Final Report on the Evaluation of the Dornier Gmbh Interactive Grid Generation System

Robert L. Brown

February 1989

Research Institute for Advanced Computer Science
NASA Ames Research Center

RIACS Memorandum 89.1

NASA Cooperative Agreement Number NCC 2-387

Final Report on the Evaluation of the Dornier Gmbh Interactive Grid Generation System

Robert L. Brown
Research Institute for Advanced Computer Science
Mail Stop 230-5
NASA Ames Research Center
Moffett Field, CA 94035-4000
415 694 6363

RIACS Memorandum 89.1

ABSTRACT

This report summarizes the work performed under a supplement to NASA Cooperative Agreement NCC 2-387, between June 1, 1988 and February 13, 1989, to investigate and evaluate an interactive grid generation program, INGRID, developed by Dornier Gmbh. The report includes a description of the task and work performed, a description and evaluation of INGRID, and a discussion of the possibilities for bringing INGRID into the NASA and Numerical Aerodynamic Simulator (NAS) computing environments. In summary, we found INGRID to be a viable approach for grid generation and determined that it could be converted to work in the NAS environment, but that INGRID does not solve the fundamentally hard problems associated with grid generation, specifically, domain decomposition.

Principal Investigator:	Dr. Robert L. Brown
Technical Monitors:	John Barton Jack Nielson Marcelline Smith
Report Date:	February 7, 1989

The limitations are as small as the fantasies of the programmer.

-- Werner Seibert

1. Introduction

This report concerns a brief study of an interactive grid generation program written by Werner Seibert of Dornier Gmbh, Friedrichshafen, West Germany. This work was performed under a supplement to NASA Cooperative Agreement NCC 2-387 to the Universities Space Research Association. The Principal Investigator for the task was Robert L. Brown of RIACS and the principal Technical Monitor was John Barton of Code RND, NASA Ames Research Center. The findings, conclusions, and recommendations resulting from this study are described herein.

2. Task Description

This section provides background for the study; the remainder contains the text of the original agreement between NASA and USRA, dated May 24, 1988. Essentially, the agreement outlines a larger project, of which only an initial step is funded. There are three items of work funded in this task:

1) Acquire the INGRID software.

2) Produce an evaluation of the software, including its style and portability.

3) Estimate the effort required to transport the software into the NAS environment, making recommendations on the approach to take.

2.1. Supplemental Agreement

The generation of grids for Computational Fluid Dynamics (CFD) calculations is a difficult process, historically requiring substantial manpower. As a result, grid generation is one of the pacing items in the practical application of CFD to aircraft design. Dornier Gmbh has conducted research on interactive grid generation and has developed a computer program (INGRID) able to use the graphical capabilities of workstations to accelerate the process of grid generation. This program can potentially decrease the manpower required for grid generation by an order of magnitude.

RIACS and Dornier have executed a Memorandum of Understanding that provides RIACS with access to the existing INGRID software and the assistance of expert help from Dornier personnel familiar with INGRID. The end objective of the proposed RIACS research work is to develop an improved interactive grid generator for use by NASA personnel in the environment of the NSPN (NAS System Processing Network). Technical liaison with Dr. John Barton of the NAS Systems Division will be maintained to assure that the RIACS research and development efforts correctly reflect the needs of NASA and the environment of the NPSN. Though the fund-

ed efforts will be incremental, we agree that ultimately, the following specific activities may be performed:

1) RIACS will acquire the INGRID software from Dornier and evaluate the feasibility of transporting it into the NPSN UNIX environment. This effort requires at a minimum the generation of a report describing the software, its language of implementation, structure, and style, and estimates of the effort required to transport the software to the NPSN system. Included in the port will be recommendations concerning the approach to be taken in transporting the software: the language to use, the graphics interfaces to use, and the overall system structure.

2) Contingent on supplemental funding, RIACS will initiate an effort to transport the INGRID software to the NSPN environment, specifically the IRIS graphics workstations and possibly including, depending on the results of the initial analysis, the NAS high-speed processors. The transporting work will be headed by RIACS staff with assistance as necessary from NAS personnel to provide technical information about the NPSN environment. RIACS will provide the software in working order to the technical liaison along with all working documentation concerning the implementation.

3) RIACS will make the software available to the CFD community within NASA and serve as a clearing house for recommendations concerning its extension and enhancement. RIACS will provide service in implementing specific enhancements as agreed upon with the technical liaison. The service can range from suggesting how to implement the enhancements to directly performing the implementation.

4) RIACS will perform the necessary research and development to implement an improved interactive grid generator designed to operate new generation superworkstations. The software will be based on a standard graphics language (e.g., PHIGS+) to facilitate portability to other modern workstations.

SCOPE OF THIS INCREMENT: This supplement contains a budget for implementing step (1) described above. If the results of that analysis indicate that subsequent steps are feasible, NAS will provide additional funding according to new budgets supplied by RIACS, subject to negotiation.

3. Work Performed

3.1. Preliminary Information Gathering

During July and August, 1988, we spent time gathering information about INGRID and explored the extent of possible collaborations. At this time, Boeing Cor-

poration also became interested in INGRID, and we discussed the possibility of three-way collaboration, using the "industrial affiliates" mechanism offered by RIACS. However, Boeing decided to not enter a formal agreement with RIACS or Dornier because doing so may prevent them from developing similar technology in-house.

During these early weeks, we learned several facts about INGRID and the environment in which it runs. We were initially concerned that, because it was developed in Europe, that it may use European hardware and software technology unavailable on-shore. However, we learned that INGRID is programmed in Fortran, runs on an IBM 3090 with IBM OS/MVS, and uses a SpectraGraphics 1500 as its front-end. We also learned that the graphics interface is a SpectraGraphics-proprietary library named PRISM which, according to SpectraGraphics, is GKS-like.

3.2. Initial INGRID Evaluation

In late August, 1988, after telephone conversations with Dr. Bernard Wagner of Dornier, Dornier shipped up a magnetic tape containing the source code to INGRID Version 2.3 and related documentation. The tape was created on an IBM 3090 system and we required assistance from Ames Code RC to read it and to move the source files to a UNIX machine inside RIACS. By the first week of September, we had the entire source code for INGRID 2.3 on a UNIX machine and began examining it.

INGRID 2.3 source code consists of 46 separate Fortran modules and one BAL module. The BAL module converts EBCDIC strings into floating-point representation and handles simple arithmetic expressions (however, INGRID 2.3 does not use the expression evaluation facility). This is nearly the same function performed by the UNIX C library routine *atof()*. The Fortran source code comprises about 4500 lines and 700 lines of comments.

The Fortran code is well written and it successfully compiled on a Sun Microsystems workstation (Sun OS version 3.2) with only a few changes, mostly dealing with static initialization of dummy parameters inside subroutines and with mixing static initialization with data type definitions. However, linking resulted in 64 unresolved external references, all but two of which are from the PRISM graphics library. Of the remaining two, one was from the BAL module and the other was ARCCOS, which is ACOS in the Sun Microsystems library. The significance of the successful compilation of the Fortran code is that the major effort in porting INGRID 2.3 to a different graphics device is potentially limited to converting the calls to the graphics library, in this case PRISM, to the new library.

The calls to the PRISM graphics routines are scattered throughout the Fortran code in INGRID 2.3 (there are over 450 individual calls in 27 source modules to the 62 routines). This presents a major problem when considering a port. Additionally, we

did not have a PRISM manual from which to draw conclusions about the feasibility of converting graphics calls into those of another library. Also, at about this time, we learned that Mr. Seibert had made extensive revisions to INGRID and that our code was obsolete. Hence, in order to gather more information about INGRID efficacy and implementation, we planned a trip to Dornier.

3.3. Dornier Site Visit

Between October 17 and 21, 1988, I visited Dornier Gmbh in Friedrichshafen, West Germany, to meet with Dr. Bernard Wagner, head of the Aerodynamics Computation group, and Mr. Werner Seibert, author of INGRID. En route, I rendezvoused with two Boeing employees, Dr. Chester Koper, Manager of CFD Computing (Boeing Computing Services), and Dr. Tsong-Jhy Kao, Senior Engineer in the CFD Lab (Boeing Commercial Airplanes). Once at Dornier, the five of us held several meetings to discuss the Memorandum of Agreement between RIACS and Dornier and potential Boeing involvement.

Dornier is very interested in establishing collaboration with outside groups interested in INGRID, but as yet does not have a formal mechanism in place for routinely handling new collaborations. Two options they were considering were to continue with ad hoc collaborations with agreement of two-way technology transfer, or to develop a formal mechanism of licensing INGRID software and interactive grid generation technology.

Dr. Wagner reported that they had established similar agreements concerning INGRID with other groups, one of which is Daimler-Benz, the major stockholder of Dornier. Daimler-Benz has initiated a project to transport the INGRID code to use PHIGS on an Evans & Sutherland PS-390.

Most of the remaining time at Dornier was spent learning about and using INGRID. I was particularly interested in the PRISM library, its semantics and language bindings, since such information is not readily available by merely reading the INGRID code.

Mr. Seibert had modified INGRID considerably since version 2.3 was frozen; the new working version number was 3.1. One of the changes was a restructuring of the source code into three layers: the main driving program, the functional routines, and a virtual graphics layer. The main program provides the main sequencing logic for INGRID. The functional routines implement the algorithms and data structures used for the grid generation portion. The virtual graphics layer provides an abstract graphics binding between the functional layer and the graphics library.

Dornier has a single stand-alone Sun Microsystems workstation and so I was able to perform some evaluations of INGRID 3.1 while in Germany. Version 3.1 com-

prises 75 Fortran source modules and the same lone BAL module. Twenty-one of the Fortran modules implement the virtual graphics layer and so all PRISM calls were in those modules. The program had grown to almost 7000 lines of code with 1000 lines of comments.

The virtual graphics layer provides a mapping from the virtual graphics interface designed by Mr. Seibert into the SpectraGraphics PRISM library. However, the interface is only documented in the 21 modules that form the layer and so an understanding of the PRISM library interface was still necessary. To aid that understanding, Dornier loaned me a PRISM library manual to bring back to the US. Additionally, after acquiring the proper authorizations, Mr. Seibert released the source code for INGRID 3.1 to me.

3.4. Werner Seibert Visits Ames

As a follow-up to the Boeing and RIACS personnel visiting Dornier in October, Werner Seibert visited Boeing and NASA Ames Research Center in early December. As a part of the visit, Mr. Seibert gave a RIACS-sponsored seminar on INGRID in which he described its capabilities and showed a short movie that demonstrated its user interface. The talk was well attended and was followed by an hour of discussion. In the days after the talk, Werner & I worked on transporting INGRID to the Ardent Titan-2 (uncompleted) as a method of evaluating the virtualization of the graphics interface.

While at Ames, Werner Seibert met individually with people from RIACS and Ames Codes RTA, RND, and RFA. There remains within Code RTA considerable interest in INGRID.

4. Evaluation of INGRID

This section presents an overview of INGRID, its algorithms, user interface, and implementation.

4.1. Description of INGRID

INGRID is an interactive computer program usable for the generation of block-structured internal grids. A typical end-to-end CFD system comprises the following steps:

1) **Geometry definition.** For fluid flow simulations, geometry definition entails describing the form of the objects in space around and through which the simulated fluid flows. This step is typically represented as inter-related surface segments which are in turn represented as spline patches or polygons.

2) **Domain decomposition.** For block-structured systems of grids, the next step is to segment the simulation space into discrete domains, each of which will have its own simulation grid. The blocks are virtual cubes the union of which fills the simulation space. The far field boundary is defined in this step.

3) **Grid generation.** Once the simulation domain has been decomposed into blocks, a grid must be generated for each block.

4) **Fluid flow simulation.**

5) **Visualization of results.**

INGRID is solely concerned with step 3, grid generation. The input to INGRID describes the boundaries of the grid blocks and the union of these blocks covers the space. The faces of the blocks may be either portions of the surface geometry or boundaries between grid blocks; INGRID does not distinguish between the two. Hence, INGRID does not address the difficult problem of domain decomposition. At Dornier, the CAD programs CADAM and CATIA are used for steps one and two above.

A user of INGRID follows a sequence of events to generate the grids for the blocks. From a high-level view, the sequence is as follows:

```
Invoke INGRID
Read in the geometry data and previously generated grids
For each block:
    Generate internal grid
Save results
```

The input geometry specification is an array of named line segments, each line segment being defined as an array of points, often called a "polyline." These lines define both the surface geometry and the block boundaries. The line segments must define the edges of all the blocks to be filled with grids. However, there may be additional line segments that define surface lines on the faces of the blocks. Such information is necessary when the edges of a block face do not sufficiently describe the face. The geometry data can come from a variety of sources -- from commercial CAD packages to hand generated. The specific grammar for the input data to INGRID is documented in Seibert's report [Seib88]. This low-level form of input was chosen, as opposed to more complicated representations such as spline patches, because it is the lowest common denominator of the representations of all the sources of surface geometries with which Dornier has had to work.

There are three major steps in generating internal volume grids with INGRID. First, generate the end points for surface grids by specifying the number of grid points

and their distribution along the edges of each face. Next, generate the surface grids by a multistep interpolation between the corresponding end points. Finally, generate the volume grids from the surface grids.

The user of INGRID follows distinct steps in the process of generating the grids. We now present a more detailed description of the steps.

1) Read the input segments that form the block edges. As described, some of these edges are a part of the original surface geometry.

2) Designate the edges of a block to work on. Because the input data does not associate line segments into blocks, this step is necessary. The procedure is to identify the edges that form the minimum and maximum for the three dimensions i, j, and k.

3) Designate additional surface topology lines. As previously described, these lines give INGRID additional information about the shape of the faces of the block. In designating each surface topology line, the user also says which plane (i_{min}, i_{max}, j_{min}, etc.) the line lies on and which sweep direction (1, 2) indicating in which direction the interpolation is done first (more on this later). At this point, INGRID removes everything from the screen except the line segments that have been designated as constituting the current block.

4) Select the number of surface grid line end points and the point distribution for each dimension. The point distribution is selected from a menu offering uniform, geometric, specify first delta, specify last delta, specify both deltas, user-defined (coded in or read in) distribution, or inherited from some other edge. When generating a grid in a block neighboring another block that has already been completed, typically the user will tell INGRID to use the same number of points and distribution as on the corresponding edge on the neighboring block. During this stage, the program guides the user by highlighting the next edge to work on.

5) Generate the surface mesh on each block face. This is an automatic procedure, invoked after the user is satisfied with each edge point count and distribution. INGRID interpolates the edge point distributions onto the surface lines, if any, and completes the surface grid by using a two-pass interpolation scheme.

6) Generate the volume grid. The user specifies the initial direction of interpolation and INGRID generates integer planes containing interpolations from the minimum face to the maximum face. The specific algorithm is documented in Seibert's report. Once the integer planes have been generated, the user can view them individually by rotating a knob on the dial box. Option-

ally, the user can perform a volume check which will red-line any cells with negative volume.

7) Optionally modify the surface grids and generate new volume grids. A combination of Poisson smoothing and reinterpolation is available for this step.

8) Optionally manually modify individual grid points. This is an infrequently used operation. Modification to internal grid points will be undone if Poisson smoothing or resplining is performed, as these regenerate the internal grids from the surface grids.

9) Save the grid. Again, INGRID uses polylines as the representation. Because of the computing environment that INGRID runs in, each block grid is stored in a separate data file.

10) Repeat steps 2 through 9 until all blocks have been filled with grids.

4.2. Native Computing Environment

This section describes the computing environment in which INGRID runs, and how it relates to the NPSN environment.

INGRID is almost entirely written in Fortran 77 with IBM extensions. At Dornier, it is compiled and run on an IBM 3090 (without vector unit) as a batch program. Hence, to run it at Dornier, they initiate a batch job on the 3090 which, when started, begins communicating with the user through the graphics terminal. INGRID is an expensive program to run, because of the memory residency charges and the CPU charges. Typical INGRID sessions take dozens of wall-clock minutes.

The graphics interface unit is a SpectraGraphics 1500 connected through a channel adapter to the 3090. The 1500 is a microprogrammable vector graphics machine capable of real-time transformations on substantial data sets without intervention from the controlling host. For the purposes of INGRID, the user first downloads (as a part of the batch job) a PRISM interpreter into the 1500; INGRID uses the PRISM graphics library (SpectraGraphics proprietary) which translates the procedure calls into messages to the graphics system.

The user interface to INGRID uses a graphics tablet and puck for picking points and lines. There is no specific dependence on the tablet, however, and a mouse could be used for the same job. INGRID uses a knob box (8 dials) to control the rotations, scaling, and integer plane viewing.

The computational requirements for INGRID appear to be fairly low; at no point in the generation of example grids was there a lengthy delay. In part, the algorithms used by INGRID were chosen because of their low computational demand. More sophisticated algorithms will require more computational power. The IBM 3090s

range in power from 3.9 to 7.4 MFLOPS on the Dongarra rating [Dong89] for straight compilation, non-vector mode (the difference is in different models of 3090s; I do not know which Dornier has). For reference, an Ardent Titan-2 is rated at 9.4.

4.3. Transporting to NAS Environment

Any consideration concerning transporting a piece of software from one environment to another must examine multiple aspects, including language, operating system, libraries, character sets, special hardware requirements. These are handled separately here.

1) **Language.** INGRID is written in Fortran 77 and makes little use of extensions. I have successfully compiled it on a Sun Microsystems Sun-3 workstation under Sun OS 4.0.1, on an Ardent Titan-2 under Unix System V Ardent 2.0 beta, and on a Sequent Balance 21000 running Dynix 3.0.14 (INGRID 2.3 only). In each case, minor changes were required in the code to accommodate compiler differences. Hence, compiling is not a barrier to transporting INGRID.

2) **Operating System.** Operating system differences impact a transporting effort mainly because of differences in how the OS provides I/O services to the programs. This is particularly true of programs written in standard Fortran. INGRID assumes that, by virtue of the JCL (Job Control Language) used to invoke it, all the data files it needs are already open and accessible through file descriptors. In a UNIX environment, this is less convenient than having the program deal with file names and then explicitly open them as needed. For example, INGRID expects the data files containing other completed block grids (which it uses when the user opts to import a point distribution) to be open and available within a range of file descriptors. In a UNIX version, the completed blocks would be stored in named files and the user would give the name of the file to INGRID as needed. In all, this is a minor change. Fortran I/O primitives do a good job of hiding OS-specific I/O services.

3) **Libraries.** Runtime libraries can present a major obstacle in transporting software, particularly when the libraries define graphics interfaces. There are two aspects to graphics libraries: functionality and language binding. The functionality presented by one library may be present in another, but differences in the language bindings can impede transporting efforts. The solution is to adopt standards for both function and binding. However, where computer graphics are concerned, there are too many standards and none of them are universally adopted.

   INGRID uses a proprietary graphics library, PRISM, for all of its output (graphics) and input (picking, dial box). As previously described, how-

ever, INGRID has an abstract graphics interface and a layer that maps that into the PRISM interface. However, that interface is designed with the PRISM model of graphics in mind, and many of the routines in the graphics layer comprise only one or two lines of code. The functionality of PRISM is similar to that presented by most modern graphics libraries: a multiple graphics segment-based display list allowing for hierarchical segments. PRISM allows transformations on graphics objects to be associated with dials on the dial box without intervention from the host program. This feature may need to be simulated in a new environment.

4) **Character Sets.** INGRID makes very little use of character data. Most places where it is used, Fortran hides the representation sufficiently well. At Dornier, INGRID uses the EBCDIC character set. At ARC, it would use ASCII. The only routine that needs to understand the difference is coded in BAL and would need to be replaced anyway.

5) **Special Hardware Requirements.** Other than for specialized graphics equipment, INGRID does not have any special hardware requirements. For graphics, all the output from INGRID is in the form of line segments and text; no raster fill or shading is required. Input operations consist of the ability to select a graphical object on the screen (as with a mouse) and the ability to rotate and scale the image on the screen. For the latter function, INGRID uses a dial box. Mr. Seibert and I discussed the possibility of replacing the dial box with screen-based objects, such as a virtual knob box, and concluded that, though it would be feasible, that it would dramatically degrade the quality of the interface. When picking an object on the screen, the user can simultaneously reorient the image so that the object is visible and easily accessible (i.e., not too close to some other object that might be picked instead) and move the pointer toward the object. The ability to perform these simultaneously would be lost without a knob box.

5. Conclusions and Recommendations

The conclusion of this study is that INGRID is a useful program for generating grids and that it would be relatively easy to transport it into the NPSN environment. The usefulness conclusion is partially based on the level of interest generated during and after Mr. Seibert's talk at Ames; several people expressed an interest in being able to try INGRID and a couple were anxious to generate a grid right away.

5.1. INGRID Limitations

Though INGRID would be a useful tool at Ames for generating relatively simple grids, it does have limitations, both in functionality and implementation, outlined below.

1) **Lightweight grid generation algorithms (functionality).** The grid generation techniques in INGRID are not as sophisticated as those being used regularly in ARC Code RFA, for example. Mr. Seibert claims that the code for generating the grids is relatively isolated from other code inside the program. However, the main controlling logic of the program (identify a block, apply point distribution law, generate surface grid, generate volume grid) establishes a grid generation paradigm that may not be applicable to other techniques. Additionally, INGRID's algorithms are fast and efficient. More sophisticated algorithms may require more computing power than can be made available in an interactive program.

2) **Simplified geometry description (functionality).** A specific problem with INGRID concerns its use of polylines as input representation. Because the true surface topology is not fully described by edge lines and a small number of surface lines, the surface grids that INGRID generates cannot be made to match the block surfaces for arbitrary blocks. This problem arises even for relatively simple block shapes. The task of sufficiently describing the surface topology is left to the domain decomposition task.

3) **Grid generation only (functionality).** INGRID only allows the generation of internal grids within blocks. There there is no inherent limitation on the number of blocks that INGRID can handle, the layout and interrelationships of the blocks is not addressed and mostly unknown in INGRID. For example, INGRID has been used at Dornier to generate and entire simulation grid around the Do328 aircraft. This grid comprises 71 blocks, 22 or which are in contact with the aircraft body. The task of decomposing the simulation space was performed manually with the assistance of only a drawing (CAD) tool. The decomposition task overwhelmed the grid generation task in time and complexity.

4) **No stored interblock relationships (functionality).** Even though INGRID allows the point distribution for a block edge to be imported from another, it does not preserve that relationship in its internal or external data structures. Hence, if the point distribution on the inherited-from edge is changed by the user, INGRID does not automatically change the corresponding one on the inherited-to edge. Likewise, if the domain decomposition is changed, the user is probably faced with starting over in grid generation. A better solution to grid generation would combine decomposition and grid generation and allow information to flow from one to the other.

5) **PRISM (implementation).** The virtual graphics interface in INGRID was inserted after INGRID was originally written specifically for the Spectra-Graphics 1500 and the PRISM library. Hence, it is not sufficiently abstract to describe a wide variety of graphics libraries and machines. For example,

menus are used heavily but not abstracted; instead they are explicitly drawn using the line and text drawing primitives of the library.

6) **Dial box (implementation).** The use of a dial box for rotations, translations, and stepping through integer planes is a fundamental part of the INGRID interface. Ames does not routinely acquire dial boxes for its graphics workstations.

7) **Ongoing support (implementation).** Any software package that is used and extended by a geographically dispersed community suffers inherently from a lack of global revision control. INGRID is such a package. No mechanism has been established allowing Ames to send enhancements back to Dornier and Dornier has no funded mechanism for incorporating them. Mr. Seibert's job at Dornier involves research into interactive grid generation techniques, not incorporating other's enhancements into his code.

5.2. Transporting Effort

Ninety percent of the effort required to transport INGRID into the NPSN environment would concern converting the program from using the PRISM graphics library to using another library, such as PHIGS or the IRIS GL. The author started a port to and Ardent Titan-2, using the Dore library for the graphics. The conclusion is that the virtual graphics interface is too PRISM-oriented to easily be remapped to another library without redesign. Hence, the approach taken was to emulate the PRISM library, and hence the SpectraGraphics 1500, using Dore. Software data structures within the emulator were designed to mirror the PRISM data structures.

Ideally, INGRID would incorporate a more abstract graphics interface that could be retargeted for several libraries. Only somewhat less ideally, INGRID would use an industry standard, such as PHIGS, as its graphics interface. However, each would require substantial retargeting of the graphics-related portions of the program. Such a retargeting would suffer from the "ongoing support" problem described above.

By changing as little of the existing INGRID code as possible, and by layering it on top of a PRISM emulator, the ongoing support issue, at least for graphics bindings, disappears. The task of porting INGRID into a different environment thus becomes equivalent to writing a PRISM emulator for the new environment. This approach has a second benefit. The SpectraGraphics 1500 runs the PRISM emulator at Dornier. The IBM 3090 passes messages to it across a channel adapter. This same scenario could be replicated here, that is, a separation of the INGRID code from the graphics machine that runs it. Hence, by using a remote procedure call interface for the PRISM emulator, the computational parts of INGRID could run on a powerful processor, such as a Cray 2 or Y-MP, and the graphics portions could run on the less powerful (numerically) graphics workstation.

5.3. Specific Recommendations

This section proposes the next step in the Ames involvement with the Dornier interactive grid generation software. There is sufficient interest at Ames in INGRID to pursue activity. The specific recommendations are as follows:

1) **Transport INGRID into the local environment.** Specifically, this means into a UNIX environment with a locally available graphics workstation as the front-end. The recommended approach is to develop a PRISM emulator targeted to a Silicon Graphics IRIS workstation. It would be advantageous to also make a PRISM emulator targeted to another, more widely available, graphics library, such as PHIGS or Dore. PHIGS is not widely available at Ames yet, and Dore is available for the Ardent Titan, the Cray 2, and Sun Microsystems workstations. Dore is a good choice for another reason: Dornier is interested in acquiring an Ardent Titan. Providing them with a mapping to Dore for that machine would enhance our relationship with them.

2) **Distribute INGRID locally.** The more people at Ames with access to INGRID, the more feedback we will be able to provide Dornier and the more potential for ongoing collaborations. INGRID is covered, however, by a non-disclosure agreement between RIACS and Dornier, which prevents redistribution of the source code. Hence, a legal mechanism for making INGRID available at Ames needs to be devised. The issue is more severe if the source code to INGRID is wanted by Ames groups so that they can extend it to include new algorithms, or adopt it to their own computing environment.

3) **Identify a support group.** Fundamental to the ongoing usefulness of INGRID at Ames is the identification of a point-of-contact for all INGRID-related work. This person would act as an architect for enhancements and a liaison between Ames and Dornier for future collaborative work. Because of the redistribution problem, the support group may, at least initially, be responsible for providing a location (machine) where INGRID can be used and worked on.

The effort required to create a retargetable PRISM emulator initially targeted for an IRIS or Dore is probably about three man-weeks for a knowledgeable graphics programmer who has already been exposed to INGRID. For someone starting from scratch, twice the manpower may be required.

6. References

[Dong89] Dongarra, Jack J., "Performance of Various Computers Using Standard Linear Equations Software in a Fortran Environment,"

Argonne National Laboratory, Technical Memorandum No. 23. February 23, 1989.

[Seib88] Seibert, W., "Interactive Generation of Blockstructured Volume Grids for Fluid Flow Analysis," Dornier Gmbh, Report No BF 4/88 B, March, 1988.