CFD-Based Shape Optimisation with Grid-Enabled Design Search Toolkits

Wenbin Song, Andy Keane, and Simon Cox

e-Science Centre, School of Engineering Sciences, University of Southampton,
Highfield, SO17 1BJ, UK
{w.song, ajk, sjc}@soton.ac.uk,

Abstract

This paper presents an application of applying Grid computing technologies in the field of engineering design optimisation using computational fluid dynamics (CFD). Three essential elements in CFD-based shape optimisation problems (CAD, mesh generation, and solution) are integrated and automated within the Matlab scripting environment augmented with Grid-enabled computation and database toolkits in the form of Matlab functions. The toolkits allow easy access to remote computational resources and data archive capabilities. A design search and optimisation package is exposed to Matlab users in various ways and applied to an engine nacelle shape optimisation problem. A response surface model is constructed and searches conducted on it reveal the effect of negative scarf angle on the aerodynamic performance of the nacelle.

1. Introduction

Applying optimisation techniques in engineering design processes can improve the performance and quality of existing products and potentially lead to novel designs, which can be crucial in maintaining competitiveness in world markets. However, engineering design optimisation can be very time-consuming when high-fidelity analysis models are involved, for example, in CFD-based shape design for complex geometries such as wing-body configurations [1]. Delivering a better design within a restricted time scale is a challenge for engineers due to a number of reasons: (1) lack of scalable and flexible problem solving environments, where integration and automation can be achieved dynamically with minimum effort; (2) efficient data management approaches and tools that allows easy re-use of previously explored models and past experience, and (3) knowledge support for the use of various optimisation techniques and strategies. Increases in model complexity arising from the increased popularity of multidisciplinary approaches to design present further challenges to the current intranet-based infrastructure typically used in major engineering firms. A multidisciplinary design process is essentially a concurrent, distributed task, often with strict time scale requirements and potentially non-uniform demands for computing resources. Although the amount of data generated is not comparable to those involved in some major science projects such as the DataGrid project funded by the European Union [2], the data formats can be quite different ranging from plain text, to proprietary binary formats. Moreover, policy around the use of different data can vary significantly. For example, costly and potentially non-reproducible wind tunnel test data may need to be archived for a very long time, while analysis models can change from one product model to another. And not just the data need to be properly archived; the software tools that process the data, and archive tools also need to be properly managed.

The complexity and intensive data and knowledge requirements of typical design optimisation processes prompted the research and development of dedicated problem solving environments for design, such as iSIGHIT [3] and ModelCenter [4]. However, these frameworks are still largely built on existing company intranet infrastructures and lack the scalability and flexibility required for dynamic composition of large scale, possibly long-lived application models built from different software packages. These are typically required for complicated optimisation tasks such as those seen in the domain of multidisciplinary design across the physical boundaries of multiple existing establishments. Another issue for the approach adopted in these commercial packages is the use of proprietary standards, protocols, etc., which prevent generic plug-and-play capabilities from being developed: a new interface would have to be written for each new component added for the framework in place.

Optimisation technology has been the focus of research since the advent of digital computers. As a result of this, a large number of software codes, both commercial and in-house, are available today. OPTIONS [5] is one such design exploration system with more than 30 different search algorithms. Although optimisation technology itself is still an active research field in terms of the development of new algorithms, it seems that applications research has attracted most recent attention in the engineering community due to the apparent benefit in improving product design. Decisions on how to apply various algorithms to a wide range of problems requires knowledge and experience in both the algorithms and problems. One potential research area is to bring knowledge technologies into the optimisation framework to effectively characterise, classify, match, and hybridize existing algorithms to improve the overall robustness and efficiency of search.

This paper discusses the use of grid technologies and optimisation techniques for the effective use of search methods for CFD-based optimisation problems, in an effort to make this approach more accessible to designers on a daily basis. With the requirements of flexibility and scalability in mind, the widely used scripting environment Matlab [6] is chosen as the hosting environment for building the toolkits. Two sets of low-level computation and database toolkits have been built to provide users with access to grid-based computing resources and data archiving facilities from within Matlab environment. From version 6.5, support for Java from within Matlab provides the just-in-time acceleration necessary to take advantage of the Java CoG tools [7] to leverage the current development of Grid middleware [8]. The use of the Java platform also allows the computation toolkits to be easily migrated to other scripting environments supporting Java such as Jython and also to be integrated into any GUI development using Java.

This paper is organised as follows. The second section describes briefly Grid computing technologies and the toolkits developed by the Grid Enabled Optimisation and Design Search for Engineering (GEODISE) Project [9]. It is followed by section 3 which defines the problem of engine nacelle optimisation. Section 4 presents the optimisation methods used along with some results. Section 5 summarises the main conclusions and future work.

2. Grid Computing and the Geodise Toolkits

Grid computing technologies aim to bring together large numbers of vastly distributed processing and storage capabilities across multiple control domains to solve particular problems in a secure, coordinated manner. Issues particularly relevant to the engineering community include remote access to resources, problem solving environments with Grid access, management of potentially long-lived state, scheduling of parallel jobs, and efficient data management approaches, etc. Some other issues are of concern to the wider community, such as security policy, host environments, etc.

Two Geodise toolkits have been developed by the Geodise team in the last 18 months, addressing aspects of computing and data archiving, respectively. Implementation details can be found in a number of publications [10-12].

Two different mechanisms are used to submit computational jobs to computing servers. The first one uses a web service interface to Condor [13] to submit ProEngineer™ [14] jobs to a cluster of Windows machines to generate geometry [12]. The ClassAd mechanism provided in Condor is therefore used to match computational requests to available Windows resources with ProEngineer installed. The second mechanism is implemented via the use of Globus [8] and Java CoG tools [15], which enables computational jobs to be submitted to computing servers with a valid proxy certificate presented [10]. The first mechanism does not require that users know in advance which resources have ProEngineer installed, the second does.

Security is also addressed in these toolkits using the Grid Security Infrastructure implementation in Globus Toolkit 2.2.4.

3. Problem Definition

Three essential elements are normally required in CFD-based shape optimisation problems: these are geometry design using computer-aided design (CAD), mesh generation and solution. These three elements are loosely coupled within the Matlab scripting environment. The STEP data exchange standard is used to exchange geometry data between the CAD and mesh generation packages. Inter-operability issues are addressed at both sides of the geometry modelling and mesh generation. A parametric geometry model suitable for CFD analysis is first generated using the parametric CAD package ProEngineer. The subsequent mesh is generated with minimum reference to the geometry information, as only the top-level entities in the CAD model (faces for the three dimensional nacelle model used here) are referenced in the meshing script executed by the mesh generation tool Gambit. This is also consistent with the fact that a top-down approach is often adopted in geometry modelling, and there is no direct control over the lower topology of the geometry entities in the CAD modelling process.

Mesh generation has long been recognized as a bottleneck to a fully automatic and robust CFD process, especially for complex CAD surfaces [16]. Two approaches are typically used for the mesh tools, either direct access is provided to CAD parts and assemblies in their native mode, or third-party standard exchange data formats are used. Although the first approach may offer better performance and robustness, a particular mesh tool might not support the CAD package in use, and also the CAD data may be interpreted differently in different mesh tools.

It is crucial for a CFD-ready mesh that nodes on disjoint edges in close proximity have matching spacing-scales, and triangles on adjacent surfaces have matching length-scales. In most systems, it remains the users' role to identify such situations and enforce length-scale compatibility between nearby entities, although, a limited number of facilities are sometines available to implement such capabilities. By partly recovering the topological information pertinent to the original geometry data, consistency conflicts on the boundaries between entities can be avoided.

The edge mesh is first generated by specifying the appropriate node spacing based on both the relative length of the edge being meshed in the model and the relative lengths of adjacent edges connected to the vertexes of the edge.

1. The relative length of the edge in the whole model is used to determine a nominal edge node spacing for the edge based on the relations described in Table 1;

2. The minimum of all the nominal node spacings of all edges connected to the common vertex is applied to all edges connected to the vertex;

3. The mesh on the edge is then defined as non-symmetric with the node spacing in the middle approaching the nominal node spacing of the edge.

The surface mesh and volume mesh are then generated by applying size functions around vertexes of the surface and volume.

The flow solver Fluent [17] is then used to solve the flow problem. The Fluent model used in this work is a subsonic laminar model with operation parameters set to the cruise conditions. The nacelle model is placed in a pressure far-field modelled using a rectangular box.

A three dimensional engine nacelle optimisation problem is presented here. This problem is both complex and has a high computational cost. The goal is to reduce the ground noise generated by the fan when the aircraft takes off. The main geometry feature being examined is the degree of negative scarf. However, the noise level is not calculated at this stage, it is simply assumed that the higher the negative scarf, the lower the ground noise. This is illustrated in Figure 1. The focus therefore is how the aerodynamic performance of the nacelle is affected by the variation in scarf angle. It is believed that the adverse effect of scarf angle on aerodynamic characteristics such as flow separation will impose a limit on the maximum negative scarf angle possible. An unstructured mesh is generated using the procedures described above, and the process is fairly robust for wide ranges of geometries with scarf angles varying from –10 to 25 degrees. A typical unstructured mesh is shown in Figure 2.

Metadata-assisted data models

As mentioned earlier, various types of data files are involved in the optimisation process, some are in proprietary binary format, others are in plain text format. To organise the data in a logical fashion and also to help data re-use, metadata is extensively used to define the model and in data archive. Metadata is attached to standard files and Matlab structures, which enables queries to be made without opening the files. This feature can be helpful in locating large binary files in proprietary formats. Another useful feature is the logical grouping of data within the repository. This allows the establishment of a hierarchical structure which effectively organises the data in the repository into a tree structure, similar to a typical file system.

Through the use of the database toolkit [11], a tree structure of design and analysis data is constructed to define the geometry model, mesh model and solver model, as illustrated in Figure 3. These models are essentially the files needed for use with the relevant packages with a set of metadata attached to them. The use of metadata not only allows the data to be queried based on descriptive information without actually opening the files, but also provides a logical abstraction and grouping for the model files, which are conventionally scattered in the file system.

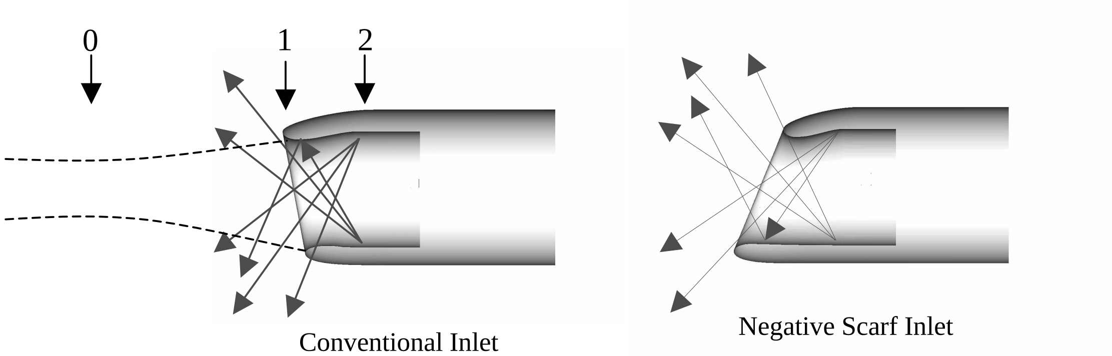
Interfaces to search engines

A design exploration system OPTIONS [5] is used as the search engine in this work. Two different mechanisms are used in exposing the functionalities of OPTIONS to Matlab users, as illustrated in Figure 4. The first approach uses the Matlab/Fortran external interface to expose search engines in the Matlab environment, it provides an easy-to-use interface familiar to Matlab users. The second approach, based on the Java Native Interface (JNI) [18], exposes a large number of OPTIONS APIs to Matlab users, thus provides more flexibility for implementing complicated search strategies such as hybrid Generic Algorithms (GAs) and local gradient descent methods. It also separates the data from the algorithms, thus enables the state of the optimisation process to be easily archived. A possible third route is also identified in the figure, which leverages the Microsoft .NET hosting environment.

Starting from the Matlab functions provided by the toolkits, a number of higher level Matlab functions are developed for optimisation tasks based on the data archive structure and interfaces to the search engines. These functions are intended to further reduce the work required for the user to carry out CFD-based shape optimisation tasks and can be used as components to compose more complicated workflows in implementations of workflow construction environment GUI.

A typical pseudo-script for the optimisation task is listed as follows:

```
...
import
org.geodise.optimisation.*;
optdat=JavaOptData;
options=JavaOptions;
optdat.omethod = 0.0;
optdat.niters = 1;
options.Load(optdat);
options.doepnts(optdat);
nvrs=optdat.nvars;
for i=1:nvrs:length(optdat.dwork)
   input=optdat.dwork(i:i+nvrs-
1);
   gd_cfdanalysis(…);
end
options.optfill(vars, cons, objs)
optdat.objmod=4.1;
optdat.update=0;
options.optrss(optdat);
optdat.niters = 400;
optdat.omethod = 4.0;
options.Search(optdat);
gd_cfdone(…);
...
```



Total Pressure Recovery (TPR) = pt_2 / pt_1

Figure 1. Illustrative effect of negative scarf angle on ground noise

Table 1 Relations determining the edge node spacing based on geometric information

Steps	Relations	Control parameters
1	edge_node_spacing_ref, specified by the user directly or by shortest_edge/alpha0	edge_node_spacing or alpha0
2	nominal_edge_node_spacing = alpha1*(edge_node_spacing_ref)* pow($\log_2$(edge_length/edge_node_spacing),alpha2)	alpha1, alpha2
3	minimum edge node spacing of all edges connected to a common vertex	
4	edge mesh using non-symmetric scheme	
5	surface mesh	
6	volume mesh	

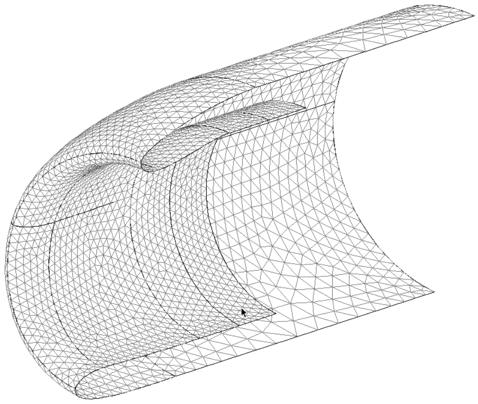


Figure 2. Unstructured surface mesh for engine nacelle

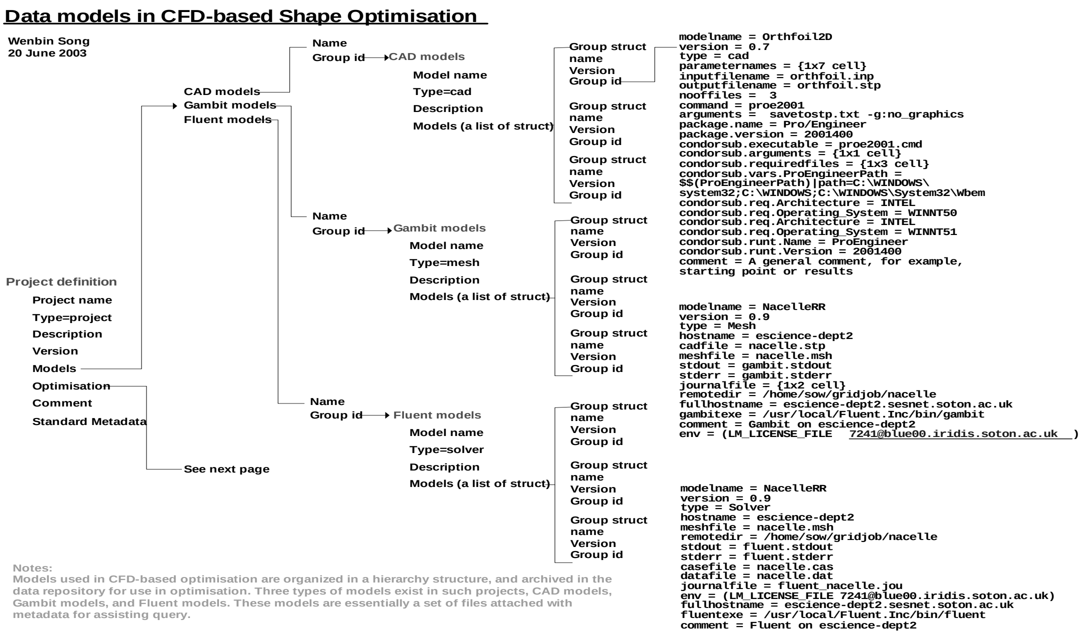


Figure 3. Hierarchy data models for CFD-based shape optimisation in data repository

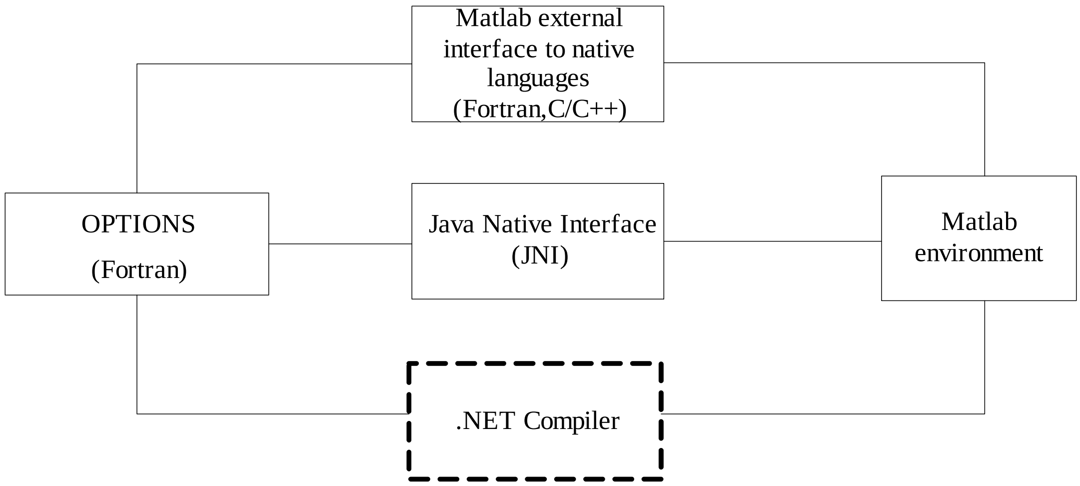


Figure 4. Interfaces to OPTIONS search engines in Matlab environment

4. Optimisation and Results

Optimisation is carried out using the design exploration system OPTIONS, which provides a collection of over 30 search algorithms including both deterministic and population-based stochastic methods. The functionalities of OPTIONS are exposed to Matlab users via the above mentioned approaches. Depending on the search procedures applied, these two approaches can be combined to produce complex workflows.

In the current work, a strategy of combining a design of experiment study, response surface modelling, and a two-stage hybrid genetic algorithm (GA) with gradient descent search is used on the problem. The procedure is illustrated in Figure 5, in which a design of experiment is first conducted and objective functions are evaluated by submitting the jobs to computational servers using the computation toolkits. The results are archived and later retrieved to construct the surrogate model. The sampled design space and constructed response surface model are shown in Figures 6a and 6b.

Optimisation is carried out on the response surface model using a Genetic Algorithm followed by a local search on the best point found using the Lagrange interval search method from OPTIONS. The result is then validated using the full Fluent model. It can be shown that the scarf angle has a much bigger effect on the pressure recovery than the axial offset. With the increase of negative scarf angle, the pressure recovery decreases, thus a compromise has to be made between aerodynamic performance and the noise reduction benefit associated with a large negative scarf angle.

5. Conclusions and future work

A Grid-enabled problem solving environment within Matlab provides easy access to computational and database capabilities to enhance the engineering optimisation process based on CFD results. An engine nacelle shape optimisation problem is tackled in this paper. The aim is to deliver better designs by leveraging high-fidelity computational analysis models with easy-to-use grid toolkits. Meta-data assisted data structure provides a high level abstraction to heterogeneous data involved in a CFD-based design process. Various search algorithms exposed in the form of Matlab functions enable complex search strategies to be constructed within Matlab. In the near future, more detailed studies on the effect of other geometry parameters on the noise reduction will be carried out.

It is also envisaged that a service-oriented infrastructure will gain increasingly wide adoption based on the Open Grid Service Architecture. And migration of existing application servers to Grid Services will enable the construction of various Virtual Organisations (VOs) across physical boundaries, while maintaining ownership and local policies.

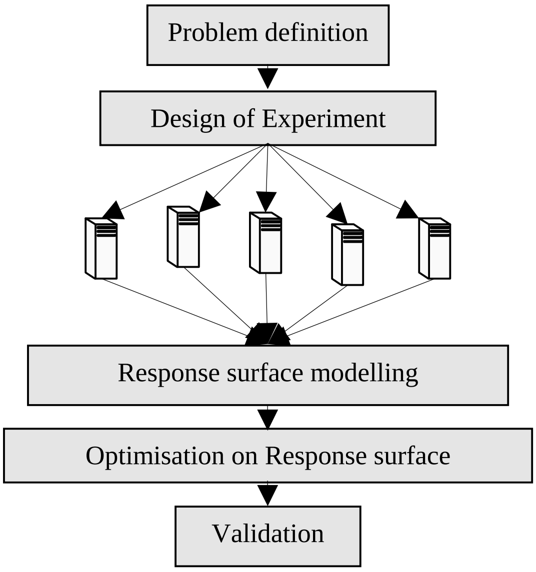


Figure 5. Optimisation Workflow Using Response Surface Modelling

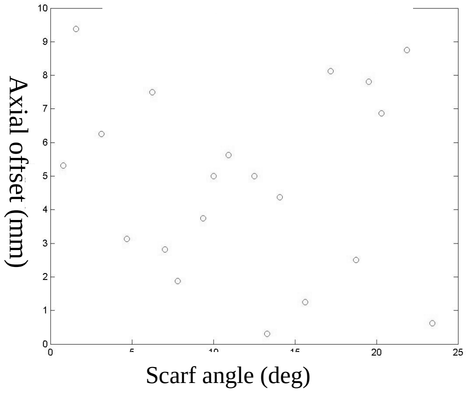


Figure 6a. Sampled Design Space

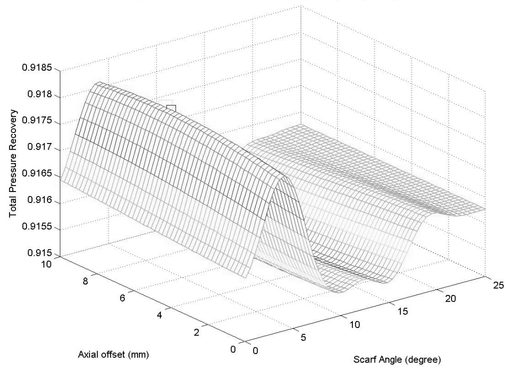


Figure 6b. Response surface model

Acknowledgements

This work is supported by the UK e-Science Pilot project (UK EPSRC GR/R67705/01). The authors gratefully acknowledge many helpful discussions with members of the Geodise team.

References

1. Sasaki, D., Yang, G., and Obay Ashi, S., "Automated aerodynamic optimisation system for sst wing-body configurations," AIAA-2002-5549, 2002.

2. The DataGrid Project, http://eu-datagrid.web.cern.ch/eu-datagrid/, 2003

3. Engineous Software, iSIGHT, http://www.engineous.com/index.htm, 2003

4. Phoenix Integration, ModelCenter, http://www.phoenix-int.com/index.html, 2003

5. Keane, A. J., OPTIONS: Design Exploration System, http://www.soton.ac.uk/~ajk/options.ps, 2003

6. Matlab, http://www.mathworks.com, 2003

7. Commodity Grid Kits, http://www.globus.org/cog/, 2003

8. The Globus Project, http://www.globus.org, 2003

9. The Geodise Project, http://www.geodise.org, 2003

10. Pound, G., Eres, M. H., Wason, J. L., Jiao, Z., Cox, S. J., and Keane, A. J., "A Grid-enabled Problem solving environment (PSE) for design Optimisation within MATLAB," 17th International Parallel and Distributed Processing Symposium (IPDPS 2003), 22-26 April 2003, Nice, France, 2003.

11. Wason, J., Molinari, M., Jiao, Z., and Cox, S. J., "Delivering Data Management for Engineers on the Grid," International Conference on Parallel and Distributed Computing, 26th - 29th August 2003, Klagenfurt, Austria, 2003.

12. Xue, G., Fairman, M. J., Pound, G. E., and Cox, S. J., "Implementation of a Grid Computation Toolkit for Design Optimisation with Matlab and Condor," International Conference on Parallel and Distributed Computing, 26th - 29th August 2003, Klagenfurt, Austria, 2003.

13. Condor Team, The Condor Project, http://www.cs.wisc.edu/condor, 2003

14. ProEngineer, http://www.ptc.com, 2003

15. Laszewski, G., Foster I., Gawor, J., and Lane P., "A Java Commodity Grid Toolkit," *Concurrency: Practice and Experience*, Vol. 13, No. 8-9, 2001, pp. 643-662.

16. Aftosmis, M. J., Delanaye, M., and Haimes, R., "Automatic Generation of CFD-Ready Surface Triangulations from CAD Geometry," AIAA 99-0776, 1999.

17. Fluent, http://www.fluent.com, 2003

18. Java Native Interface, http://www.sun.com/docs/books/tutorial/native1.1/, 2003