1 Introduction

Modern design of large complex systems requires cooperation among many agents (humans or computers) that may be physically separated and/or operating under diverse environments (e.g., cultural or computational). Global product development requires conceiving and implementing a distributed cooperative system design capability. This is both a managerial and a technical challenge. Here we address some key technical characteristics of such a system motivated by automotive system applications. The major elements of the design capability are: (i) an appropriate computational infrastructure that uses an object-oriented environment; (ii) a system synthesis process that uses mathematical optimization with formal model-based decomposition and coordination strategies; and (iii) a common object database that uses voxels to represent geometric and material data.

2 Distributed Cooperative Computation

Current design and computational environments typically treat simulation or analysis software (SAS) as stand-alone tools, conceived for component or subsystem analysis and executed on a single computing platform. Integration with other commercial-off-the-shelf software is hardwired by function calls or parsing and writing text files. Automated synthesis tools inherit the limitations of the underlying SAS they use to estimate the design criteria and constraints. System synthesis requires integration of several subsystem SAS. Moreover, even at the component level, SAS are typically designed for a single iteration instead of several consecutive ones needed in an optimization study. For example, model building or initialization of all design parameters is not needed after the first optimization iteration, and its obligatory execution in subsequent iterations is a wasted, substantial, computational burden.

Software interoperability in the form of object-oriented paradigms and protocols have so far been used primarily in business processes integration [Goldstein, 1996]. An object-oriented distributed design environment is eminently suitable for engineering systems simulation and synthesis. Its desirable characteristics are as follows:

- Platform independence: support distributed, heterogeneous and reconfigurable computing resources.
Multiple levels of model abstraction and fidelity: support distributed optimization algorithms with model hierarchies required in decomposition and coordination strategies.

Reconfigurability of the design problem: "plug and play" of models, databases, SAS and synthesis tools to allow use of progressively more complex analytical models.

Accommodation of both custom-developed and legacy (existing) codes: this is critical for practical, commercial environments.

Synchronized simulation: e.g., in a realistic powertrain system simulation, a FORTRAN simulation of the engine must be synchronous to a MATLAB/SIMULINK simulation of the driveline and to a multibody dynamics simulation of the suspension and vehicle.

Adequate time performance for design purposes: although real time calculation is not necessary for design, the computational time (including model preparation) should not add significantly to design cycle time; also, speed-ups from distributed computing should not be overshadowed by network communication overheads.

Security of operation across untrusted networks: analysis tools should be developed and maintained at different locations while still being used to solve a system problem.

The Common Object Request Broker Architecture (CORBA) is an appropriate medium for building a distributed cooperative design environment [OMG, 1995]. CORBA is an industry standard for distributed, heterogeneous, object-oriented applications, which is open, robust, interoperable, multiplatform and multivendor supported. It uses an Interface Definition Language (IDL) to describe the data types and operations that a server provides for the implementation of a given object. In our context, examples of objects are models of subsystems or design optimization "search engines." The Internet Inter-ORB Protocol (IIOP) enables the interconnection of large distributed applications across the Internet. Legacy SAS can be integrated into a CORBA environment by stripping off any I/O from the source code and creating a library of SAS functions that can be called from C or C++ code. A CORBA IDL interface enables entering data into the external code, executing defined functions and retrieving data. Commercial software can also be wrapped with a CORBA interface.

Figure 1 shows the architecture of the design environment with the CORBA standard and the Voxel representation (see Section 4 below) at its core. Objects having functionality inherited from simulation and analysis codes are distributed in a multiplatform environment. Synthesis and control objects provide the capabilities for design optimization and synchronized simulation, respectively. The architecture also allows remote access on an Intranet or Internet by means of Java client applets that communicate with the CORBA-based objects.
3 Synthesis Based on Decomposition and Coordination

Given an existence of mathematical models, the preferred synthesis process is a search engine driven by goals. Gradient-based optimization methods have been successful for relatively small, well-behaved, single-discipline models. Evolutionary or stochastic methods are suitable for computationally very inexpensive models. Multidisciplinary design optimization (MDO) practitioners face both lack of robustness and of interoperability of SAS and optimization tools.

Multilevel optimization takes advantage of a distributed system model by solving smaller optimization problems at the subsystem and system levels. Same level subproblems may be solved in parallel, using the optimization technique most suitable for the underlying model, gaining in robustness, speed and engineering interpretation of results. Rapid reconfigurability of analysis and simulations enables dynamic redefinition of subsystem and system level subproblems throughout the optimization process. Low fidelity models (e.g., surrogate models) may be used for the first iterations whereas high fidelity models only to refine the search. For hybrid optimization approaches, low fidelity models are more adequate for the global search stage (using, e.g., evolutionary, pattern search or simulated annealing algorithms), whereas high fidelity models work better during the local search stage (using a nonlinear programming algorithm). An object-oriented design environment facilitates such implementations [Eldred et al, 1996]. Moreover, plug-and-play capability allows upgrading models and optimization algorithms without reintegration of the software system.

Distributed cooperative design requires decomposition of the original design problem and coordinated solution of the resulting subproblems. Although engineering insight may help in recognizing clusters of subsystem models for multilevel distributed optimization, the decision may be difficult for a large unprecedented system. System optimization cost critically depends on how loosely the individual subproblems are coupled by coupling or coordinating variables. Moreover, model cluster identification should consider the availability and throughput of the computational resources and the cost of performing the individual analyses [Michelena and Papalambros, 1996, 1997].

Figure 2 shows a formal methodology for distributed design optimization. Model partitioning is formulated as a higher level optimization problem and solved using graph partitioning or integer programming algorithms. Partitioning constraints are specified to account for physical system requirements, computational resources and simulation costs. The model structure is extracted by partitioning, and a multilevel optimization model is defined in conjunction with a selected coordination strategy. A coordination strategy is needed to make sure that the solution generated by subproblem optimizations fulfills the system level optimality criteria of the original design problem. Convergence properties of the coordination strategy are critical for successful distributed optimization.
[Nelson and Papalambros, 1997].

The functional dependence table associated with the system models assigns a row to each design relation and a column to each design or state/behavior variable. The term design relation is very general and may correspond to either an algebraic or differential equation, a discretized continuum equation, a response surface, or a simulation used to evaluate state/behavior characteristics or constraint functions. A black-box simulation may be treated as a single design relation or as a collection of relations, each one corresponding to an output from the simulation. The system model corresponds to an optimal design problem or to a general design problem depending on whether or not design objectives have been selected from design requirements (represented by constraints in the design model), respectively. The process allows for the proper selection of objective functions so that a partitioned model can be created suitable for coordinated solution [Krishnamachari and Papalambros, 1996]. Figure 3 shows an example of a functional dependence table for a vehicle powertrain system model after optimal model-based partitioning has been applied [Michelena and Papalambros, 1995].

4 Unified Geometry/Material Databases: Voxel Digital Representation

The systems design strategy described so far is usually left as an unrealizable conceptual “framework” in mechanical design. Geometry and material properties make such systems tightly coupled. Moreover, building analysis models for complex 3D geometries and translating equivalent geometries across different analyses (e.g., structural FEM vs. CFD) is very time consuming—often rendering design iterations impractical. The recently developed voxel-based digital representation is a promising technology to address these issues, making model-based design iterations attractively fast [Voxelcon, 1997].

The Voxel Conversion (Voxelcon) technology uses voxel representations of 3D objects. The representations can be created from CAD surfaces or taken directly from techniques like Computed Tomography (CT) Scanning. The Voxelcon interface allows the user to go from the voxel representation into a number of different engineering modalities, including CAD, Rapid Prototyping, Finite Element Analysis, and Topology Optimization. Conversions can be done in minutes in con-
trast to days or weeks needed for conventional technology. Large scale analysis of voxel-based finite element meshes containing up to 1 million 3D elements can be performed on engineering workstations.

To illustrate the voxel approach, consider the example of an automotive pump scanned on a 450 KV CT scanner. Two slices from the scan are shown on Figure 4. The scan voxel data are read, re-sampled and directly converted into a large scale 3D FE mesh containing nearly 60,000 elements. The total time to generate the mesh from the CT data was about 5 minutes on a workstation. Note that including the scanning effort and depending on the available equipment, a process that would take several weeks is compressed to hours or days.

Following application of boundary conditions, the mesh was analyzed using the large scale Voxelcon finite element module Vox3d. The resulting Von Mises stress distribution is shown on Figure 5. The finite element analysis method employed has been specially developed for use with voxel representations and is very fast. In particular, if the design goal is to study topological changes, methods for topology optimization with full finite element analysis can be used realistically as no remeshing is necessary. In fact, since the number of elements that can be used is very large while the computational time on a workstation is relatively short, topology can be studied in such detail that complete shape description can be also achieved during one optimization run.

5 Concluding Remarks

The distributed computational environment described above is already widely used in business applications. The main ingredients described above should allow the same wide use occur in product development. The formal ability to synthesize systems that can be efficiently coordinated
and the availability of new voxel data manipulation can deal effectively with the major technical impediments of the past. The organizational management issues remain daunting. However, speed, accuracy and complexity of model-based system design will improve dramatically, and the internet paradigm will force truly distributed product design to become commonplace. Organizational structures will have to adapt or risk obsolescence.

Acknowledgment

This research was partially supported by the Automotive Research Center at the University of Michigan. The Voxelcon data were provided courtesy of Dr. Scott Hollister and Psiphics Technologies, Inc.

References


Authors

Prof. Panos Y. Papalambros (pyp@umich.edu)
Dr. Nestor F. Michelena (nestorm@umich.edu)
Prof. Noboru Kikuchi (kikuchi@engin.umich.edu)

Department of Mechanical Engineering and Applied Mechanics and Automotive Research Center, College of Engineering
The University of Michigan, Ann Arbor, Michigan 48109-2125, USA
phone: (313) 764-8464, fax: (313) 647-3170