NASA-CR-198828

Final
IN-61-CR
OCIT
~~56870~~
P- 28

DEPARTMENT OF COMPUTER SCIENCES
COLLEGE OF SCIENCES
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

# GEOMETRIC MODELING FOR COMPUTER AIDED DESIGN

By

James L. Schwing, Principal Investigator

Stephen Olariu, Co-Principal Investigator

Summary of Research (Final Report)
For the period ended June 30, 1995

Prepared for
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

Under
**Research Grant NCC1-99**
Lawrence F. Rowell, Technical Monitor
SSCD-Vehicle Analysis Branch

**Submitted by the**
**Old Dominion University Research Foundation**
**P.O. Box 6369**
**Norfolk, VA 23518**

July 1995

# Geometric Modeling for Computer Aided Design

## 1. Introduction

The primary goal of this grant has been the design and implementation of software to be used in the conceptual design of aerospace vehicles particularly focused on the elements of geometric design, graphical user interfaces, and the interaction of the multitude of software typically used in this engineering environment. This has resulted in the development of several analysis packages and design studies. These include two major software systems currently used in the conceptual level design of aerospace vehicles. These tools are SMART, the Solid Modeling Aerospace Research Tool, and EASIE, the Environment for Software Integration and Execution. Additional software tools were designed and implemented to address the needs of the engineer working in the conceptual design environment. It should be noted that all software designs and implementations have been carried out with important feedback and continual review from members of the Vehicle Analysis Branch. Further, the actual design and implementation of these software tools has been the joint efforts the grant personnel of Old Dominion University, members of the Vehicle Analysis Branch and the Automatic Computation Division, and contract programming help from Computer Science Corporation.

SMART provides conceptual designers with a rapid prototyping capability and several engineering analysis capabilities. In addition, SMART has a carefully engineered user interface that makes it easy to learn and use. Finally, a number of specialty characteristics have been built into SMART which allow it to be used efficiently as a front end geometry processor for other analysis packages.

EASIE provides a set of interactive utilities that simplify the task of building and executing computer aided design systems consisting of diverse, stand-alone, analysis codes. Resulting in a streamlining of the exchange of data between programs reducing errors and improving the efficiency. EASIE provides both a methodology and a collection of software tools to ease the task of coordinating engineering design and analysis codes.

The engineering design cycle is characterized by iteration steps that attempt to optimize a given design for specified criteria, see for example [1,2,3]. While this usually occurs for each individual stage of a design, it rarely is carried out where the relative interactions of multiple design criteria can be optimized simultaneously. The integration of software analysis tools as described above has created an environment where it is possible to begin investigation of building optimization techniques that begin to address this problem. With this in mind, a significant portion of the effort associated with this grant concentrated first on the integration process and, once integration was available, on issues associated with optimization.

Software analysis tools used in the conceptual design environment pose several unique problems relative to integration. This implies that the integration tools associated with

EASIE were not immediately applicable. Consider, for example, a modern analysis codes, such as POST. It provides an excellent analysis capability with a high degree of flexibility. Flexibility is further added to the design system by providing design engineers with tools to interface analysis codes together (the tools of EASIE for example). The price for this high degree of flexibility is two fold. Within a given program the high degree of flexibility leads to complex data structures. The user of the program is made responsible for the creation and proper formatting of the input data file. The user is also responsible for tracking the definition of the appropriate sets of input parameters. In addition, allowing users to freely define execution sequences of analysis programs adds a high degree of interdependence in the definition of data items. Consistency and integrity of the data were left to the responsibility of the user.

Another point to consider is that the design efforts of engineers in the Vehicle Analysis Branch and indeed of engineers across the country has resulted in a mountain of design studies, data and related recommendations. Recalling information used for one design study that might be applicable to another relies on access to individuals that performed the original work is an important undertaking. Given the current stage of computer technology, it was possible to create a simple interface, working with system programs, to allow the engineers to more easily track changes to programs and data.

In the optimization arena, much progress has been made recently on tools that automatically create derivatives based upon the code used in the analysis programs. Initial work was carried out to introduce this tool into the design environment at the Vehicle Analysis Branch.

Work for this grant was carried out using the computing facilities of NASA Langley in general and the Vehicle Analysis Branch in particular. As such it was important to design software to take advantage of the hardware. To this end all graphical software used the excellent tools provided by Silicon Graphics hardware. In addition, basic research into parallel algorithm development was carried out to take advantage of equipment that will provide the true computational advances for the future.

Finally, although it is cliché, clearly the most important aspect of any software package is the development of the user interface. The most powerful software in the world will not be used if the interface is lacking.

The remainder of this report will be organized as follows: section 2. will look at the SMART package, section 3 will summarize the contributions of EASIE, section 4 will consider the development of appropriate user interfaces, section 5 will cover several aspects of data management, section 6 will describe research into parallel algorithms, section 7 will consider other consulting activities and section 8 will summarize the results of this work.

## 2. SMART

### 2.1 Overview

As mentioned in the introduction, SMART provides conceptual designers with a rapid prototyping capability and several engineering analysis capabilities. In addition, SMART has a carefully engineered user interface that makes it easy to learn and use. Finally, a number of specialty characteristics have been built into SMART which allow it to be used efficiently as a front end geometry processor for other analysis packages.

SMART provides both general purpose design objects such as spheres and toruses and specialized design objects such as airfoils and fuel tanks. These objects can be combined using the general Euler combinatorial cations of union, intersection, and difference. SMART provides access to these objects through a number of natural interaction techniques. Specifically SMART uses constructive solid geometry, sweep representation, and where appropriate boundary representations. The underlying representation of each of these is a collection of Bezier surface patches. SMART is capable of a number of low level analytic computations. These include that calculation of physical properties such as weights and measures and the visualization of data surfaces once calculated, such as pressure forces, heating rates, or lift and drag coefficients. The ultimate output of the package is a geometric model that can feed the more detailed analysis packages.

Development of the SMART system was a major accomplishment of the grant. The SMART development team consisting of grant personnel and members of the Vehicle Analysis Branch received a NASA Group Achievement Award for the development of this system. Please refer to the following reports produced in conjunction with this grant for a more complete analysis of the development of solid modelling systems for aerospace design in general and the details of SMART specifically.

1 M.A. McMillan, J.J. Rehder, A.W. Wilhite, J.L. Schwing, J.L. Spangler and K.G. Mills, *Solid Modeler For Aerospace Vehicle Preliminary Design*, Proceedings of Aircraft Design Systems and Operations, St. Louis, 1987.

2. J.L. Schwing, J.J. Rehder, M.A. McMillan, and S. Schwartz, *Providing a Common Geometry for Multidisciplinary Analysis*, Proceedings of the 3rd Air Force, NASA Symposium on Recent Advances in Multidisciplinary Analysis and Optimization, Sept. 1990.

A portion of the work was completed in the form of two master's projects.

3. D. McMillan, *Solid Modelling Design Systems*, ODU Master's Project, Department of Computer Science, 1986.

4. G. Wright, *User Interface Design for CAD*, ODU Master's Project, Department of Computer Science, 1987.

In addition to the principal investigators and the master's students mentioned above work on this section of the grant was carried out by R. Lin, a doctoral student and J. Randall, at the time an instructor at Old Dominion. The sections that follow will briefly look at several of the unique aspects of the SMART Design System.

## 2.2 Real-Time Physical Properties Calculation

Calculation of physical properties is an important step in the analysis of aerospace vehicles. Conceptual design requires the evaluation of a multitude of vehicle concepts. To achieve high productivity, a designer must be able to generate complete and accurate three dimensional models of complex vehicle shapes easily, quickly and in a natural way. Completeness of the design process requires, among other things, calculation of physical properties to be carried out as efficiently as possible with in required accuracy constraints. Ease of definition is aided by the use of hierarchical data structures (trees). Trees provide a natural technique for defining and manipulating a configuration.

Work on this grant lead to the development and implementation of algorithms which improved the efficiency of the calculation of physical properties by a factor of four. However, even with this improvement, calculations for a complex vehicle, such as a concept for a combination booster and orbiter, would require several minutes to complete. Such times are acceptable (and unavoidable given the complexity of vehicle configuration) for a one-time calculation of a given concept.

It should be noted however that the conceptual design process requires the frequent redefinition of major parameters leading to the need for continuing recomputation of physical properties. This raises questions concerning the time required to carry out that process by completely reintegrating the entire vehicle. The sections that follow present conditions under which the amount of computation involved in a revised configuration is trivial. Algorithms are then presented which apply this theory to the visualization of properties such as the centroid. Current implementation has provided the SMART designers with a tool that graphically represents the real-time, dynamic changes in the position of the centroid as various sub-assemblies are moved relative to one another. Questions such as how the centroid of a vehicle changes as a function of the position of the wings or fuel tanks are answered graphically in real- time.

The basic data structure is hierarchical in nature and can be represented by a tree. For initial conceptual design efforts, mass properties are based upon the assumption of uniformly dense objects. For example, a solid is represented by a uniform density times its volume obtained via integration over the analytic surface. Notice that certain objects may be thought of as the combination of more than one type of density. A fuel tank may be represented by both a solid, density per unit volume, for the mass of the fuel and a shell, density per unit area, for the mass of the tank structure. Under these assumptions, the following observations can be made.

Observation 1:

Mass of an assembly node can be derived from the sum of its component nodes.

Observation 2:

Mass of an assembly node is unaffected by the relative positioning, rotational and translational, of its component nodes.

With this in mind algorithms for the following cases were developed.

- Centroids with translation and rotation
- Centroids with uniform scaling
- Moments and products of inertia with translation
- Moments and products of inertia with rotation
- Moments and products of inertia with uniform scaling
- Calculation of properties at leaf nodes
- Propagation of properties throughout the hierarchy
- Updating properties on the fly

Implementation of the above algorithms lead to two results. Virtually instantaneous updating of physical properties and the ability to visualize the change of position for calculations such as the center of gravity as the engineer edits the relative position of parts.

## 2.3 Surface Intersection

Many modern modeling packages produce aero-surfaces in some form of bicubic parameterization. For many reasons such a surface representation may not be compatible with analysis required for the surface. For example, when using a finite element analysis to perform a structural analysis patches defining the wing must be made to correspond to the structural elements of the wing. This specific problem will be discussed in depth in the next section of this report. However a key to the reformulation of such surfaces is the ability to cut the original surface into new patches.

Part of the effort of this grant was the development and implementation of a technique for the efficient determination of the intersection of a line and a bicubic surface. An example will be given that illustrates how this can be applied to helping derive new representations of vehicles being analyzed. In addition, it is interesting to note that this technique could also have important applications in the ray tracing of surface which are represented by bicubics and in the calculation of the union and intersection of bicubic surfaces.

Finding the intersection of a line and a bicubic surface results from finding the solution of a pair of degree six algebraic curves. As the solution to most non-linear problems, if the appropriate formulation can be found along with reasonably good starting guesses, Newton's Method will converge both quickly and accurately.

Since it is not known *a priori*, which patch the given line will intersect, a search technique must be employed which compares the given line for potential intersection with each patch. It should be noted that the given line may intersect the bicubic surfaces defining the patches without intersecting the patch itself. That is, the intersection may occur outside of the region of parameterization. Thus it will speed the search considerably if there exists a method for eliminating from consideration those patches that have no chance of intersecting the given line.

Such methods exist for most bicubic parameterizations. For example, both Bezier and B-spline patches lie within the convex hull of their control points, thus if the given line does not intersect the min-max box containing the control points, it cannot possibly intersect the surface. Such methods apply to the surfaces used in the SMART system for which these techniques are principally developed. A summary of the algorithm follows.

1. Set the initial guess for any patch to (0.5,0.5).

    2. Loop over patches until the desired intersection is found or there are no more patches.

    3. Test the current patch to see if it has a potential intersection with the given line (for example the min-max test described above).

    4. For a patch with a potential intersection carry out the Newton's Method described in the previous section.

    5. Check the resulting value for containment in the region of parameterization.

6. End loop.

7. If no convergence has occurred, subdivide the region of parameterization and pick the center of the subdivision as the next candidate for an initial guess. Go to step 2.

A portion of this work appeared in the following publication.

5. V.C. Crisp, J.J. Rehder, and J.L. Schwing, The Intersection of Three-Dimensional Geometric Surfaces, NASA-Langley Technical Paper, 1985.

In addition, part of the work was carried out in a master's project by H-Y. Wang.

**2.4 Offset Surfaces**

The generation of planar surfaces bounded by a curve and its offset has two major applications in SMART. The baseline curve in both situations is typically a cross section of the fuselage. The first problem is involved with defining the internal structures of this cross section and looks to the generation of ring frames and bulk heads. A second related problem has to do with the generation of grids for analysis code either interior or exterior to the body of the vehicle, for example when preparing the vehicle for CFD analysis.

It is obvious that the two problems have similar graphical requirements. However, there are differing mathematical requirements for each of the cases. For instance, in constructing CFD grids, it is usually a requirement that the elements be built so that their natural curvilinear coordinates have a perpendicular intersection with the fuselage. On the other hand, when designing ring frames, it would generally be permissible to build structural elements whose associated coordinate system may intersect the fuselage at any angle. The software has been designed to allow the user the option of specifying which set of conditions are appropriate to the given design requirements.

There are two major problems faced in defining each of these sets of elements for which there is currently no known analytical automated solution. First, the offset curve may be mathematically unacceptable for modelling purposes. For example the curve may become self-intersecting. Secondly, certain fuselage shapes lead to elements that spill into one another.

To overcome these problems our software has been designed to allow the user to easily edit the elements in two fashions. First, the user controls the amount of offset and halts growth when self intersection occurs. The system will then reconfigure the number of defining elements allowing the user to extend further with a new set of elements. To control spill an editing function is provided which allows the user to point at the offending patch or patches and draw them back. It is an effort of continuing research to investigate how to automatically detect such situations and correct for them without requiring input from the user.

In addition to the principal investigators, part of the work on this portion of the grant was part of a master's project for T. Lee.

6. T. Lee, *Offset Surfaces for CAD*, ODU Master's Project, Department of Computer Science, 1988.

### 2.5 Parametric Data

Visualization of data obtained in the analysis process is one of the most important aspects for the aerospace engineer. The conceptual design process uses several different analysis programs to carry out its mission. Specifically, initial geometric design and configuration is carried out in SMART while aero analysis is conducted using APAS and structural analysis uses PATRAN and EAL. SMART has the capability to take the basic model and convert to model representations appropriate for any of these programs. However data generated in the aero-analysis is later required in the structural analysis. Unfortunately, the aero model varies greatly form the structural model and data generated cannot be directly applied.

The solution adopted is to return to SMART, since it is the repository of the original model and has the ability to output data in an appropriate form for either system. Current work allows the data generated by APAS to be read back into SMART. The data is then check for consistency. There are two potential sources of inconsistencies at this point. Namely, that designer at either end, SMART or APAS, may have found it necessary to modify the initial structure with the results not communicated to the other model. SMART will detect and flag such inconsistencies.

Once the parametric data is verified as consistent, SMART proceeds to convert the data from one model to the other. This is carried out in a three stage process. SMART begins by taking the elemental point data of APAS and creates an interpolated surface using the bilinear blending technique. This parametric surface is then sent to the surface reconfiguration module which will redefine the surface with elements that correspond to the underlying structural configuration. The result is a collection of bicubic data patches which are finally output in a form useable by the structural analysis program.

In addition to the principal investigators, part of the work here was conducted by a master's student, Q. Zhu, and resulted in a master's project.

## 2.6 Refinement of Data Definitions in Structures

One of the major achievements of this period of research was the determination of those requirements that structural engineers across the NASA-Langley base need in order to more readily carry out their model preparation. The requirements are contained in the document *System Software Requirements for SMART - Vehicle Structure Modules.*

This document specifies the functional requirements for software components which address the geometric and data modelling needs of the aerospace structural engineer. The modules have been included as part of the SMART package. Hereafter, these software components will be referred to as SMART - Vehicle Structure Module (VSM).

The software requirements document was used by the following groups: the SMART Development Team, structural engineers in VAB, other interested structural engineers and SMART users at NASA LaRC, and the design and implementation group at Old Dominion University. SMART provides conceptual designers with a rapid vehicle geometry prototyping capability. Of current interest is the definition and implementation of those characteristics which would provide the design engineer with a more effective and efficient tool for building structural models. Construction of such models is currently a bottleneck toward carrying out the analysis process. The goal of SMART VSM is to address this bottleneck.

The SMART VSM modules will be a set of software tools designed to aid in the development of geometric and data models to be used in the structural analysis of aerospace vehicles. SMART VSM provides the following general capabilities:

> creating and editing structural elements for the wing and fuselage of a given aerospace vehicle,
>
> integrating wing and fuselage structural assemblies,
>
> integrating tail and fuselage structural assemblies,
>
> remapping of aerodynamic loads data in a manner consistent with the developed structural model,

applying point and area based loads to the model, and

preparing loads data for visual presentation.

The personnel on this grant aided in the development of the implementation plans for the structural analysis portion of the codes including consulting on the general development of data transfer algorithms. Implementation of the actual code for the final portion of these system improvements was given to programmers with Computer Sciences Corporation. A portion of this code, that used to develop interior structure for fuselages, was developed as a Master's project by S. Schwartz.

7. S. Schwartz, *Surface Generation and Editing Operations Applied to Structural Support of Aerospace Vehicle Fuselages*, ODU Master's Project, Department of Computer Science, 1992.

**2.7 Smooth Data Surfaces**

Recently there has been an increasing interest in applying computational fluid dynamics (CFD) analysis to models at the conceptual level. Currently, none of the software systems which generate CFD grids and provide the corresponding analysis, provide tools for model generation.

As a first stage of this, personnel on this grant investigated the smoothing of data sets while reducing the size of the data. In essence algorithms were developed to find non-uniform rational bicubic B-spline approximations to given data sets. NURB's were chosen as the basic building blocks since they exhibit properties necessary to carry out the desired smoothness conditions needed for future surfaces. This work resulted in a Master's project by Carol Macri.

8. C. Macri, *Implementation of an Algorithm for Data Reduction using Cubic-Rational B-Splines*, ODU Master's Project, Department of Computer Science, 1992.

## 3. EASIE

**3.1 Overview**

The following sections contain a summary of EASIE. A more complete version is presented in an AIAA paper prepared jointly with members of VAB and CSC.

9. K. Jones, L. Rowell, and J. Schwing, *Software Tools for the Integration and Execution of Multidisciplinary Analysis Programs*, Proceedings of Aerospace Applications of CAD, AIAA-88-4448, Sept. 1988.

The increasing complexity of today's aerospace vehicles is requiring ever more sophisticated design approaches and analysis techniques which can only be realized by computer-aided engineering systems. In order to accomplish the many iterations required of the designer, these complex analysis codes must be coordinated to provide user-friendly, quick-turnaround, computer-based design systems. EASIE provides a methodology and set of utility routines for a design team to build, maintain, and apply computer-aided design systems consisting of large numbers of diverse, stand-alone analysis codes. The EASIE approach uses a central database containing all the variables (the data dictionary) needed as inputs to the analysis programs that will make up the integrated system. Utilities exist for: constructing the data dictionary and database schema, generating the subroutines to read from or write to the database, incorporating the analysis programs into the database, interactively reviewing and modifying values in the database, incorporating the analysis programs into an interactive executive for easy selection and execution, building menus and procedures to assist the process.

Over the past 25 years, the application of computers in aerospace engineering has increased exponentially. Today, many steps in engineering design and analysis that ten years ago would have taken weeks to accomplish can now be completed in hours. As the computer environment (processors, support software, workstations, communications) has increased in capability, so have the analysis techniques and programs for the engineering disciplines. However, this growth has not been without a price. While information may be obtained at a faster rate and in greater abundance, the absence of standardization and lack of coordination among software developers have resulted in the proliferation of computer programs that are unable to communicate data easily to other programs. The result is often the manual transfer of data from one program to another with the associated manpower loss, time delay, and potential for error introduction. Therefore, the efficiency with which design studies can be accomplished is often more dependent on the coordination of the analysis programs and their data interchange requirements than on the processing speed of the computer system.

In typical design and engineering studies, use will be made of both commercially and in-house developed programs, so that the purchase of integrated analysis tools still leaves at issue the coordination of the commercial database with the local programs. This problem is particularly acute for conceptual design studies which are often characterized by advanced technologies requiring new analysis codes and by numerous computing configurations requiring many analysis iterations and refinements. The constant change in design methods does not allow the system tools to reach the maturity that is possible in production situations.

In particular, the most needed capabilities include easy selection of application programs, quick review and modification of program input/output data, and logging of the actual steps that were executed during a study. Although the application programs used by engineers differ, the design methods are quite similar, and great efficiencies can be gained by providing a computer environment that yields the capabilities mentioned above.

## 3.2 The Approach

Wilhite [4] described the development history of several integrated design systems used for launch vehicle design and gave an overview of the capabilities needed in future enhancements. EASIE implements these needed capabilities and provides both a methodology and a set of software utility programs to ease the task of coordinating engineering design and analysis codes. EASIE consists of a user interface and a set of utility programs which support rapid integration and execution of programs about a central relational database.

The purpose of the EASIE is to aid in the complex task of integrating a collection of application programs into a single system. The tools offer a program integration approach critical in defining and forming the data paths for inter-program communication. Although the acronym EASIE implies simplicity, the problems encountered when integrating two or more programs are often formidable. Consequently, the potential EASIE user should anticipate a relatively high learning curve upon initial exposure to the database management system (DBMS) terminology and software tools. However, one of the principal advantages of using the EASIE tools is that once the initial learning curve is climbed, the learned program integration techniques are applicable to other programs or sets of programs.

In addition to understanding program integration problems (addressed below), the reader should have some experience with DBMS terminology and techniques. In particular, the EASIE tools rely heavily on the techniques of the relational approach to DBMS such as those described by C. J. Date [5]. More specifically, the current version of the EASIE tools is built in FORTRAN 77 based upon the Relational Information Management DBMS [6] (RIM).

A generalization of current program coupling techniques is given to show the basis for the EASIE program approach. There are a number of techniques currently being used to couple independent analysis programs into design systems, and these are generalized here in four categories: close-coupled integration, loose-coupled integration, close-coupled interfacing, and loose-coupled interfacing. The integration technique uses direct communication between a program and a central database or another program. Interfacing means indirect data communication among programs that operate in the normal input-output file mode using intermediary programs to properly format the data. Close-coupling means that the path through the analysis techniques or programs is fixed. With loose-coupling, the programs can be individually executed, or execution paths can be programmed.

Close-coupled integration is analogous to one large program where the analysis is performed by modules (subroutines, segments, or overlays) and data transfer is accomplished by global common and data files. The advantage of close-coupled integration is the speed of execution because data communication is direct. The disadvantage of close-coupled systems are the difficulties of integrating all the analysis pieces into one computer program and of adapting to unique configurations requiring new analysis techniques because these are deeply rooted into the program and are not easily changed. Loose-coupled integration is the technique employed by most business-oriented systems today. A central database management system is used for data communication to all the separate programs. The main advantage of loose-coupled integration is that the programs (analysis techniques) can be developed independently and can later be coupled together to form a complete design system.

The disadvantage of the loose-coupled integration is that these programs must be written to communicate with the database ether initially when the program is developed or later after completion of the program. For a program that has been developed independently of any database considerations and has a large data input with many analysis options, it is an awesome task to integrate the database communication code into the program unless some support tools and automated procedures are developed to simplify the task.

Close-coupled interfacing is one of the first techniques used for coupling independent analysis programs. The interfacing starts with the coupling of one program to another by having the first program write an input file for the next program that is in the precise form of the input file the program used before coupling. This process is repeated from program to program and eventually creates a network of closely-coupled analysis programs that comprise a design system. As the network of programs grows, it becomes more and more difficult to couple new programs into the system. A new program may require input from several programs that are not directly linked. A separate program (called an intermediary program) must be written to read the output of several programs and create an input file for this new analysis program. Because of this problem of de-centralized information and because the analysis is regimentally predefined by the program network, the current approach is to centralize the data. Loose-coupled interfacing uses a central database as a repository of data, but a pre-processor program must be written to retrieve data, transform the data, and format the data into an input file that the program expected before coupling. After program execution, analysis results are written to the database by adding a database code to the analysis program or by creating a post-processor to read output files and place appropriate data in the database. The advantage of interfacing with a pre-processor program is that it requires no knowledge of the internal coding of can be used when program source code is not available as with many commercial programs. Only the program input requirements are needed which are usually well documented. The disadvantages of interfacing are that a pre-processor program must be written for each analysis program and that there is a computer overhead for writing an input file and reading that file by the analysis program. Loose-coupled integration has, for this reason, been selected as the approach of choice, but the EASIE tools will equally support
integration or interfacing requirements as will be discussed in the next section.

The advantages of loose coupling are: (1) the ease of incorporating new programs by using a central database for communication; (2) analysis programs can be executed as needed to aid the design process, or several can be executed for multidisciplinary analyses; and (3) depth of analysis can be changed as the configuration matures. A system architecture based upon loose coupled integration will also allows the development of a single, standard data editor/reviewer when appropriate constructs are created. Once programs are satisfactorily integrated around a common database, a flexible method is needed for accessing the chosen program, gaining help information if necessary, reviewing and modifying the input data, executing the program, and examining the results. These activities can obviously be performed within the computer operating system by using its native command language and editors, but such an approach does not offer any guidance to the designer, nor does it provide any specific framework for creating or tracking design procedures, i.e., sequences of program executions intended to optimize or iterativly arrive at a design goal.

### 3.3 The System

The EASIE tools provide a powerful and flexible executive which presents the user a viewpoint within the integrated design system where either menus or a simple command language can perform activities typical in design studies. The next section describes the functions provided by the EASIE tools to enable construction of loose-coupled, integrated systems of programs, a common editor, and a powerful executive. For more in-depth discussion of these tools, the four-volume set of EASIE documentation provides an overview, user's guides, and installation information.

10. L.F. Rowell, *The Environment for Application Software Integration and Execution (EASIE) Version 1.0 - Volume I - Executive Overview,* NASA TM-100573, August 1988.

11. K.H. Jones, D.P. Randall, S.S. Stallcup, and L.F. Rowell, *The Environment for Application Software Integration and Execution (EASIE) Version 1.0 - Volume II - Program Execution Guide*, NASA TM-100574, July 1988.

12. J.L. Schwing, L.F. Rowell, and R.E. Criste, *The Environment for Application Software Integration and Execution (EASIE) Version 1.0 - Volume III - Program Execution Guide*, NASA TM-100575, April 1988.

13. D. P. Randall, K.H. Jones, and L.F. Rowell, *The Environment for Application Software Integration and Execution (EASIE) Version 1.0 - Volume IV - System Installation and Maintenance Guide,* NASA TM-100576, April 1988.

## 4. User Interfaces Implemented

### 4.1 X-Windows and the EASIE Interface

Once the initial version of EASIE described above was released to the public, the importance of the menu driven aspect of EASIE became obvious. The initial user interface was designed for simple ascii terminals and did not take advantage of advances in technology for presenting the user interface. On the forefront of these advances is the windowing system for the Athena project at MIT, X-Windows. Most of the software in this system is in the public domain and hardware in the form of X-servers and X-terminals is rapidly becoming available. We did some crystal gazing, and it was apparent that this combination of public domain software and low-cost hardware would lead to the next revolution of the user interface design.

With this in mind, work under this grant developed a new user interface for EASIE. This work has been carried out as Master's projects by students Y-C. Kao and C-L. Tsai. The research carried out in this area resulted in two master's projects. The effort was divided since EASIE can operate in two highly different modes. These are the complete command environment, CCE and the application derived environment. They developed MOTIF based interfaces for both the ADE and CCE modes of execution of EASIE.

14. C-L. Tsai, *User Interface Design for EASIE*, ODU Master's Project, Department of Computer Science, 1992.

15. Y-C. Kao, *Application Driven Interface Generation for EASIE*, ODU Master's Project, Department of Computer Science, 1992.

## 4.2 The POST User Interface

Many modern analysis codes provide an excellent analysis capability with a high degree of flexibility. Within a given program the high degree of flexibility leads to complex data structures. The user of the program is made responsible for the creation and proper formatting of the input data file. The user is also responsible for tracking the definition of the appropriate sets of input parameters.

Consider the example of the Program to Optimize Trajectories, POST. POST is an event driven program, the input to which falls into the above categories. POST is batch oriented taking input data from an ascii *event* file. The flexibility of POST leads to a high degree of interdependence in the definition of data items. For example, when an alternate method of guidance is selected, a completely different set of input parameters must be specified. POST provides no tools for the definition of such input.

This research was designed and implemented as a multi-stage solution which provided users with a new interface for manipulating the variables of POST. In the first stage, a prototype was generated to deal with only a limited set of the parameter variables using ascii interface techniques. With a positive response from the user community, an interface was developed to handle all POST variables including tabular variables. Again, the response was positive. The final stage was to prepare a prototype that provided users with other that an ascii interface; namely an X-based interface. The prototype was developed using the graphical user interface extensions to the EASIE system discussed above.

In addition to the principal investigators, work on this portion of the project was carried out by three undergraduate students, A. Grimm, S. Casey and W. Denny and one doctoral student, V. Bokka.

### 4.3 SMART and the On-Line Help System

Until recently, one of the least developed portions of many systems was the help provided to the users of the system. In addition, tools for system developers to easily create documentation and help facilities were also lacking. However, much current research has begun to address the issues of design and implementation that arise in on-line help systems. A list of such issues and references can be found in [7].

Such research has shown that the major uses of help systems will be three fold: first, for the presentation of summary and tutorial information, secondly as a general source of on-line system capabilities and finally, as a detailed compendium of system operation. As a master's project research assistant J.L. Spangler looked into the development of such a system for SMART. A complete system called ManualWriter was the result.

16. J.L. Spangler, *ManualWriter*, ODU Master's Project, Department of Computer Science, 1987.

## 5. Data Management Concerns

### 5.1 Data Consistency

One of the challenges facing the designers of an integrated computer-aided engineering system is to blend in a robust and efficient way a wide variety of independently developed design and analysis programs, each with its specific requirements for input and output. As previously noted, EASIE provides a methodology and set of utility routines to support building, maintaining, and applying computer-aided design systems consisting of large numbers of diverse, stand-alone analysis codes. Conducting the engineering activity usually calls for manually changing the contents of a given set of input variables in the data base and for the subsequent execution of a set of analysis programs. As a result of the computations, other data base values will be altered. To support a high-productivity environment, a tool is needed to make the engineer aware of potential inconsistencies introduced in the data base as a result of the experiment. A technique is described for identifying, beforehand, the set of all
the variables in the data base susceptible to inconsistency.

As an example consider the Space Station Conceptual Design Model (SSCDM) programs. The purpose of the program is to identify key subsystems technology alternatives and logistics requirements and then to estimate the size of a cylindrical space station unit. Modules in SSCDM include configuration analysis, environmental control/life support systems, communications, data processing systems, stability and control systems, electrical power and thermal control systems, structures and reaction control systems. The SSCDM originally configured a total system with no user interference after input, but by embedding SSCDM in EASIE the design engineer can concentrate on developing and refining certain subsets of the entire station.

The loose coupling of EASIE allows the designer to iterate through system components as desired. During this process, the designer will use engineering insight to define and possibly modify those input parameters which may also be outputs defined by other analysis modules. This technique helps decrease development time but may introduce inconsistencies into the data base. The purpose of this work is to define techniques to identify those inconsistencies. Parameters will be flagged in the data base so they can be queried at any time to determine the consistency status.

Work on this grant has formalized the essence of the design process and designed a solution to the database integrity problem. A full description of this process has been published by both of the principal investigators in conjunction with members of VAB and ACD.

17. K. Jones, S. Olariu, J.L. Schwing, and A.W. Wilhite, *A Database Consistency Checker for EASIE*, Proc. Fifth International Conf. on CAD/CAM and Robotics, Springer-Verlag, v. 1, 55 - 60, 1990.

## 5.2 A Knowledge Base

The purpose of this section is to discuss the vision for a knowledge base package to enhance the availability of engineering and management data. This vision was developed in conjunction with a working group in the Vehicle Analysis Branch.

For over three decades, researchers at NASA and in the aerospace industry have been involved in the development of all aspects of technology that support the design and construction of aerospace vehicles. Yet today when a new design efforts are requested, access to previous work is haphazard at best, depending upon the prior knowledge of design team members and the success of library searches and contacts with others involved in this research. The major thrust of the proposed knowledge base development is to help support the decisions that will be made by aerospace leaders about the future of space transportation by a knowledge base, STKB, which contains decision making information and aids in its dissemination.

Mere analysis of one-point designs based upon single-mission requirements lend of little help to the decision making process when those mission requirements are changed in any way. These studies take several years and unless that exact concept is being constructed, they currently add little value to the knowledge necessary to make the decision. Similarly, results of prior studies may or may not be known and their impact may also be minimal. In contrast, STKB would provide multi-media electronic access to a full repository of databases and studies that have been conducted over the years in the framework of a decision support system. It is envisioned that STKB would support management and researchers at all levels. For example, at NASA and in industry upper levels of management might search for answers to questions like: for a given mission, what is the single stage to orbit vehicle, SSTO, with the lowest recurring cost or what expendable launch vehicles are there in the world with a payload capacity of 5,000 pounds to low earth orbit? On the other hand, one could imagine study leaders or discipline experts looking at issues like: graphing the dry weight and gross weight versus payload or determine what avionics systems are developing with technology level 6.

Clearly, the knowledge that has been collected over the years comes in many forms: databases, trade studies, graphs, charts, summaries and many more. Just as clearly, this knowledge is distributed widely across the country. Also as noted above the range of potential users as well as their questions is large. All of these factors contribute to the realization that the creation of STKB will be an complicated task. It is important to note however that the computing environment has arrived at a stage where support of these objectives is now feasible due to advances such as the advent of high speed networks, distributed information support and retrieval systems and the proliferation of powerful, high speed workstations to name just a few. As an additional note, it is important to set STKB in a decision support framework. Much work has been done in recent years in the areas of decision support systems and the analytic hierarchy process which should form an important base for this framework.

The first version of the system would concentrate on the development of STKB for the studies, previous, current and proposed, of the Vehicle Analysis Branch. In this form, the STKB is viewed as an integral part of a decision support system for the branch. The system would bring together various databases that exist for structures, aerodynamics, operations, costs, and technologies into a unified database that will be used in a variety of models for analysis. The results of this analysis will be combined with point design studies of representative concepts to build a matrix of results. Such a matrix will provide the sensitivity results needed to support queries made to STKB.

### 5.3 Version Control

With the profusion of various enhancements to analysis tools and, as mentioned throughout the discussion above, a profusion of data and analysis studies, it has become imperative to track changes in software and data sets. Many tools now exist to help aid in this process. At the request of VAB, work on this grant included development of techniques to easily track and control versions of programs and data through the RCS version control system.

## 6. Parallel Algorithms

It has become clear that much of the future improvements in computing power will arise in the use of parallel and/or distributed computing environments. Indeed, this can be seen in the new SGI computers that have been brought in to support the VAB analysis and design programs. They are all multi-processor machines. While these machines can and do provide a certain amount of automatic algorithm adjustment to take advantage of this environment, true efficient use of any parallel or distributed environment requires careful investigation of the algorithms being developed. Algorithms initially developed for sequential single processor machines may not perform anywhere near optimally under automated conversion.

It is the belief of the principal investigators of this grant that the next major impact on the development of analysis programs will come via the proper utilization of parallel and distributed processing. Further, this can only occur if the proper ground work is developed. The personnel on this grant have been extremely productive in providing both the basis for understanding algorithm development on a number of exciting new architectures and also in a number of application areas that will have direct impact on research being conducted by Joe Rehder of formerly of VAB. This work is contained in the following publications.

**6.1 Journal Publications**

18. S. Olariu, J.L. Schwing, and J. Zhang, *Optimal Parallel Algorithms for Problems Modeled by a Family of Intervals*, IEEE Transactions of Parallel and Distributed Systems, v.3 no. 3, 364 - 374, 1992. A preliminary version was published in Proc. 28-th Annual Allerton Conf. on Communication, Control, and Computing, 282 - 291, 1990.

19. S. Olariu, J.L. Schwing, and J. Zhang, *On the Power of Two-Dimensional Processor Arrays with a Reconfigurable Bus System*, Parallel Processing Letters, no. 1, 29 - 34, 1991.

20. S. Olariu, J.L. Schwing, and J. Zhang, *Optimal Parallel Encoding and Decoding Algorithms for Trees,* International Journal of Foundations of Computer Science, v. 3 no. 1, 1 - 10, 1992. A preliminary version of this paper appeared in Proc. 1991 ACM Computer Science Conference, San Antonio, Texas, 1 - 10, March 1991.

21. S. Olariu, J.L. Schwing, and J. Zhang, *Integer Problems on Reconfigurable Meshes, with Applications*, Journal of Computer and Software Engineering, accepted for publication. A preliminary version has appeared in Proceedings of the 29th Annual Allerton Conference on Communications, Control, and Computing, 821 - 830, 1991.

22. S. Olariu, J.L. Schwing, and J. Zhang, *A Constant-time Channel Assignment Algorithm for Reconfigurable Meshes*, BIT, v. 32, 586 - 597, 1992.

23. S. Olariu, J.L. Schwing, and J. Zhang, *Fast Computer Vision Algorithms on Reconfigurable Meshes*, Image and Vision Computing Journal, v.10 no. 9, 610 - 616, 1992. A preliminary version of this work has appeared in Proceeding of the 6th International Parallel Processing Symposium, Beverly Hills, 1992.

24. D. Bhagavathi, P. Looges, S. Olariu, J.L. Schwing, and J. Zhang, *Selection on Meshes with Multiple Broadcasting,* BIT, v. 33, 7 - 14, 1993.

25. R. Lin, S. Olariu, J.L. Schwing, and J. Zhang, *Simulating Enhanced Meshes with Applications*, Parallel Processing Letters, v. 3, 59 - 70, 1993.

26. S. Olariu, J.L. Schwing, and J. Zhang, *Applications of Reconfigurable Meshes to Constant-time Computations*, Parallel Computing, v. 19, 229 - 237, 1993. A preliminary version of this paper appeared as *Constant Time Integer Sorting on an n x n Reconfigurable Mesh* in Proc. of the International Phoenix Conf. on Computers and Communications, Scottsdale, Arizona, 480 - 484, 1992.

27. S. Olariu, J.L. Schwing, W. Shen, L. Wilson, and J. Zhang, *A Simple Selection Algorithm for Reconfigurable Meshes,* Parallel Algorithms and Applications, v.1, no. 1, 29 - 42, 1993. A preliminary version of this work appeared in Proc ISMM Conference on Parallel and Distributed Systems, Pittsburgh, 257 - 261, October 1992.

28. S. Olariu, J.L. Schwing, and J. Zhang, *Data Movement Techniques on Reconfigurable Meshes with Applications*, International Journal of High Speed Computing, v. 6, no. 2, 311 - 323, 1994. A preliminary version of this paper appeared in Proc. of the International Phoenix Conf. on Computers and Communications, Scottsdale, Arizona, 472 - 479, 1992.

29. R. Lin, S. Olariu, J.L. Schwing, and J. Zhang, *An Efficient EREW Algorithm for Minimum Path Cover and Hamiltonicity on Cographs*, Parallel Algorithms and Applications, v. 2, nos. 1 & 2, 99 - 114, 1994. A preliminary version of this paper appeared in Proc. of the 26th Annual Hawaii International Conference on Systems Science, Wailea, Maui, v. II, 283 - 292, January 1993.

30. D. Bhagavathi, S. Olariu, J.L. Schwing, and J. Zhang, *Convex Polygon Problems on Meshes with Multiple Broadcasting*, Parallel Processing Letters, v. 2, nos. 2 & 3, 249 - 256, 1992.

31. S. Olariu, J.L. Schwing, and J. Zhang, *Fast Component Labelling and Convex Hull Computation on Reconfigurable Meshes*, Image and Vision Computing Journal, v. 11, no. 7, 447 - 455, 1993. A partial version of this paper appeared as *Fast Component Labeling on Reconfigurable Meshes* in Computing and Information - Proc. International Conference on Computing and Information, Toronto, 121 - 124, 1992.

32. D. Bhagavathi, P. Looges, S. Olariu, J.L. Schwing, and J. Zhang, *A Fast Selection Algorithm on Meshes with Multiple Broadcasting,* IEEE Transactions of Parallel and Distributed Computing, v. 5, no. 7, 772 - 777, 1994. A preliminary version of this paper appeared in Proc. International Conference on Parallel Processing, St. Charles, Illinois, III-10 - III-17, 1992.

33. S. Olariu, J.L. Schwing, and J. Zhang, *Computing the Hough Transform on Reconfigurable Meshes*, Image and Vision Computing Journal, v. 11, n. 10, 623 - 628, 1993. A preliminary version of this paper appeared in Proc. of Vision Interface, 1992, Vancouver, British Columbia, May 1992.

34. D. Bhagavathi, S. Olariu, J.L. Schwing, and J. Zhang, *Time- and Cost-Optimal Parallel Algorithms for the Dominance and Visibility Graphs*, VLSI Design, accepted for publication. A preliminary version of this paper appeared in The 7th International Conference on VLSI Design, Calcutta, India, to appear, January 1994.

35. S. Olariu, J.L. Schwing, and J. Zhang, *Optimal Convex Hull Algorithms on Enhanced Meshes*, BIT, v. 33, pp. 396 - 410, 1993.

36. S. Olariu, J.L. Schwing, and J. Zhang, *Interval Graph Problems on Reconfigurable Meshes*, ORSA Journal on Computing, accepted for publication.

37. S. Olariu, J.L. Schwing, and J. Zhang, *Computing on Dynamic Buses*, Parallel Processing Letters, accepted for publication. A preliminary version of this paper appeared as Computing on Reconfigurable Buses, Proc. of the International Phoenix Conf. on Computers and Communications, Scottsdale, Arizona, 30 - 36, March 1993.

38. G-H. Chen, S. Olariu, J.L. Schwing, B-F. Wang, and J. Zhang, *Constant-Time Tree Algorithms on Reconfigurable Meshes of Size n x n*, Journal of Parallel and Distributed Computing, accepted for publication. A preliminary version of this paper appeared in Proc of ICPADS '93, 559 - 566, December 1993.

39. D. Bhagavathi, S. Olariu, J.L. Schwing, W. Shen, L. Wilson, and J. Zhang, *Convexity Problems on Meshes with Multiple Broadcasting*, Journal of Parallel and Distributed Computing, accepted for publication. A preliminary version of this paper appeared in Proc. 4th Annual Canadian Computational Geometry Conference, St. John's, 365 - 370, August 1992.

40. D. Bhagavathi, V. Bokka, H. Gurla, S. Olariu, J.L. Schwing, I. Stojminovic and J. Zhang, *Time-Optimal Visibility-Related Algorithms on Meshes with Multiple Broadcasting*, IEEE Transactions on parallel and Distributed Systems,, accepted for publication. Preliminary versions of this paper appeared in Proc ASAP '93, Venice, Italy, 226 - 237, October 1993 and Proceedings of IPPS '94, Cancun, Mexico, 110 - 114, April 1994.

41. V. Bokka, H. Gurla, S. Olariu, J.L. Schwing, and J. Zhang, *Constant-Time Convexity Problems on Dense Reconfigurable Meshes*, Journal of Parallel and Distributed computing, accepted for publication. A preliminary version of this paper appeared in Proc of the 23rd Annual International Conference of Parallel Processing, III-210 - III-213, August 1994.

### 6.2 Refereed Conference Proceedings and Refereed Technical Publications

42. S. Olariu, J.L. Schwing, and J. Zhang, *An Optimal Parallel Algorithm for Channel-Assignment*, Proc. Fifth International Conf. on CAD/CAM and Robotics, Springer-Verlag, v. 2, 203 - 216, 1990.

43. S. Olariu, J.L. Schwing, and J. Zhang, *A Fast Adaptive Convex Hull Algorithm on Processor Arrays with a Reconfigurable Bus System*, Proc. Third Annual NASA Symposium on VLSI Design, 13.2.1 - 13.2.9, 1991.

44. S. Olariu, J.L. Schwing, and J. Zhang, *Fundamental Algorithms on Reconfigurable Meshes*, Proc. 29-th Annual Allerton Conf. on Communication, Control, and Computing, 811 - 820, 1991.

45. S. Olariu, J.L. Schwing, and J. Zhang, *Fast Mid-level Vision Algorithms on Reconfigurable Meshes*, Parallel Computing: From Theory to Sound Practice, Proceedings of EWPC '92, IOS Press, 188 - 191, 1992.

46. R. Lin, S. Olariu, J.L. Schwing, and J. Zhang, *Sorting in O(1) time on a Reconfigurable Mesh of Size nxn*, Parallel Computing: From Theory to Sound Practice, Proceedings of EWPC '92, Plenary Address, IOS Press, 16 - 27, 1992.

47. S. Olariu, J.L. Schwing, and J. Zhang, *Efficient Image Processing Algorithms for Reconfigurable Meshes*, Proc. of Vision Interface, 1992, Vancouver, British Columbia, May 1992.

48. S. Olariu, J.L. Schwing, and J. Zhang, *Time-Optimal Sorting and Applications on nxn Enhanced Meshes,* Proc. IEEE Internat. Conf. on Computer Systems and Software Engineering, Comp Euro `92, The Hague, 250 - 255, 1992.

49. S. Olariu, J.L. Schwing, and J. Zhang, *Interval-Related Problems on Reconfigurable Meshes*, Proc. ASAP '92, Berkeley, 445 - 455, August 1992.

50. S. Olariu, J.L. Schwing, and J. Zhang, *Efficient Image Computations on Reconfigurable Meshes*, Proc. of CONPAR '92, Lyon, France, 589 - 594, September 1992.

51. S. Olariu, J.L. Schwing, and J. Zhang, *Convex Polygon Problems on Reconfigurable Meshes*, SPIE Conference on Vision Geometry, Boston, 111 - 121, November 1992.

52. R. Lin, S. Olariu, J.L. Schwing, Z. Wen and J. Zhang, *An Optimal Parallel vertex Cover Algorithm for Cographs*, Proc ISCIS Conference, Antalya, Turkey, 49 - 55, November 1992.

53. S. Olariu, J.L. Schwing, and J. Zhang, *Geometric Problems on Meshes with Multiple Broadcasting,* Proc ISCIS Conference, Antalya, Turkey, 41 - 47, November 1992.

54. D. Bhagavathi, H. Gurla, R. Lin, S. Olariu, J.L. Schwing, and J. Zhang, *Square Meshes Are Not Optimal For Convex Hull Computation*, Proc of the 22nd International Conference on Parallel Processing, St. Charles, IL, III-307 - III-310, August 1993.

55. D. Bhagavathi, H. Gurla, S. Olariu, J.L. Schwing, W. Shen, L. Wilson, and J. Zhang, *Time- and VLSI-Optimal Sorting on Meshes with Multiple Broadcasting,* Proc of the 22nd International Conference on Parallel Processing, St. Charles, IL, III-192 - III-195, August 1993.

56. V. Bokka, H. Gurla, S. Olariu, and J.L. Schwing, *Constant-Time Convexity Problems Problems on Reconfigurable Meshes,* Lecture Notes in Computer Science - Parallel & Distributed Computing, Theory and Practice: Proceedings of CFPAR '94, 169 - 180, May 1994.

57. D. Bhagavathi, V. Bokka, H. Gurla, S. Olariu, and J.L. Schwing, *Time-Optimal Tree Computation on Sparse Meshes,* Proceedings WG94, to appear, June 1994.

58. V. Bokka, H. Gurla, S. Olariu, and J.L. Schwing, *Constant Time Triangulation Problems on Reconfigurable Meshes,* Proceeding of ASAP '94, 357 - 368, August 1994.

59. R. Lin, S. Olariu, and J.L. Schwing, *An Efficient VLSI Architecture for Digital Geometry,* Proceeding of ASAP '94, 392 - 403, August 1994.

60. D. Bhagavathi, V. Bokka, H. Gurla, R. Lin, S. Olariu, J.L. Schwing, W. Shen, and L. Wilson, *Time-Optimal Multiple Rank Computations on Meshes with Multiple Broadcasting,* Proc of the 23rd Annual International Conference of Parallel Processing, III-35 - III-38, August 1994.

61. S. Olariu and J.L. Schwing, *A Faster Sorting Algorithm in the Broadcast Communication Model,* Proc of IPPS '95, to appear.

62. V. Bokka, H. Gurla, S. Olariu, and J.L. Schwing, *Time- and VLSI-Optimal Convex Hull Computation on Meshes with Multiple Broadcasting,* Frontiers of Massively Parallel Computing '95, 506 - 513, February 1995.

## 7. Consulting Activities

### 7.1 Optimization and Automatic Differentiation

Many applications of mathematical analysis require the values of derivatives of the functions being considered. For example engineering design optimization frequently involves the derivative of the measure of merit function as a predictor of how to improve parameter values. Over the years, several automatic differentiation tools have been created to automatically compute the derivatives of computer programs.

As noted in [8], basically there are four methods to compute the derivatives necessary for a given optimization. The first is by hand which is notoriously prone to human error especially as the complexity of the problem increases. Next is the method of divided differences which may suffer from inaccuracies due to the fact that derivatives are only approximated. Third is symbolic differentiation which may be highly inefficient, particularly when one has a large number of subexpressions in different derivative expressions. Finally, there is automatic differentiation which derives code for the derivatives based upon the code of the given functions and equations.

In his paper, Juedes [9] surveys 28 different software packages for automatic differentiation and examines the variety of techniques and the maximum degree of derivatives that can be computed. Five basic classes are determined. These include the forward method, the reverse method, integral tools, operational tools and symbolic tools. Recently work at Argonne National Labs [8] has produced a package that seems to combine many of the advantages of each of these. This tool is known as ADIFOR.

Due to the fact that automatic differentiation produces exact derivatives of the analysis code, it appears to produce results as accurately as the original code allows. When the optimization problem can be characterized in a straight forward manner, the application of the automatic differentiation is also straight forward. Unfortunately, even a tool like ADIFOR is not so automatic in less straight forward cases.

Work under this grant has successfully tested the ADIFOR procedure with several simple cases. In addition, consideration has been given to the application of ADIFOR to the analysis program CONSIZ. This program provided a special problem to ADIFOR. CONSIZ is a vehicle weight and sizing program which provides for highly variable models. Indeed sizing equations are not coded into the program itself but rather read in as a series of coefficients and operations at execution time. Therein lies the problem for ADIFOR. It is a preprocessor and must know what lines of code are to be differentiated prior to execution.

A novel technique has been developed to automatically preprocess the CONSIZ input file before the application of ADIFOR so that relevant information is available. Work on this preprocessor was done by the principal investigators in conjunction with a doctoral student, H. Gurla.

## 7.2 Integration and Application of Analysis Tools

Continuing research under this grant has been focused on the design and implementation of computer aided design tools to support conceptual level aerospace design. This has included the use of a number of finite element design and analysis codes involved in several design studies currently underway in the VAB. As conceptual aerospace design moves to consider future vehicles such as the Advanced Manned Launch System (AMLS) and the Personnel Launch System (PLS), one factor conceptual designers face is a pressing need to enhance their analysis capabilities as traditional formulae and historical data are exceeded by new conditions and requirements. The effort to generate new formulae and tables then proceeds to the application of higher order analysis packages such as EAL and PATRAN. Geometric input for structural development in such packages is tedious at best. Integration of models through the vehicle provided by SMART is the best answer to this problem. In order to provide the detailed understanding necessary to design and implement the integration of SMART and advanced structural analysis programs as well as providing insight into the analysis programs themselves, the services of an expert are required. This support has been supplied by J.C. Robinson.

For example, one task has been the development of a finite element model of the primary structure of the air-breathing first stage of a two stage launch vehicle. The model includes a lifting-body configuration fuselage, a discrete wing and internal multi-bubble tanks. Development of this model involved converting sections from hardcopy with linear and quadratic scaling between the sections. At present the vehicle is changing and the analytic model needs to be changed to meet these changes.

A major portion of the effort has been expended on the study of a two-stage launch vehicle involving an air-breathing first stage and a rocket second stage. The primary task has been the development of a finite element model of the primary structure of the first stage. The model includes a lifting-body configuration fuselage, a discrete wing and internal multi-bubble tanks. The model was developed from sections of a hard-copy drawing that was scaled by hand with both linear and parabolic interpolation between sections. Multi-bubble tank coordinates for both cylindrical a barrel sections and ellipsoidal domes were calculated in the program to provide an accurate representation of a membrane tank configuration. At present, the vehicle configuration is changing and the analytical model will be modified to reflect these changes.

Another major activity has been the continuing support of the Personnel Launch System (PLS) study. This activity has included three trips to the Rockwell International Space Systems Division (Los Angeles, CA) for program reviews and two one-day trips to North Carolina State University for reviews of NC State's efforts in fabricating a full-scale mock-up of the proposed PLS vehicle. In addition, there are weekly meetings and local contractor presentations to attend.

Secondary activities have included the editing of two AIAA papers for presentation at the SDM Conference, one of which required extensive recalculation and rewriting. Other activities include an investigation of the strength and buckling sensitivities of a large aeroshell model and familiarization with the Computational Structural Mechanics Testbed for Structural Analysis program.

Additional work during this period has been the analysis of the Rockwell developed configuration of the HL-20 vehicle. The existing HL-20 finite-element model was modified to conform to the Rockwell concept of the HL-20. This required the addition of a partial circular cylindrical pressure shell and a flat floor area. The existing upper part of the vehicle exterior forms the remainder of the pressure shell. Existing frames in the previous conformal shell model were removed from the access doors. Frames were modeled for the new pressure shell structure. The remainder of the conformal model was converted into access panels (large doors) on the upper surface and a frame stiffened, heatshield structure on the lower surface.

The model was analyzed and resized for five loading conditions. Two loading conditions controlled the sizing of most of the structure. The first is the internal pressure case plus the 3-G axial acceleration of a normal launch. The second is an abort condition which subjects the vehicle to an 8-G axial acceleration and a 10 psi over-pressure due to the explosion of the launch vehicle. The Rockwell concept with doors exterior to the pressure shell causes the external and internal pressure loadings to be supported by two different load paths. The considerable pressure loads in the vehicle exterior caused by the abort condition required that part of the structure to be resized to resist over-all buckling of the vehicle. This capability was not present in the resizing program.

The creation and implementation of an algorithm to size a finite element model to prevent over-all buckling was the second task for this period. Rigorous mathematical programming methods exist that may be used to resize a structure for buckling but application at a level consistent with the strength resizing (approximately 3000 elements and five load cases) is not practical for preliminary design efforts. The method implemented uses EAL generated strain-energy-densities for the critical buckling mode shapes and several small AWK scripts to calculate new element sizes. While the method is heuristic in nature, it appears to provide a "hands-off" solution to the problem.

Finally, the principal investigators have consulted with the members of the Vehicle Analysis Branch in the areas of graphics, software integration, numerical computations, and data management. Some of the consulting work described in this section has resulted in relevant publications.

63. J.C. Robinson and D.O. Stanley, *Structural and Loads Analysis of a Two-Stage Fully Reusable Advanced Launch System*, Fourth Symposium on Multi-disciplinary Analysis and Optimizations, AIAA paper # AIAA-92-4774, September 1992.

64. C Cruz, W. Engelund, and J.L. Schwing, *Conceptual Level Aerodynamic Heating Predictions Using the Aerodynamic Preliminary Analysis System*, Proc. AIAA Aircraft Design Conference, paper no. AIAA-91-3087, 1991.

## 8. Summary of Results

Work performed under this grant includes the completion of two major software projects, numerous added features and upgrades, integration of a variety of design tools, implementation of data management tools, and the design of user interfaces for several software tools. In addition to the software generated the work has resulted in the publication of 64 journal publications, conference proceedings, refereed technical reports and master's project reports.

The major thrust of the work centered on the development of the software systems SMART and EASIE. Major additional features include: real-time properties calculation, surface intersection calculation, determination of offset surfaces, calculation of parametric data surfaces for visualization, significant upgrade in the design of aerospace vehicle structures, and computation of smooth surfaces from noisy data. Design of user interfaces include the development of on-line help in the form of ManualWriter, an X-based interface for EASIE, and a new interface for POST. Data management work included the development of a database consistency checker, version control and a proposal for a space technology knowledge base. In the area of multidisciplinary optimization, work was performed to look at the feasibility of using automatic code differentiation as a tool. Significant progress was made on the understanding and construction of fundamental parallel algorithms. Finally work also included consulting expertise in the design, use, integration and application of computer aided design tools.

The work associated with this grant has been carried out by the principal investigators, two research professionals, four doctoral students, ten master's students, and three undergraduate students.

All software products were either developed an the computer systems of the Vehicle Analysis Branch or have been delivered to the branch, setup, tested and are being run currently. Copies of all technical reports and master's projects cited in this report have been delivered to the Vehicle Analysis Branch. Cited journal publications and conference proceedings are in the open literature and are available as listed.

## 9. Additional References

1. *Computer Aided Design Modelling, Systems Engineering, CAD-Systems*, J. Encatnacao Ed., Springer-Verlag, 1980.

2. M.P. Groover and E.W. Zimmers, Jr., *CAD/CAM, Computer Aided Design and Manufacturing*, Prentis Hall, 1984

3. T-R. Hsu and D.K. Sinha, *Computer Aided Design: An Integrated Approach*, West, 1992.

4. A. Wilhite, S.C. Johnson, and V. Crisp, *Integrating Computer Programs for Engineering Analysis and Design*, Proceedings of the 21st Aerospace Sciences Meeting, AIAA Paper No. 83-0597, January 1983.

5. C.J. Date, *The Systems Programming Series Volume I - An Introduction to Database Systems*, Third Edition, February 1982.

6. Boeing Computer Services Company, *BCS RIM-Relational Information Management System Version 6.0 Users Guide*, May 1983.

7. G. Kearsley, R.L. Campbell, J. Elkerton, W. Judd, and J. Walker, *On-Line Help Systems: Design and Implementation Issues*, panel discussion, Human Factors in Computing Systems, SIGCHI Conference Proceedings, 287-289, 1988.

8. C. Bishof, A. Carle, G. Corliss, A. Griewank and P. Hovland, *Generating Derivative Codes from FORTRAN Programs*, MCS-P263-0991, Argonne National Labs, 1991.

9. D. Juedes, *Automatic Differentiation of Algorithms: Theory, Implementation and Application*, SIAM, 1991.