JOURNAL OF SPACECRAFT AND ROCKETS
Vol. 31, No. 2, March–April 1994

Object-Oriented, Optimization-Based Design of Satellite Structures

Srinivas Kodiyalam,* Catherine M. Graichen,† Isobel J. Connell,‡ and Peter M. Finnigan§
General Electric Corporate Research and Development Center, Schenectady, New York 12301

This paper focuses on the application of rigorous optimization methods to the design of complete satellite structural subsystems. It discusses a software system being developed which is based on a combination of finite-element analysis using MSC/NASTEAN, numerical optimization, approximation concepts, and object-oriented technology. The benefits of object-oriented design, in the context of satellite structural optimization, are identified. In addition, an approach to system optimization, including multilevel and coupled subsystem optimization is explored. Finally, the application of this software to the actual design of two satellite structures is presented.

Nomenclature

$F(X)$	= design optimization objective function
f_j	= jth frequency
$g_j(X)$	= jth design constraint
g_{lb}	= lower bound on design constraint
g_{ms}	= margin of safety constraint
g_{ub}	= upper bound on design constraint
K	= structural stiffness matrix
M	= structural mass matrix
X	= vector of design variables
x_i	= ith design variable
λ_j	= jth eigenvalue
σ_{allow}	= minimum allowable stress
σ_{max}	= maximum stress
ϕ_j	= jth mode shape vector

I. Introduction

TODAY, more than ever before, the need to reduce design cycle time, reduce weight, improve product quality and reliability, and lower cost are primary drivers in the design of satellites in general and the satellite structural system in particular. These requirements constitute a complex design problem that is difficult, if not impossible, to seriously address if automated methods are not employed.

This paper addresses the application of rigorous optimization methods to the design of the complete satellite structural subsystem, with the objective being to minimize the weight, subject to constraints on frequency, strength, and buckling. Implicit in this statement is the expectation that the time required to produce optimal designs will dramatically decrease, that engineering productivity will be substantially increased, and that the overall quality of designs will be better than that which could be achieved with conventional design approaches.

The satellite structural subsystem is comprised of a set of structural subassemblies, including the core bus structure and those structures intrinsic to such things as the solar array panels and antennas. There are several interrelated obstacles that must be overcome when automating the optimization process of a complete satellite structure including: computational efficiency, finite-element modeling, design problem formulation, and representation.

The approach adopted here is to use numerical optimization, in combination with the MSC/NASTRAN finite-element code, and object-oriented technology coupled with a general purpose finite-element preprocessor.

Computational efficiency is critical because of the iterative nature of the design problem, the potentially large number of design parameters, and the size of the finite-element model for the complete satellite structure. Approximation concepts combined with analytical design sensitivity calculations are used to address this issue.

Finite-element modeling is a concern because commercial systems do not have the intrinsic capabilities to directly specify the design problem or the physical decomposition of a satellite structure into its natural subsystems. For example, satellites are typically comprised of honeycomb sandwich panels, where the face sheets may be metallic or composite materials. Thicknesses of the core and face sheets, as well as details of the composite ply schedule, can be design variables. This information needs to be communicated between the finite-element modeling system and the structural optimization system. Furthermore, an automated system must be able to derive the analytical model for each subsystem from the overall satellite model with minimal user specifications.

Design problem formulation involves specifying an objective function, along with constraints, design variables, and bounds on the design variables. It is important to be able to capture this information, relate it to the finite-element model, and manipulate it under program control. Clearly, the design problem for each subsystem is different. Formulation of the system level optimization design problem is not a simple aggregation of the subsystem optimization problems, but rather must be able to dynamically capture the constraints imposed among the various subsystems.

Finally, the representation of the finite-element model and the design problem is critical, because it directly impacts code development, especially the flow control of the iterative design process. We have found that object-oriented technology offers a powerful mechanism for designing and implementing advanced analysis procedures such as design optimization. Independent of the modularity enforced by object-oriented programming, the principal benefit is the logical mapping that occurs between physical objects and processes, and the corresponding computer objects. Although object modeling techniques are beneficial for representing structures in a single-level optimization environment, they are particularly important for multilevel optimization and coupled subsystem optimization where decomposition and representation of structures, along with more sophisticated control strategies, are mandated. Figure 1 shows the general architecture for system optimization.

Section II describes the design optimization information for the satellite problem addressed, identifying the design variables, objective function, and constraints. Section III discusses some of

Received Dec. 26, 1992; revision received March 22, 1993; accepted for publication March 22, 1993. Copyright © 1993 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Mechanical Engineer, Engineering Mechanics Laboratory, Building K-1, Room: 2A25, P.O. Box 8. Member AIAA.

†Software Engineer, Engineering Mechanics Laboratory.

‡Currently consultant, Superior Design Co., Inc.

§Manager, Mechanical Design Methods Program, Engineering Mechanics Laboratory.

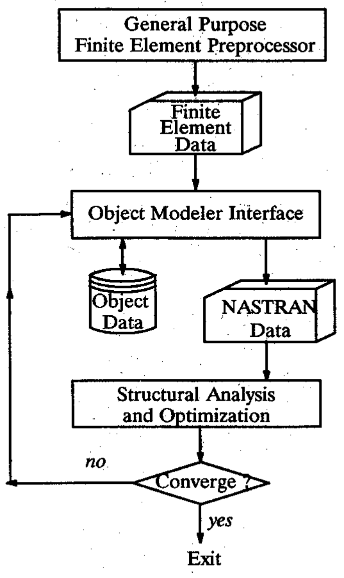

Fig. 1 System architecture for satellite structure optimization.

the details required for the analysis of satellite structures. Section IV discusses the sensitivity analysis procedures. Section V explains the approximation concepts used with MSC/NASTRAN to minimize the computational effort required. Section VI describes the use of object-oriented design and programming to support the satellite optimization process. Section VII describes the software system, or satellite optimization system (SAT-OPT), being developed at the General Electric Corporate Research and Development Center (GE-CRD) to address optimization of the satellite structural subsystem. Section VIII illustrates these optimization techniques on some realistic design problems. Section IX will describe how system optimization techniques can be used to further improve efficiency and achieve better designs. Finally, Section X provides the authors' concluding remarks.

II. Satellite Structural Optimization

The satellite structural optimization task can be stated as a nonlinear programming problem in the following form: Find the vector of design variables X that

$$\text{Minimizes:} \quad F(X) \quad \text{objective function} \tag{1}$$

Subject to:

$$g_j(X) \leq 0, \qquad j = 1, \text{ the number of equality constraints} \tag{2}$$

$$x_i^l \leq x_i \leq x_i^u, \; i = 1, \text{ the number of design variables} \tag{3}$$

In this work, the method of feasible directions algorithm, described in Ref. 1 and programmed in the numerical optimizer described in Ref. 2, is used for solving the problem defined above.

For satellite structural optimization, the design objective, Eq. (1), is to minimize the structural weight while satisfying the constraints imposed by Eqs. (2) and (3). Due to their high stiffness-to-weight ratios, honeycomb sandwich panels are used extensively in the design of satellite structures. However, great care must be exercised in the design optimization of these panels, because there are several failure modes possible with sandwich structures. To prevent any local instabilities, several different strength and buckling constraints are imposed during the iterative optimization process. These design constraint requirements are detailed in Sec. III. In addition to the strength and buckling constraints, it is also necessary to impose certain frequency constraints on the design model to avoid dynamic coupling either between two subsystem frequencies or between the satellite structure and launch vehicle frequencies.

The design variables used for optimization of satellite structures are primarily sizing-type parameters, including cross-sectional dimensions of struts and stiffeners, plate thicknesses, and honeycomb panel face sheet and core thicknesses.

Apart from the satellite's structural subsystem, optimization-based design of satellites involves several other subsystems, including solar array, reflector, antenna, and propulsion. The disciplines associated with the various satellite subsystems include structures (stiffness and strength), thermal, power, attitude controls, etc. Since the design optimization of such a complex engineering problem cannot be formulated and solved as one integrated unit, a decomposition-based optimization strategy, described in Ref. 3, can be adopted. In decomposition-based optimization, the satellite system is first broken down into several subsystems, such as primary structure, solar arrays, reflectors, etc., and then the coupling between the different subsystems is considered. A nonhierarchical decomposition method, described in Ref. 4, which does not restrict the interaction among the subsystems and which gives the designer more flexibility with decomposing the satellite system, is suggested here.

III. Strength and Local Buckling of Honeycomb Sandwich Panels

A honeycomb sandwich panel is a layered construction typically formed by bonding two thin face sheets to a thick core. The face sheets take the membrane and bending loads while the core resists the shear loads. Based on the type of loading, sandwich panel structures need to be designed to meet several different strength and local buckling requirements. These requirements, described in Ref. 5, include:

1) The face sheets must be sufficiently thick to withstand the tensile, compressive, and shear stresses.

2) The core must be designed with sufficient thickness and shear modulus to withstand the shear stresses from the loads and to prevent overall buckling of the sandwich panel.

3) The core cell size must be designed to prevent intercell buckling of the face sheets.

4) The compressive strength of the core must be adequate to prevent wrinkling of the face sheets.

For design optimization, the above strength and buckling requirements are imposed as constraints on the honeycomb panel margin of safety (MS) in the form

$$g_{\text{ms}}(X) = g_{\text{lb}} < \text{MS} < g_{\text{ub}}$$
$$\text{MS} = \frac{\sigma_{\text{allow}}}{\sigma_{\text{max}} * \text{FS}} - 1.0 \tag{4}$$

with the factor of safety (FS). Since a positive margin of safety is required, a lower bound of 0.01 is used. The maximum stress is the maximum absolute value of maximum principal stress, minimum principal stress, von-Mises stress, and maximum transverse shear stress, depending on the particular buckling failure mode of interest. The various stresses are obtained from a MSC/NASTRAN finite-element stress analysis, described in Ref. 6.

Intercell buckling, face sheet wrinkling, and core shear failure relations are required for evaluating the panel margins of safety. Constraints are also imposed on the compression, tension, and shear strengths of the material.

IV. Design Sensitivity Analysis

Design sensitivity analysis deals with the calculation of the gradients of the structural responses with respect to the problem parameters. References 7 and 8 describe the relevant theory and methods for design sensitivity analysis. The gradients are used both in the numerical optimization process for calculating the search directions in design space and for setting up the approximate optimization problem outlined in the following section. In this work, the semi-analytical sensitivity calculations in MSC/NASTRAN are used for the frequencies and stresses. The stress sensitivity coefficients are in turn used to analytically calculate the sensitivities of the honeycomb panel buckling modes. For completeness, the relevant calculations are outlined hereafter.

The sensitivities of the structure frequencies with respect to the design variables are computed as

$$\frac{\partial \lambda_j}{\partial x_i} = \frac{\phi_j^T \left[\frac{\partial K}{\partial x_i} - \lambda_j \frac{\partial M}{\partial x_i}\right] \phi_j}{\phi_j^T M \phi_j} \tag{5}$$

$$\frac{\partial f_j}{\partial x_i} = \frac{1}{4\pi\sqrt{\lambda_j}} \frac{\partial \lambda_j}{\partial x_i} \tag{6}$$

The sensitivities of the honeycomb panel buckling margins of safety are calculated by

$$\mathrm{MS}^k = \frac{\sigma^k_{\mathrm{allow}}}{\sigma^k_{\max} * \mathrm{FS}} - 1 > 0.0 \tag{7}$$

$$\frac{\partial \mathrm{MS}^K}{\partial x_i} = \frac{(\sigma^k_{\max} * \mathrm{FS})\left(\frac{\partial \sigma^k_{\mathrm{allow}}}{\partial x_i}\right) - (\sigma^k_{\mathrm{allow}} * \mathrm{FS})\left(\frac{\partial \sigma^k_{\max}}{\partial x_i}\right)}{(\sigma^k_{\max} * \mathrm{FS})^2} \tag{8}$$

where the superscript k denotes the particular failure mode (e.g., intercell buckling, face sheet wrinkling, shear crimping). The maximum stress gradients that are required for computing the above sensitivities are in turn obtained from a MSC/NASTRAN Solution 200 analysis, described in Ref. 9, and the gradients of the stress allowables with respect to the design parameters are obtained by analytically differentiating the empirical equations given in Bruhn of Ref. 5.

V. Approximate Analysis

A satellite structural optimization system (SAT-OPT) that accounts for weight, strength, and stiffness design requirements is currently being developed at General Electric's Corporate Research and Development Center. Structural optimization based on the philosophy of approximation concept, described in Refs. 10 and 11, has been implemented within SAT-OPT. In this approach, the original nonlinear programming problem, defined by Eqs. (1–3), is replaced by a sequence of approximate optimizations that involve a substantially reduced number of design constraints which are approximated as functions of the design variables. Because the approximate models essentially use first-order linearizations, their evaluation requires a substantially lower amount of computational effort than the exact model. The approximate optimization problem, as formulated in Ref. 12, is given by:

$$\text{Minimize:}\quad \hat{F}(X) = F(X_o) + \sum_{i=1}^{i=n} \left(\frac{\partial F}{\partial x_i}\right)_{x_o} (x_i - x_{oi}) \tag{9}$$

$$\text{Subject to:}\quad \hat{g}_j(X) = g_j(X_o) + \sum_{i=1}^{i=n} B_i \left(\frac{\partial g_j}{\partial x_i}\right)_{x_o} (x_i - x_{oi}) \tag{10}$$

where the x_{oi} are the initial design variables and where:

$$B_i = 1 \quad \text{if} \quad x_{oi} * \frac{\partial g_j}{\partial x_i} > 0$$

$$B_i = \frac{x_{oi}}{x_i} \quad \text{if} \quad x_{oi} * \frac{\partial g_j}{\partial x_i} \le 0 \tag{11}$$

and

$$x_i^l \le x_i \le x_i^u$$

VI. Object-Oriented Design

Object-oriented techniques are emerging as powerful new approaches for addressing problems associated with engineering design. This is due in part to reductions in software development and maintenance costs, but more importantly to the logical correspondence between computer objects and the physical problem being modeled. By organizing data in common classes, the software structure mimics the objects within a particular system. In the case of design optimization of satellite structures, several classes of objects can be easily identified. They include material data, physical properties, optimization variables and constraints, finite-element data, etc. Furthermore, to support multilevel and system optimization, object-oriented design plays a key role by providing the structure for subdivision of the model into its logical constituent subsystems.

The use of classes leads to the concept of abstraction, a critical aspect of an object-oriented system.Through abstraction, the basic requirements of an object are concentrated in a single program structure. Specialization is achieved through inheritance, that is, the definition of subclasses. Subclasses inherit both form (data structures) and function from the parent class. A subclass may add more data structures and functions and may also redefine inherited functions. This scheme allows high-level routines in a program to concentrate on the common aspects of the functionality and ignore the details of how the functionality must be implemented for each case. This approach also enables additional functionality to be added without disrupting the control routines by creating a new subclass that supports the protocol of the class, as well as providing the new features.

The following subsections provide an overview of the basic object-oriented concepts. Section VI.B will describe how these techniques can represent design optimization data and Sec.VI.C. illustrates how object-oriented classes can represent cross-sectional data for structural optimization problems. Section VI.D will describe how object-oriented design facilitates the development of system optimization software involving the optimization of interrelated subsystems. The implementation of these concepts was performed with C^{++}, so the vernacular used herein is slanted towards C^{++} terminology, described in Ref. 13.

A. Object-Oriented Programming

Rumbaugh et al.[14] characterize an object-oriented approach by four concepts: identity, classification, inheritance, and polymorphism. To illustrate how these four concepts can be applied, consider a class Material, with subclasses as shown in Fig. 2.

Identity means that the data is sectioned into discrete entities or objects. The objects are unique and distinguishable. Each material created will represent a different, unique material. The materials are unique even if they have similar properties. The implementation may associate a user name (in this case mat-name) to allow humans to easily distinguish between the materials.

Classification is the means by which objects are organized. Classes group together objects that logically belong together as a consequence of their properties or behavior. Classes provide the means whereby functions and data are related. Because all materials will need to print data and have a name (not to mention other

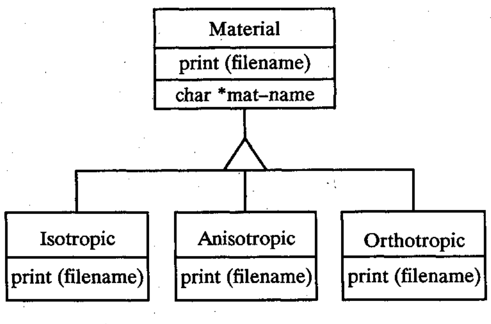

Fig. 2 Material class and subclass.

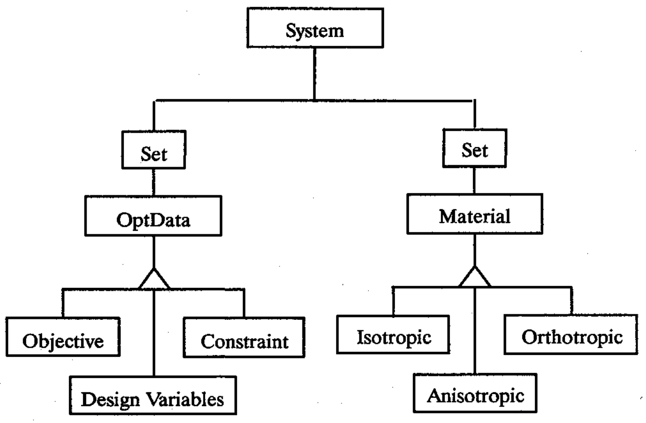

Fig. 3 Object model showing sets, classes, subclasses, and their relationship to the system.

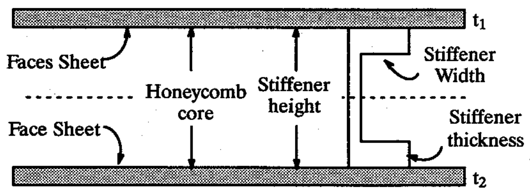

Fig. 4 Example of linked stiffener width with honeycomb core thickness.

functions), they logically form a class. In Fig. 2, the Material class contains a function to write out material properties and contains a string to represent the name of the material.

Inheritance allows subclasses to be created from parent classes. In particular, Isotropic and Orthotropic are two subclasses of the Material class, and inherit functions and data from the Material class, such as the print function and the mat-name character string. In this case, instances of materials should be classified according to one of the Material subclasses and not as a Material object. In this situation, Material is an abstract class. Inheritance is one of the main features of object-oriented implementations, because it allows code to be easily reused. With polymorphism, the program determines at run time the correct function to call. Subclasses can redefine functions inherited from the parent class to act differently. In this case, Isotropic materials will print different data than Orthotropic materials, and thus both subclasses redefine the print function. A function that simply printed the name of the material would not need to be reimplemented, because all materials can perform the function precisely the same. To print out the material properties for all the materials in a set, the program can take advantage of the polymorphic properties of the Material print function. The high-level routine to print the material properties contains a list of material objects. Each material object is a particular type (from its construction as an Isotropic or Orthotropic material). When the Material print function is called, then the object itself determines which print function to call. When the material is isotropic, the print function defined by the Isotropic class is executed; when the material is orthotropic, the print function defined by the Orthotropic class is executed. In C^{++}, classes specify which functions can be reimplemented by subclasses by identifying them as virtual functions.

When compared with traditional programming styles, polymorphism eliminates if tests or switches that can proliferate throughout a code. These tests are eliminated by associating the correct function with the object when it is constructed. An abstract layer, such as Material, identifies the protocol for subclasses through the set of virtual functions that are defined. For instance, if the default Material print function is not appropriate for a new subclass of anisotropic materials, the developer would have to implement the appropriate function. Furthermore, the developer will not need to search for all the if tests or switch statements to add the new case.

Virtual functions are one way to class definitions (in C^{++}) and control access to the functions associated with the structure. Additionally, data and functions can be identified as public or protected. Public data and functions can be accessed without any restrictions. Protected data on the other hand can only be accessed by other objects of the same class (or the object itself). This approach allows the implementation to isolate itself from how it is used and to protect the data from corruption. Typically, a class will protect the data and provide functions that retrieve the information.

Using these concepts, the structure of a problem is determined by identifying the classes in the domain and the operations to be performed on each class. The relationships between classes must also be identified, that is, which objects are used to define other objects (part-of) and which objects are types of other objects (subclasses), as described in Ref. 15. For instance, a satellite panel needs to know what material it is made out of, but it is not a type of material. So the material is a part of the panel definition. As discussed above, Isotropic is a type-of, or subclass of, Material.

B. Implementation of Optimization Data

Many aspects of a satellite optimization problem can be implemented using object-oriented programming techniques. The previous discussion used materials simply to illustrate the main concepts of object-oriented design. This section will focus on the specific implementation of optimization data.

There are three types of optimization-specific data with which we are concerned: the objective-function data, the design-constraint data, and the design-variable data. While these data do not have much in common, other than being required for optimization, they should be logically grouped together to form subclasses of a new class, OptData. An OptData object knows what type of OptData subclass it is, and the only data directly stored in the OptData class is an identifier. All other relevant data is stored in the appropriate subclass structure.

As with material properties, the optimization data is a part of the complete model description and therefore needs to be associated with the top level System object. One way of doing this is by grouping data together into Sets and associating the Sets with the System, as shown in Fig. 3. A Set is a standard object used in object-oriented systems. It is essentially a container, used to group collections of objects together.

C. Implementation of Cross Sections

For the optimization of the satellite structural subsystem described in Sec. VIII.B, some of the design variables are directly related to cross-sectional properties of truss and beam members. Figure 4 illustrates this by showing a slice through a honeycomb panel where a channel section is used as an edge stiffener, and which is modeled in NASTRAN as a beam element. If the panel core thickness is a design variable and it changes during the optimization process, then the height of the stiffener must also change, thereby requiring a modification of the cross-section properties. In addition, the specific dimensions of the channel may be design variables, and any changes in them will also require a recomputation of the section properties. The analysis code does not need to know the specific shape of the cross section, but merely the numerical values of the requisite properties.

A cross section class, Xsection was created, together with subclasses that define specific cross-sectional shapes, for example the channel or "C" shape in Fig. 4, or a circular tube cross section. The dimensions of the cross section are defined and stored in the subclass either as explicit values that do not change during the optimization process, or as references to design variables. These values are stored as protected members of the object, but public functions to retrieve each dimension are provided. These functions determine if the requested dimension is defined by the value of a design variable or explicitly.

The Xsection class includes virtual functions to retrieve cross-sectional properties, such as area and moment of inertias. Each subclass, or cross-section type, identifies the function to calculate values correctly for the specified shape using the subclass functions to retrieve the appropriate dimension values, as shown in Fig. 5. The driver routine only needs to recognize the Xsection class and its functions. When a new cross-section shape is added, the only additional coding required is implementation of a new subclass, along with the required virtual and dimension functions. The driver routine requires no modifications.

D. Hierarchic System Representation

A major requirement imposed on the design of the data structures and functions to support system optimization is the ability to generate analysis models for the entire system and for individual subsystems while minimizing the effort required by the user to decompose the model. A hierarchical design of the system is shown in Fig. 6. A system is comprised of a Set of Subsystems and a Set of OptData, whereas Subsystems are comprised of a Set of Components and a Set of OptData. By developing such a representation, it can be applied to different scenarios for satellite design depending on the application. One scenario is where the System object is the overall satellite itself and is decomposed into a number of distinct Subsystem objects such as the structural, thermal control, propulsion, and electrical subsystems. As a minimum, interdisciplinary optimization is implied. Another scenario would be where the System object corresponds to the satellite's structural subsystem and the Subsystem object corresponds to logical structural subassem-

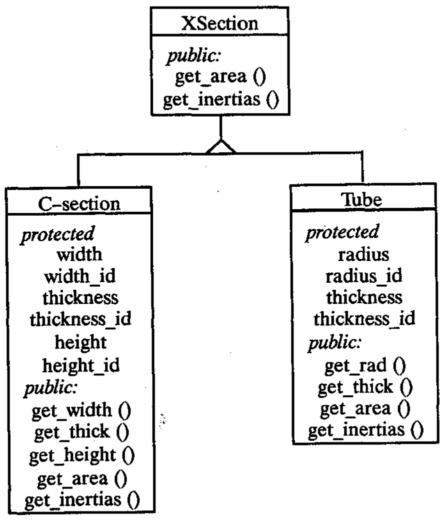

Fig. 5 Example of Xsection object structure.

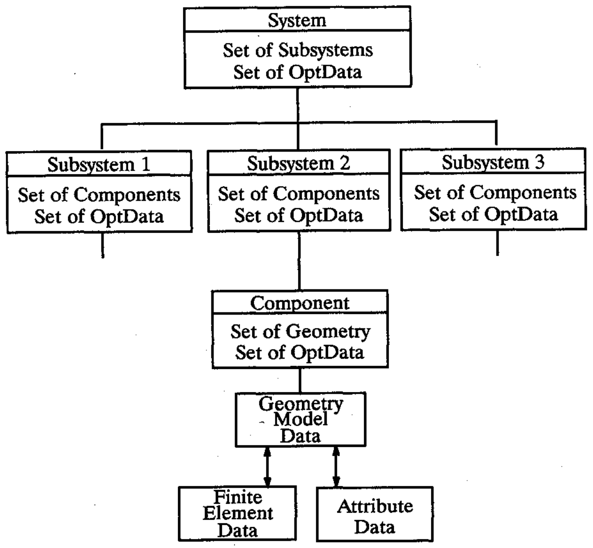

Fig. 6 Object-oriented system modeling.

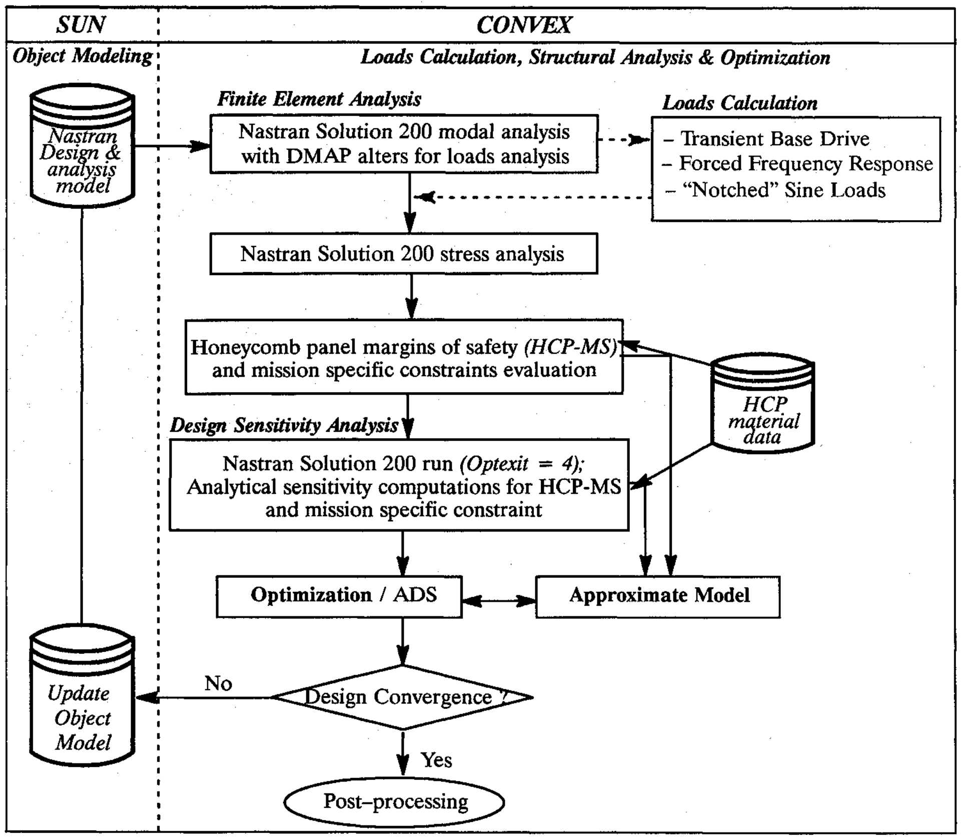
SUN
CONVEX
Object Modeling
Loads Calculation, Structural Analysis & Optimization
Nastran Design & analysis model
Finite Element Analysis
Nastran Solution 200 modal analysis with DMAP alters for loads analysis
Loads Calculation
- Transient Base Drive
- Forced Frequency Response
- "Notched" Sine Loads
Nastran Solution 200 stress analysis
Honeycomb panel margins of safety (HCP-MS) and mission specific constraints evaluation
HCP material data
Design Sensitivity Analysis
Nastran Solution 200 run (Optexit = 4); Analytical sensitivity computations for HCP-MS and mission specific constraint
Optimization / ADS
Approximate Model
No
Design Convergence ?
Yes
Update Object Model
Post-processing

Fig. 7 SAT-OPT system architecture.

blies that support the satellite's payload, propulsion system, antenna, etc. For the purposes of this paper the later scenario is of interest, and thus Component objects are comprised of geometry data and related finite-element and analysis-specific attribute data. Such a system representation allows for the development of multilevel, multi-objective optimization, where hierarchic and nonhierarchic approaches can be employed.

VII. Implementation with MSC/NASTRAN

The satellite optimization system, SAT-OPT, that solves the structural synthesis problem is shown in Fig. 7 and uses the methodology described previously. MSC/NASTRAN's solution sequence 200 is used for finite-element static and modal analysis. The flight loads for which the satellite structure is designed are computed using transient base excitations and harmonic analysis. Reference 16 describes the procedures involved in flight-loads calculation. The stress responses computed from the finite-element analysis are then used to compute the honeycomb panel failure margins of safety. The sensitivities of the structure frequencies and panel forces and stresses are computed using the solution 200 capability. The panel force and stress sensitivities are then used for analytically computing the sensitivities of the panel failure margins of safety. The approximate optimization problem, defined by Eqs. (9–11), is then solved using ADS, and the whole process is repeated until a convergence is achieved.

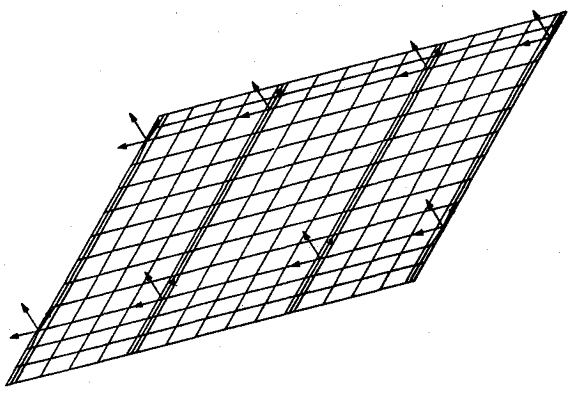
Fig. 8 Honeycomb panel with doublers.

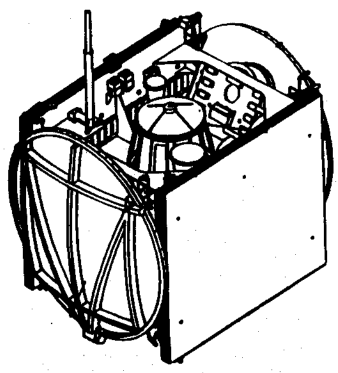
Fig. 9 Satellite structural subsystem.

VIII. Design Examples

A. Honeycomb Panel with Doublers

A honeycomb panel with doublers, shown in Fig. 8, was optimized for frequency and local buckling requirements. Both the base panel and the doublers were modeled with aluminum honeycomb. The local use of doublers in heavily loaded regions of the panel improves the quality of the part by avoiding the use of heavier face sheets over the entire structure.

The design optimization problem was to minimize the weight of the structure subject to a minimum fundamental frequency requirement of 10 Hz and positive buckling margins of safety. The design variables were the honeycomb face sheet and core thicknesses. The initial design had a fundamental frequency of 10.16 Hz with an overall weight of 58.664 lbs. This weight included a nonstructural mass of 7.72e-3 lb/unit area on the base panel and doublers. However, the local buckling criterion of the doubler, corresponding to the core shear through-thickness direction, was violated with a margin of safety value of -0.33. After four design iterations, a feasible design with an overall weight of 67.9547 lbs was obtained. The increase in weight is attributed to overcoming the infeasible starting design. The core shear margin of safety, corresponding to the final design, was + 0.04.

B. Satellite Structural Subsystem

The structural subsystem of a generic satellite, shown in Fig. 9, is the second design example. The structural subsystem primarily consists of the bus core and payload modules. The bus core module is composed of the spacecraft primary structure (cylinder/conical adapter assembly and bulkheads that join the cylinder to the surrounding panels). The payload module is comprised of the tran-

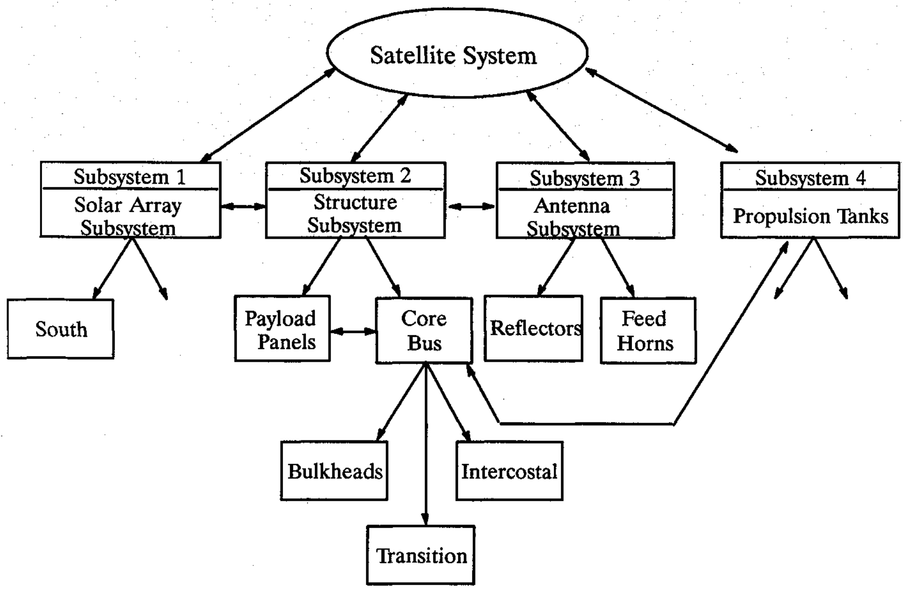

Fig. 10 Representative satellite decomposition with cross coupling.

sponder panels, the Earth panel, and other small panels that provide structural interfaces for the antennae. Certain panels have heat pipes embedded in the honeycomb core to help distribute the heat and radiate the heat out into space. Some of the functional requirements of the structural subsystem include providing structural stiffess in accordance with dynamic analysis requirements, accommodating all spacecraft components, and supporting the solar array panels, reflectors, and antennas. The design of the complete structural subsystem for strength/local buckling and dynamic requirements is a complex task, and therefore, structured design methods are essential for rapid evaluation of configuration changes and for achieving near-optimal designs.

The design optimization problem was to minimize the structural weight subject to constraints on frequencies, for panel buckling margins of safety to be positive, and for the stress in the trusses not to lead to a violation of the Euler buckling allowables. In accordance with launch vehicle specifications, the frequency requirements were a 12-Hz minimum bending frequency and a 31-Hz minimum axial frequency. The design model included a total of 42 design variables representing honeycomb core and face-sheet thicknesses, truss cross-sectional properties, and cylinder thickness. The high number of design variables provides the designer with more control to fine tune the structure, provided one can determine how to select the design variables. The initial design had a structural weight of 416 lb, with the frequency requirements satisfied and with violated (or negative) buckling margins of safety in several of the honeycomb panels. After three design iterations, a feasible design, satisfying all frequency, strength, and buckling requirements, and weighing 4 lb less than the initial design was obtained.

IX. Multi-Level Optimization

The design examples presented in the previous section were limited to single-level optimization of subassemblies within the satellite's structural subsystem. Although the structure is the first subsystem to be designed for a satellite, as mentioned in Sec. II, the optimization based design of a complete satellite system involves other subsystems such as solar array, antenna, propulsion, etc. Since a distinct hierarchy is not present, and because of the lateral coupling that exists between various subsystems, a nonhierarchical decomposition approach, as described in Ref. 4, with necessary variations for satellite design, is recommended here. An example of the satellite system decomposition is shown in Fig. 10. The lateral coupling between the structure's subsystems and the other subsystems is primarily in the form of interface loads, dynamic coupling, subsystem alignment and pointing, etc. Clearly, each of these individual subsystem designs would lend themselves to a hierarchic, decomposition-based optimization, whereas the overall satellite system optimization would necessarily be based on nonhierarchic decomposition.

X. Concluding Remarks

Of strategic importance today is the design of high-performance satellites, where weight, cost, and time-to-market are the primary drivers. Typically, a satellite's structural subsystem is geometrically simple, but is comprised of complex material systems, where stringent, often conflicting (e.g., strength vs. frequency vs. buckling), design requirements are imposed. In addition, the number of design variables can be very large, and thus, the ability to rapidly produce optimum, or even near-optimum designs with conventional analytical tools is a major challenge. This problem is exacerbated by the need to use large, complex, finite-element models to accurately model the physics. To address these problems, an automated analysis system, aimed at the optimum design of satellite structural subsystems, is being developed. This system, known as SAT-OPT, provides a rigorous, cost-effective approach to achieving optimum designs of spacecraft structures, which are predominantly composed of honeycomb sandwich panels. SAT-OPT employs numerical optimization procedures, leveraging the design model capabilities that currently exist within NASTRAN and exploiting the notion of approximate analysis to achieve substantial computational benefits. Also, this system uses object-oriented techniques to simplify the representation and control of the iterative design process.

The system was demonstrated on two satellite design problems where weight minimization was the objective, with constraints on frequency, strength, and buckling. In both cases, SAT-OPT reduced the manually derived optimum weight and successfully met the multiple requirements. It is expected that further gains in computational efficiency can be achieved when the current system is extended to include a multilevel optimization capability. This extension will be greatly facilitated by the object-oriented foundation that currently exists.

In summary, the possibility of achieving a design that efficiently meets multiple requirements, together with the difficulty of selecting the values of a large set of design variables, makes structural optimization an obvious and useful technology for the design of complex satellite systems.

References

[1]Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design: with Applications,* McGraw-Hill, New York, 1984.

[2]Vanderplaats, G. N., *ADS User's Manual,* VMA Engineering, Goleta, CA, March 1988.

[3]Sobieski, J., "A Linear Decomposition Method for Optimization Problems—Blueprint for Development," NASA TM-83248, Feb. 1982.

[4]Sobieski, J., "Optimization by Decomposition: A Step from Hierarchic to Non-Hierachic Systems," *Recent Advances in Multidisciplinary Analysis and Optimization,* NASA CP-3031, Sept. 1988.

[5]Bruhn, E. F., *Analysis and Design of Flight Vehicle Structures,* Jacobs Publishing, 1973.

[6]Anon., *MSC/NASTRAN User's Manual,* Version 67, The MacNeal-Schwendler Corp., Los Angeles, CA, 1991.

[7]Haug, E. J., Choi, K. K., and Komkov, V., *Design Sensitivity Analysis of Structural Systems,* Academic Press, New York, 1986.

[8]Kodiyalam, S., "Sensitivity Analysis in Static and Dynamic Problems," *Advanced Techniques in Structural Optimization,* edited by S. Hernandez, Computational Mechanics Publications, Southampton, UK, 1993, pp. 33–53.

[9]*MSC/NASTRAN Design Sensitivity and Optimization Manual,* Version 67, The MacNeal-Schwendler Corp., Los Angeles, CA, 1992.

[10]Schmit, L. A., and Miura, H. "Approximation Concepts for Efficient Structural Synthesis," NASA CR-2552, March 1976.

[11]Miura, H., Vanderplaats, G. N., and Kodiyalam, S., "Experiences in Large Scale Structural Design Optimization," *Applications of Supercomputers in Engineering: Fluid Flow and Stress Analysis Applications,* edited by C. A. Brebbia and A. Peters, Elsevier Science Publishers, B. V., The Netherlands, Sept. 1989, pp. 251–268.

[12]Starnes, J. H., and Haftka, R. T., "Preliminary Design of Composite Wings for Buckling, Stress, and Displacement Constraints, *Journal of Aircraft,* Vol. 16, No. 8, 1979, pp. 564–570.

[13]Ellis, M. A., and Stroustrup, B., *The Annotated C^{++} Reference Manual,* Addison-Wesley, New York, 1990.

[14]Rumbaugh, J., Blasha, M., Premerlani, W., Eddy, F., and Lorensen, W., Object-Oriented Modeling and Design, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[15]Booch, G., *Object-Oriented Design with Applications,* Benjamin Cummings Publishing, New York, 1991.

[16]Kodiyalam, S., "Automation of Transient Flight Loads Estimation for Spacecraft Design," DE-Vol. 65-1, *Advances in Design Automation,* ASME, 1993, pp. 245–255.

Earl A. Thornton
Associate Editor