 9 shows the resultant blocking structure after the O-grid is created. The interface of the blocks between the pressure cylinder and the control cylinder are generated automatically by the system..

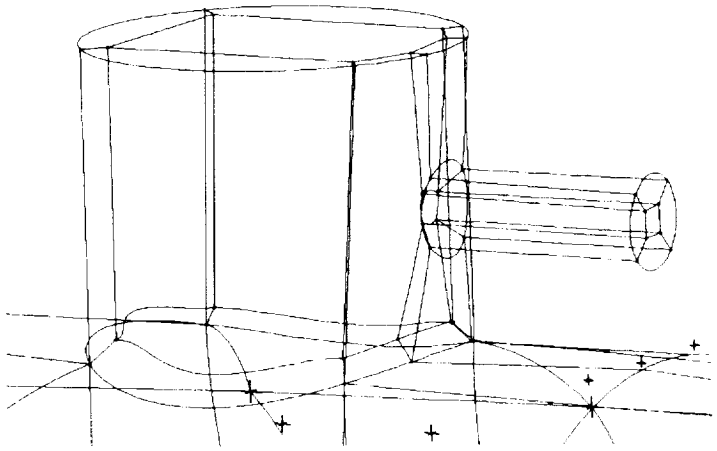

Figure 9: Surface and the cross sectional volume grid.

Figure 10 shows the computational grid around the control port with O-Grids. Using this approach the resultant computational grid skewness is 60% and above. The skewness on the perfect rectangular brick is measured to be 100%.

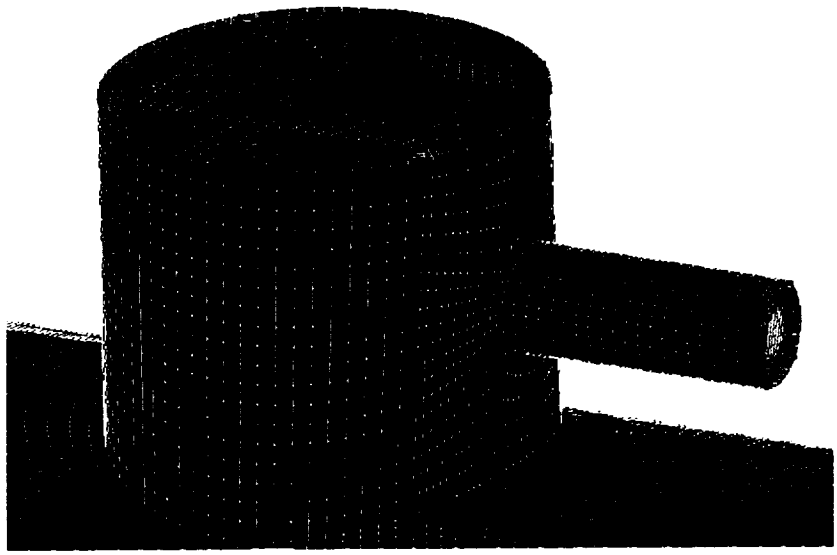

Figure 10: Automatically created O-Grid around the control port.

THE FULLY AUTOMATIC OBJECT BASED GRID GENERATION USING ICEPAK

ICEPAK is a CFD application for thermal management of electronic enclosures. This very easy to use object based package combines model (geometry) generation, automatic grid generation, flow solution with coupled thermal-flow simulation, and post processing into a single environment. It will help the designer to reduce enclosure sizes, eliminate the hot spots, meet noise considerations, increase component density into smaller areas and optimize the locations of fans and vents.

Once the model has been defined using objects, the computational grid is generated automatically. The user either defines the sizes of the elements for each objects, or selects to have ICEPAK calculate the grid automatically based on the objects. A multi-block hexahedral grid is calculated and is body fitted with o-grids around most objects. ICEPAK allows generation of grids at varying levels of complexities. The rules governing each object results in a prioritized O-grid methodology whereby each object is meshed individually as tightly as the user specifications permit in order to resolve the physics of the solution. Objects like blocks, cylinders, thin inclined walls, and wedges can be considered. The following example illustrates the automatic grid generation capability:

As shown in Figure 11, the cabinet is housing a vent, an opening, a box representing a disk drive, 2 fans, one stack of vertical PCB's (4 in a stack), and a power unit. The generation of the geometry is done by selecting objects and bringing them into the cabinet and placing them using the mouse or entering the coordinates from the key board. Using the mouse one can also resize the objects interactively. As the geometry is being build, ICEPAK compiles the boundary conditions on each object. ICEPAK also monitors the

information on each object for grid generation. Each Icepak object has a set of rules or parameters associated with it which are used to guide the generation of the grid around the object. User can selectively modify any individual parameter for any object and then regenerate the computational grid automatically. Any parameter specifically modified and toggled on will be enforced during the grid generation process. This procedure is used to

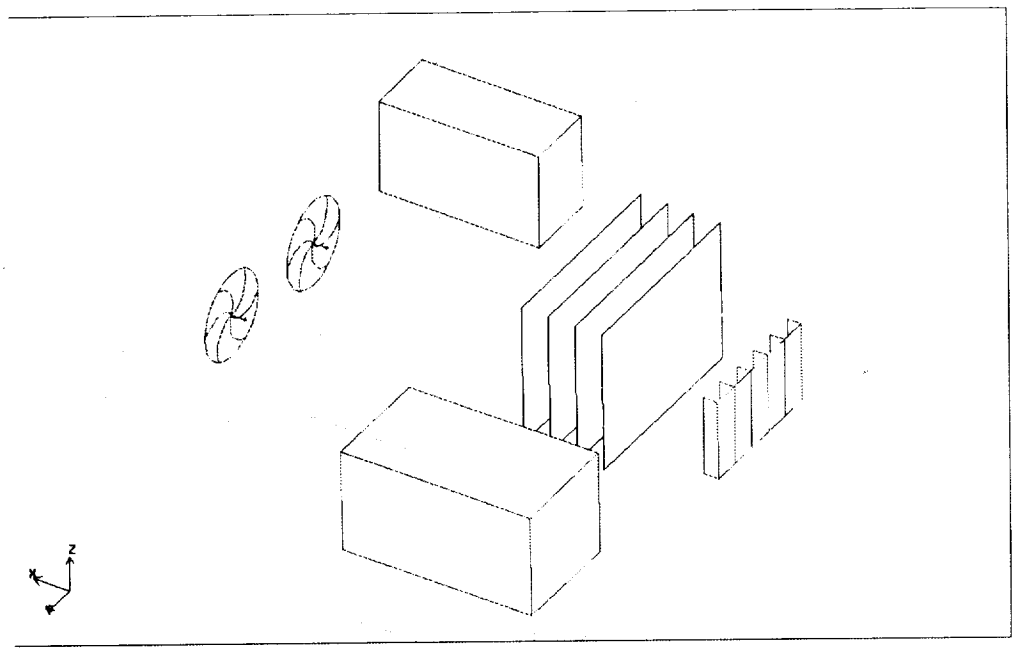

Figure 11: Cabinet housing PCB's, a disk drive, a power unit, 2 fans, vent and a opening.

selectively refine the grid around a particular object.

Shown in Figure 12 and Figure 13 are the result of the automatic grid generation process. The plane goes through the middle of the cabinet where the griding of the disk drive and the fans result in O-type grids. General grid generation guideline is to use the minimum count option in conjunction with the maximum x, y, z size option and possibly the maximum initial height option. The maximum x, y, z size option limits the maximum length of any grid element in the corresponding x, y or z coordinate direction.The maximum unit height specifies the maximum height of the first element layer generated around any object (PCB, block, fan, etc.) in the grid. Surface grids on individual objects can be displayed. Thus, for example, you could study the grid on the surface of all blocks in the model or restrict the view to a single specific block. It is also possible to view the surface grid on all objects simultaneously or display the entire mesh

Fully automated grid generation is made simple and fast with ICEPAK. User can generate very complicated grids in minutes rather then weeks. Since the knowledge and the decision making of grid generation process is build into the software, the user can concentrate on other aspects of the analysis such as if objects have been put in the correct place, the solution costs and post processing of computed results.

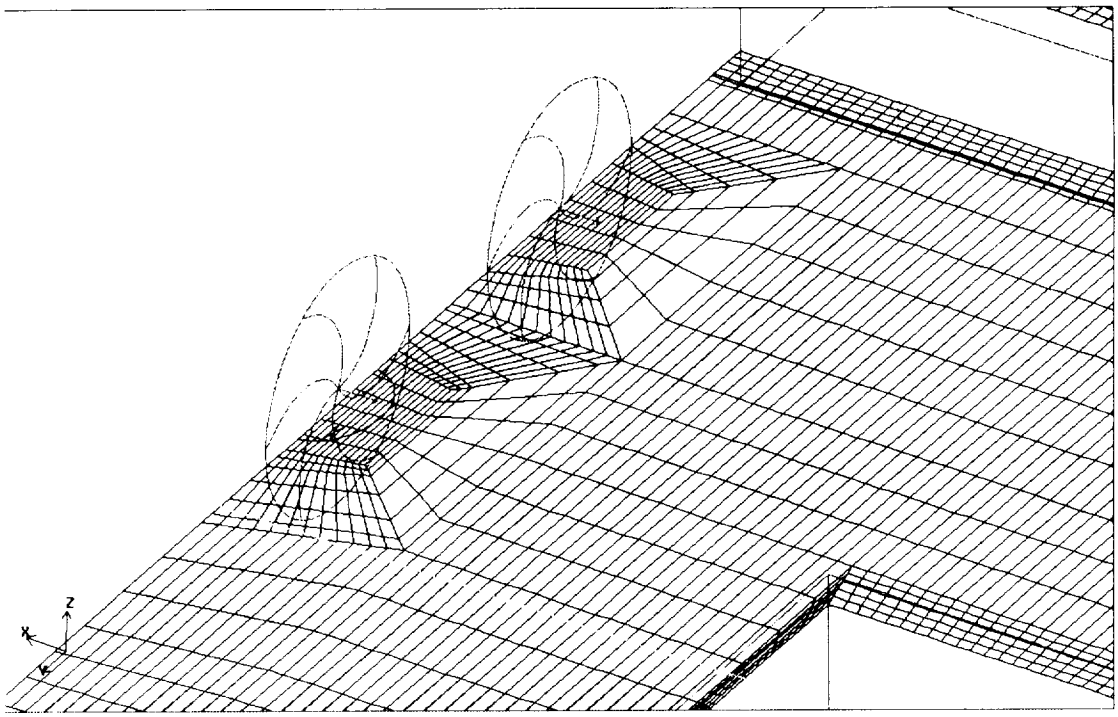

Figure 12: Automatically generated O-Grid around the fans

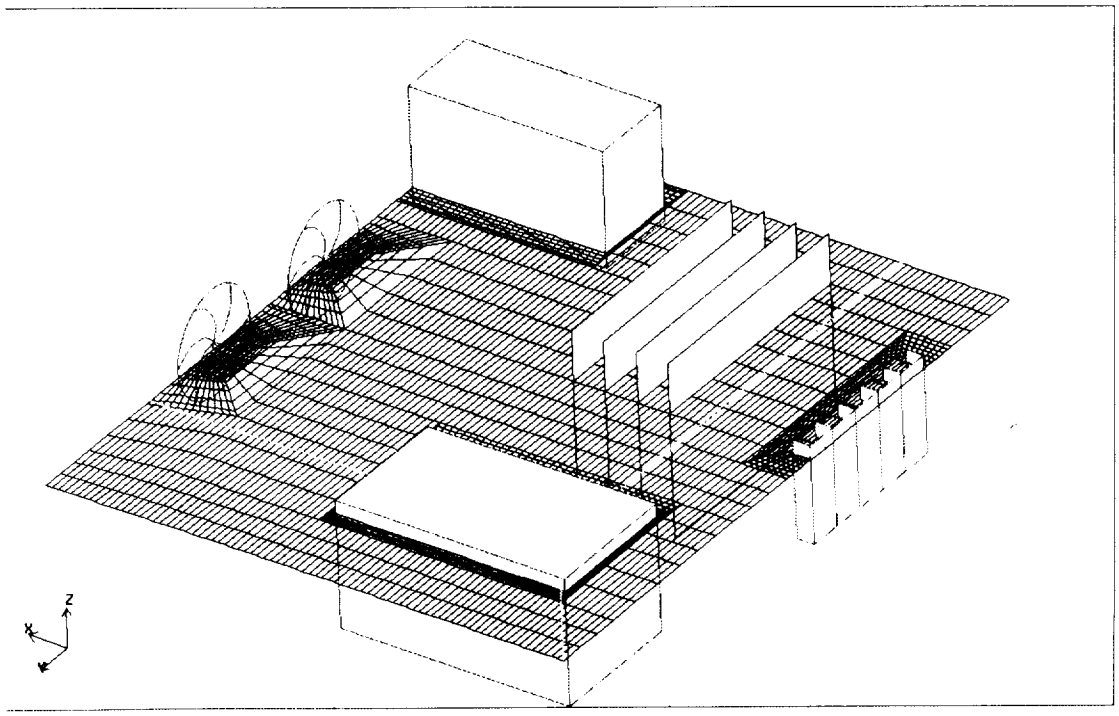

Figure 13: Fully automatic grid generation inside the cabinet housing

CONCLUSIONS

Given the current trend towards more accurate and complete representations of complex flowfield, it is important to have enhanced tools suitable for geometry modeling and grid generation. The current interactive approach for geometry modeling and grid generation allows the direct decision making needed to handle the wide variety of geometries possible. But the expense of the time of the application engineer is needed to be reduced further by providing smart tools for grid generation. The added tools to the current ICEM CFD, such as ICEM COMAK for parametric grid generation and ICEM HEXA for rapid grid generation; will shorten the time for computational grid generation significantly.

The object oriented grid generation tool as implemented in ICEPAK is definitely the new trend in computational grid generation. It is necessary to extend these capabilities to cover a wide variety of shapes.

REFERENCES

1. Integrated Geometry and Grid Generation System for Complex Configurations, Surface Modeling and Grid Generation Symposium, NASA Langley Virgina, 1992

2. ICEPAK Reference Manual, June 1994, Revision 1.0, 1st Edition. Fluid Dynamics International, Inc.

3. Tcl and the Tk Toolkit, John K. Ousterhout, Addoson-Wesley Publishing Company, July 1994

4. A new Automatic Grid Generation Environment for CFD Applications, D. Bertin, C. Castles, J. Lordon, AIAA CFD Applications Meeting, June 1994, Stanford, CA U.S.A.

SURFACE GRID/GEOMETRIC GRID GENERATION