DEVELOPMENT OF A SIMULATION-BASED FRAMEWORK FOR EXPLOITING NEW TOOLS AND TECHNIQUES IN MULTIDISCIPLINARY DESIGN OPTIMIZATION

K.F. Hulme and C.L. Bloebaum
Multidisciplinary Optimization and Design Engineering Laboratory (MODEL)
Department of Mechanical and Aerospace Engineering
State University of New York at Buffalo - 1012 Furnas Hall
Buffalo, NY 14260 (USA)

Keywords: Multidisciplinary Design Optimization, Framework, Simulation, CASCADE

ABSTRACT

The primary goal of many Multidisciplinary Design Optimization (MDO) researchers is to simplify or shorten the design process in any of a number of ways. To address this ongoing research need, a number of previous research efforts have focused on the development of computational "frameworks". There appears to be a need for a framework which can exploit many of the newly developed tools, strategies and techniques in MDO. Another inter-related problem is the lack of availability of design data and benchmark (test) problems. Researchers must have a safe and robust means for testing a newly developed strategy prior to its implementation on an actual "real-world" design. Hence, the development of a framework that is focused on the incorporation of new MDO tools and techniques should have a robust means for coupled system simulation, both at the system analysis and optimization levels, as its foundation. This paper will discuss the design and development of an MDO framework entitled FACETS. This research effort further describes some of the modular components that are being designed, implemented, and integrated within FACETS, all of which hope to aide in the simplification of some portion of the costly MDO design cycle.

MOTIVATION

Multidisciplinary Design Optimization (MDO) is an emerging field of study which strives to promote concurrency of operations amongst participating design groups in a large scale engineering design. One of the primary goals of MDO is decomposition; the break-down of a large, complex engineering system into a grouping of smaller and inter-related (coupled) subsystems. An example of decomposition in design engineering is the break-down of the design of an aircraft into subsystems of design specialty, such as aerodynamics, control systems, structure/materials, propulsion, and so forth. Sobieski pioneered this concept when he applied a linear decomposition method for utilization in serial (hierarchic) environments (Sobieszczanski-Sobieski, 1982). However, the decomposition of a large, complex engineering system is rarely 100% serial in nature, thereby requiring that all subsystems transfer design information in some non-serial fashion. Ultimately, this requires that each subsystem take an "educated guess" as to the information it will receive from other subsystems, and thereafter formulate its design accordingly, on the present design iteration. The more accurate that the "educated guess" is, the fewer total iterations that will be required to achieve a converged design solution. The Global Sensitivity Equation method (Sobieszczanski-Sobieski, 1990; Bloebaum et al., 1990) extended the modularity concept of Sobieski's approach to include applications in the non-serial environments that typically exist in multidisciplinary problems.

The design cycle associated with non-serial multidisciplinary systems has often been referred to in literature as the “Multiple-Discipline-Feasible”, or MDF approach (Cramer, et al., 1992; Balling and Sobieszczanski-Sobieski, 1994). Refer to Figure 1. The design must first be initialized to some logical set of starting values. Perhaps current design data is utilized as the initial design, upon which improvements are desired. Design initialization is typically followed by a system analysis, which is typically an iterative and high cost process. Subsequent to the system analysis is a sensitivity analysis, which can be either numerical (i.e. finite differencing) or analytical (i.e. Global Sensitivity Equations) in nature. These sensitivities are then used in a formal optimization procedure, during which the converged behavior variables from the system analysis are held constant, and the design variables, which were held constant during the system analysis, are now changeable parameters. After the optimization procedure, the design variables have changed from their starting values. Because the behavior variables are a

function of these design variables, they will have changed as well. The design is updated, a convergence check is implemented, and the entire procedure repeats itself until a converged solution is attained.

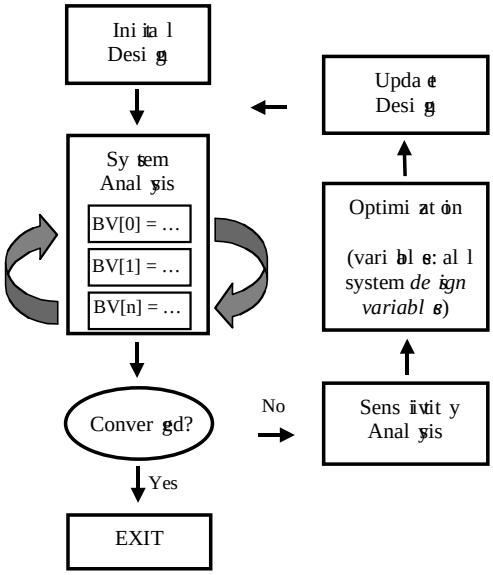

Figure 1: Multiple-Discipline Feasible

The advantages of the MDF approach include its commonality to most MDO researchers, and the inherent nature of its optimization problem, which treats only subsystem design variables as optimization variables. The primary disadvantage of the MDF approach is that it is potentially very time and cost consuming. At each optimization iteration, complete multidisciplinary feasibility is enforced. At each design cycle, a great deal of time may be inefficiently spent during the full re-convergence of the system analysis portion of a design that is still very far from its optimal solution. A large number of recent research efforts have focused on increasing the efficiency of the most costly component of the MDF design cycle, the iterative system analysis, in the interest of reducing the overall number of iterations expended per design cycle. Design or task sequencing is the procedure of re-ordering the subsystem modules in a way such that the number of analysis iterations are minimized. Steward's Design Structure Matrix concept (Steward, 1981) provides a graphical means for interpreting this concept. Refer to Figure 2. The modules are typically ordered such that the number of feedbacks (which represent an exchange of data that is presently unknown, and whose value requires an "educated guess") are minimized. Complete elimination of feedbacks would imply a truly serial collection of modules, and iteration would be eliminated altogether. Rogers' DeMAID is a computational task sequencing tool (Rogers, 1989) that has been used extensively by both academia and industry. A more recent work (McCulley et al., 1997) saw the development of convergence strategies based not only on module sequence, but on the strength of the couplings between the modules, quantified by locally computed sensitivity information.

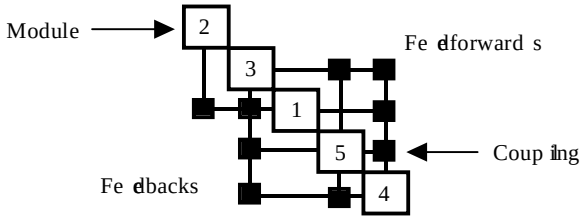

Figure 2: Design Structure Matrix

In an attempt to address these and other deficiencies in the MDF approach, numerous recent research efforts have focused on alternatives to MDF for posing and subsequently solving the multidisciplinary design problem. These include an approach that has been termed both "Simultaneous Analysis and Design" (SAND) and "All-At-Once" (AAO) (Haftka, 1985; Cramer, et al., 1992), which treats the entire multidisciplinary design cycle as a single optimization problem. This is accomplished by treating the system analysis equations as equality constraints, and by treating both design and behavior variables as changeable parameters (i.e. optimization variables) during the optimization procedure. A second alternate approach has been referred to as "Individual-Discipline Feasible" (IDF) (Haftka et al., 1992; Cramer, et al., 1992), which is intermediate to both MDF and AAO in the following ways. Similar to MDF, IDF does incorporate a system analysis of sorts, though it is non-iterative in nature. In addition, IDF is such that any behavior variable that promotes coupling (i.e. is required by at least one other subsystem as input) is promoted to being an optimization variable, similar to AAO.

Clearly, there are many research avenues in the field of MDO which ultimately strive to address to very same research issue – the reduction of time and cost within a multidisciplinary design cycle. To address this seminal issue, numerous computational problem-solving environments, commonly referred to as “frameworks”, have been developed in recent years. A framework has been loosely defined in literature as a “hardware and software architecture that enables integration, execution, and communication among diverse disciplinary processes" (Rogers et al., 1998). (Salas and Townsend, 1998) outlined many of the essential requirements which a viable MDO framework must contain. These include, but are not limited to the aspects of: “architectural design” (i.e. the framework should have an intuitive Graphical User Interface), “problem formulation construction” (i.e. the framework should support legacy and proprietary codes), “problem execution” (i.e. the framework should automate the execution of processes, and allow for parallel execution

and user interaction), and “information access” (i.e. the framework should provide visualization and monitoring capabilities). Some recently developed frameworks in MDO which incorporate some combination of these characteristics include FIDO (Framework for Interdisciplinary Design Optimization), which serves as a general distributed computing system for executing multidisciplinary computations on a heterogeneous network or workstations. FIDO automates the coordination of analyses by numerous design disciplines into an integrated optimization scheme, and allows for visualization and “steering” by the designer. A number of other frameworks and/or problem solving environments are presently under development. These range from alternatives to FIDO which focus more on exploiting distributed, heterogeneous computing, such as Access Manager, developed by Boeing. Others include commercial optimization toolkit environments such as iSIGHT, which allows the user to flexibly integrate analyses with optimization methods of all forms (i.e. numerical, heuristic, Design of Experiments, etc.) Finally, others include design tools that focus more on data management within the design process, such as IMAGE, which provides data management utilities for use during design processes.

The primary shortcomings of many existing MDO frameworks are that they tend to be hard-coded, discipline or problem specific, and have limited capabilities when it comes to the incorporation of new technologies. There appears to be a need for a framework which can exploit many of the newly developed tools, strategies and techniques in MDO which strive to “simplify” (i.e. reduce the time and cost of) the design cycle associated with large, coupled engineering design problems. Many of these tools and techniques were outlined at the beginning of this section. These and other computational techniques can be viewed upon as "islands of research" in that they are independently developed computer codes and concepts, which are (at present) physically separated, yet functionally related. A design manager may benefit from an environment which allows the combination and/or integration of such research islands, such that related research concepts can be merged, and numerous "what if?" scenarios can be explored quickly and easily. An inter-related problem involving many of these research islands, and MDO research in general, is the lack of availability of design data and benchmark (test) problems. Researchers must have a safe and robust means for testing a newly developed MDO strategy prior to its implementation on an actual "real-world" design. Hence, the development of a framework that is focused on the incorporation of new MDO tools and techniques should have a robust means for coupled system simulation, both at the system analysis and optimization levels, as its foundation.

FACETS

The Framework that is presently under development at SUNY Buffalo is called FACETS, which is an acronym that stands for "Framework for the Analysis of Coupled Engineering Techniques in Simulation". The original concept for this research effort was to utilize the Java programming language, an interpreted programming language whose compiler uses byte-code rather than native machine code, to program the exterior structure of the framework. In doing so, the framework would have taken strides towards being heterogeneous; capable of execution on any architecture that has a Java interpreter. Java programs that are written as applets can be executed through the use of a web browser, which are typically available on all computational platforms (PC/Macintosh/Workstation) and operating systems. Unfortunately, the shortcomings of using Java as a large-scale MDO tool are numerous; namely, security limitations on Java applets for both a.) reading and writing files, as well as b.) instantiating system-level commands. These are operations that are fundamental to any large scale software tool that relies on communication between related computer codes and functions.

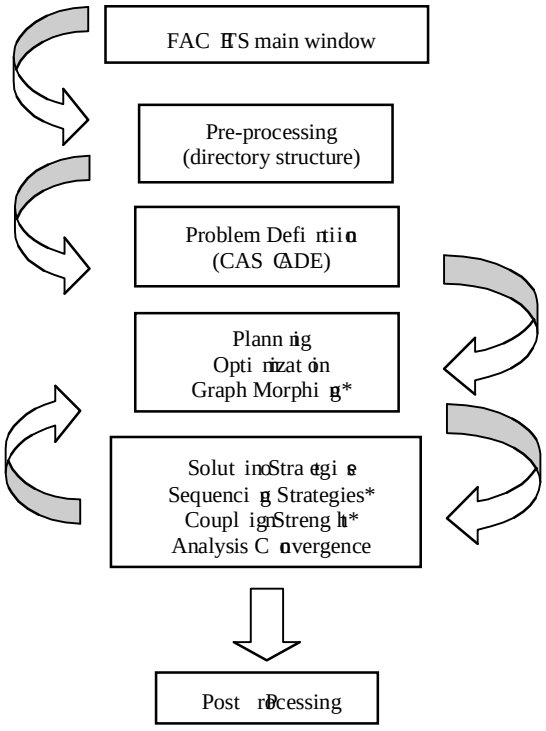

Figure 3: FACETS: general structure

As a result of this, the exterior front-end of the FACETS framework has a Graphical User Interface (GUI) that has been coded using the *Motif* toolkit. Motif, which is coded in the ANSI C programming language, was designed by the Open Software Foundation. The OSF is a consortium of companies such as Hewlett-Packard, IBM, Digital, and others, whose charter calls for the "development of

technologies that will enhance interoperability between computers from different manufacturers". Motif is based on the X Windows System, which is a network-based windowing system that has been implemented for UNIX, DOS, Macintosh, and other operating systems, and serves as a flexible foundation for GUI-based programming (Heller and Ferguson, 1994). Each of the modules within FACETS implements the easy to use control "widgets" that are typical of all Graphical User Interfaces. These include push buttons, scrollbars, toggle (radio) buttons, textfields, and others. A generic flowchart of the structural operability of FACETS is seen in Figure 3. Note that the modules marked with an asterisk (*) are not yet incorporated into FACETS. The other modules will be discussed in the next section.

When the user instantiates FACETS, the main GUI window appears as shown in Figure 4. Hereafter, the user clicks on the FACETS name, which is in fact a colored button. This event then brings up a window which allows the user to establish his directory structure; this window is shown in Figure 5. Here, the user uses the textfields to prescribe the location of file paths for each module. This is necessary because FACETS makes use of quite a large number of modules, each of which requires a number of input and output files for successful operation. It is for this reason that this "directory structure" window is the first module presented to the user. This information must be established prior to entering any of the feature modules of FACETS.

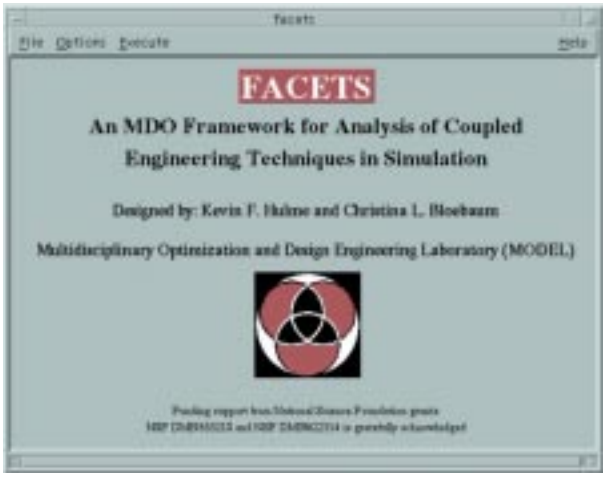

Figure 4: FACETS: main window

As hinted upon previously, FACETS operates upon the concept of "modularity"; each application or feature of FACETS functions as an individual entity. This flexibility allows for the addition of new technologies to, or the removal of dated applications from the framework at any time. The modules within FACETS then "communicate" through numerous means. Typically, simple numerical data is stored in data files which can be written as output by one module, and read in as input by another module. For example, the final values of the constraint functions are written to a data file by a given module, and are then read in by the post processing module for presentation. Some modules create output in the form of a compilable language file. For example, the analysis equations generated by the CASCADE simulator (explained in the next section) are written in the form of ANSI C based functions, and are all encapsulated within a single header file. At present, all output files are written in ANSI C, but this process could easily be extended to other languages, such as Fortran, Java or even HTML. These language files can then be compiled with other codes and fully used by other modules. Any "command line" activity is accomplished by FACETS via the UNIX-based "system" command:

system("command"); (1)

where "command" is a character string in the form of an operating system command. For example, should the user wish to compile two output files that are located in the /tmp directory, the system command might appear as follows:

system ("cc –o /tmp/out /tmp/out1.c /tmp/out2.c"); (2)

The system command is used primarily for compiling and executing codes, and is triggered through a widget event, namely the depression of a push button.

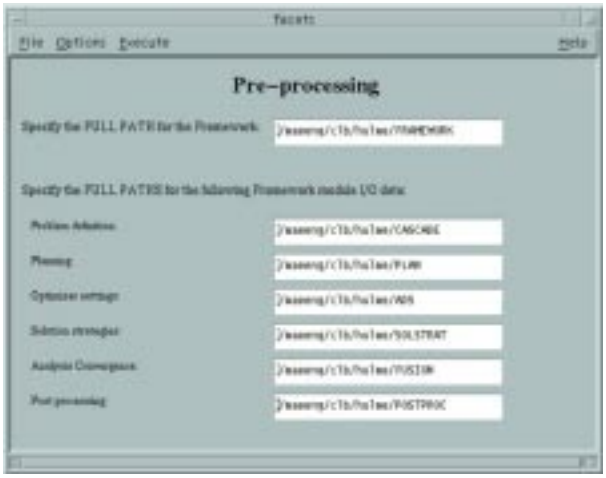

Figure 5: FACETS: directory structure window

Some additional general features of the FACETS framework are worthy of mention. The user has the power to interrupt the execution of a module prior to its completion. This might be desirable if the user notices that the solution process is proceeding into a disadvantageous region. Eventually, FACETS will allow the user to make some form of an adjustment during the interrupt period, and then restart execution after this alteration has been made. An additional feature is data storage. The user may wish to save or "store" result data after an execution of FACETS, which can later be retrieved and "opened" for further use. This process can take place before or after the execution of any of the feature modules in FACETS.

The next section will address some of the specific modules that have been or will soon be incorporated into the FACETS framework.

MODULES OF FACETS

This section will discuss all of the modules that will appear in the "first release" of the FACETS framework. The modules that have already been developed by the authors will be emphasized. It is envisioned that the incorporation of future modules will include MDO strategies developed by exterior researchers.

CASCADE

After instantiating the framework and establishing a directory structure, the user must define the multidisciplinary problem data. The long-range goal for this framework is to accommodate "real world" design data. At present, problem data is artificially generated using the CASCADE simulator. CASCADE is an acronym which stands for “Complex Application Simulator for the Creation of Analytical Design Equations”. A thorough description of CASCADE can be found in past literature (Hulme and Bloebaum, 1997). A brief overview will be presented here for completeness.

CASCADE is a computer tool that generates a coupled system that consists of polynomial equations of user-specified size. CASCADE has the capability of generating equations whose structure can be thought to represent both a coupled system analysis and an associated coupled optimization problem. The analysis portion consists of a band of nonlinear equations which are meant to represent the coupled nature of the subsystem outputs, sometimes called behavior variables. The optimization portion consists of an objective function, inequality constraint functions, and side constraints. The CASCADE module allows the user to custom-generate a multidisciplinary test system to his needs. For the system analysis portion (shown), the user defines the number of subsystems, behavior and design variables per subsystem, convergence and system volatility (stability) characteristics, and others. For the optimization portion, the user defines the "nature" of the objective function(s) (i.e. system level, or subsystem level), the number of inequality constraints per subsystem, and others. In addition, associated generic evaluation costs can be defined for the analysis behavior variables, and for the objective function and inequality constraint functions. Such information could be useful when comparing multidisciplinary solution strategies. Being that the FACETS framework is primarily interested in testing MDO methods and strategies, CASCADE, which ultimately provides the multidisciplinary problem data, is truly its flagship module.

Planning

The Planning module was inspired by a similar (and much more elaborate) module found in Rogers' DeMAID program. This module allows the user to visualize initial information about the problem being solved. This module informs the user as to whether or not there are a). behavior variables which are not coupled (i.e. are not required as input by other behavior variables), or b). behavior variables which do not require any other behavior variables as input. Such information could be useful when comparing strategies for multidisciplinary analysis. The Planning module also allows the user to visualize the initial value of the system objective function (or the summed value of the subsystem objective functions). At present, the user is given the option to seek out an alternate initial feasible starting design point by way of a random search.

Optimization

The FACETS framework presently makes use of the Automated Design Synthesis (ADS) program (Vanderplaats, 1985) as its sole optimization tool. ADS allows the user to change a large number of options corresponding to the gradient-based optimization search. Such parameters relate to the optimization strategy, optimization method, the one-dimensional search method, move limits, constraint thickness, finite difference parameters, variable scaling, and many others. A module has been devoted to providing the user with an easy means for assigning and altering such options quickly and easily. This will allow for the efficient execution of "what if?" scenarios within the context of any MDO techniques which require numerical optimization.

Solution Strategies

As discussed previously, there are numerous means for posing and subsequently solving a multidisciplinary design problem. The most popular and well known strategy has been called Multiple-Discipline Feasible (MDF). Two of the more popular alternate strategies are called All-at-Once (AAO) and Individual-Discipline Feasible (IDF). A FACETS module has been created which allows the user to compare these solution strategies within the context of the same multidisciplinary design problem. Depending on the strategy chosen, there are numerous implementation issues to be considered. For example, if an explicit system analysis is utilized in the solution strategy chosen (MDF and IDF), there are numerous convergence-related options to be assigned. Also, if there are additional optimization variables introduced in the solution strategy chosen (IDF and AAO), corresponding side constraints must be defined. As a baseline, Random Search is also included as an available multidisciplinary solution strategy.

Analysis Convergence

As discussed previously, the system analysis tends to be the most costly component of the multidisciplinary design cycle. Hence, a module has been devoted to allow for the comparison of strategies for approaching a multidisciplinary analysis. These range from well-known formal convergence strategies such as Fixed-point Iteration which is based upon successive substitution, and Newton's Method, which is derived from a Taylor Series expansion. Both of these techniques are available as analysis convergence options within FACETS. Each of these techniques have different strengths and weaknesses. Because of this, alternative heuristic techniques are under development. The authors have devised a scheme for analysis convergence which implements a neural network model - each subsystem output is modeled as an output neuron. For coordination amongst the output neurons, a systems-based procedure known as data fusion is then utilized. Four fusion models have been developed thus far, all of which are available as options within the FACETS analysis convergence module.

Post Processing

Finally, a means for quantifying results is imperative. FACETS post-processor provides the user with a variety of means for result visualization. These include an "optimization feedback" sub-module, which allows the user to view the final value of the objective function, as well as a color-coded means for visualizing constraint status (active/violated/satisfied). The "time and cost" sub-module allows the user to view the overall CPU time and the generic cost associated with the multidisciplinary solution cycle. Finally, a "plotting" sub-module uses the XMGR (Turner, 1991) routine to provide the user with 2D plots of iteration convergence at the analysis level, objective function convergence at the MDO-cycle level, and design and behavior variable histories at both the analysis and MDO-cycle levels.

The framework has been used to attain preliminary results in two major research endeavors thus far. These will be discussed briefly in the next section.

FRAMEWORK APPLICATION: PRELIMINARY RESULTS

The first major application of the framework came in the form of a large-scale comparison of multidisciplinary solution strategies (Hulme and Bloebaum, 1998). Clearly, many of the modules of FACETS were required for this research endeavor: CASCADE was used to generate the test systems; the planning module was used to ascertain the initial design point and the initial coupling status of the system; the optimization module was used to assign the options for the various optimizations associated with the different implementations; the solution strategies module was used to define the strategy of choice for each execution, and the post-processing module was then used to view convergence plots, evaluation costs, and constraint activity. Many results were attained, and numerous conclusions were drawn. Refer to Figure 6. The general findings are described as follows for the following test system:

Initial objective function: *197.40*
Subsystems: *20*
Behavior variables: *100*
Design variables: *40*
Inequality constraints: *3*

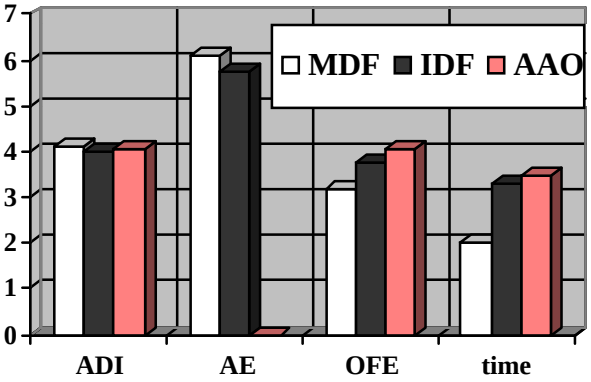

Figure 6: Solution strategy comparison ($\log_{10}$)

For substantially large multidisciplinary systems, MDF, IDF, and AAO are all useful for attaining an improved solution. The MDF strategy, which often attains the greatest design improvement (DI), suffers from the fact that it requires a huge number of analysis evaluations (AE) per MDO cycle. On the other hand, AAO has a more complex optimization problem and requires a large number of optimization function evaluations (OFE), and typically takes the longest time to execute. IDF usually shows itself to be a good intermediate choice to the MDF/AAO extremes.

The second major application of the framework came in the form of a comparison of formal and heuristic convergence strategies for multidisciplinary analysis (Hulme and Bloebaum, 1999). Again, many of the modules of FACETS were required for this research endeavor: CASCADE was used to generate the test systems; the optimization module was used to assign the options for the various sub-optimizations associated with the neural network error minimizations; the analysis convergence module was used to define the analysis strategy for each execution; and the post-processing module was then used to view convergence plots. Many results were attained, and numerous conclusions were drawn. Refer to Figure 7. The general findings are described as follows for the following test systems:

Systems 1, 2, and 3: 3 behavior variable equations
System 4: 8 behavior variable equations
System 5: 11 behavior variable equations

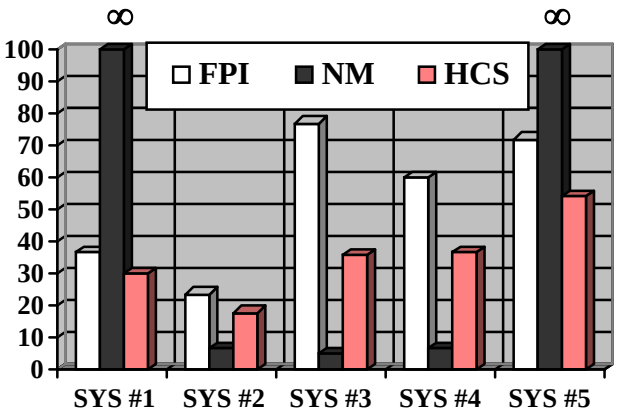

Figure 7: Analysis convergence comparison

Newton's Method (NM) was found to be the most appropriate convergence strategy for three of the five test systems. For the other two test systems, Newton's Method diverged. In other words, the starting design was not "sufficiently close" enough to the converged design to allow for convergence. In these cases, the "heuristic" (i.e. the previously described neural network / data fusion model) convergence strategy (HCS) was found to be the superior convergence strategy. In fact, the heuristic strategy outperformed Fixed-point Iteration (FPI) for all five test systems.

These two implementations of FACETS have been quite sequential in nature, in terms of modular usage. For baseline testing, this is more than sufficient. However, for future applications of FACETS, the authors hope to implement more complex (i.e. non-serial) combinations of its modular MDO tools and technologies.

SUMMARY AND FUTURE WORK

This paper has presented a discussion of FACETS, a new multidisciplinary design optimization framework that is presently under development. FACETS differs from other MDO frameworks in numerous ways. Namely, it seeks to exploit new tools and technologies in MDO which deal with large-scale, coupled engineering design problems. Because of this, FACETS uses simulation as its backbone. More specifically, the CASCADE simulator has been implemented as its means for generating benchmark problem data upon which new MDO methodologies can be tested. The structure of FACETS has been coded in Motif, an X-Windows based toolkit for GUI programming. In addition to the problem generation simulator (CASCADE), numerous modules have been incorporated into FACETS thus far. These include a module for comparing MDO solution strategies, a module for comparing convergence methods, and planning, optimization, and post processing modules. FACETS has already been extensively used for two MDO method comparisons; future usage and modular integration has already been planned for.

There are numerous research avenues that must be pursued to further the FACETS framework. First and foremost, exterior technologies shall be incorporated. All of the major features that lie within the framework have been developed and coded by the authors. The next step will be to incorporate related research technologies performed by other member of the SUNY Buffalo MODEL laboratory team. These will include graph morphing – a means for visualizing and "steering" the multidisciplinary design anytime within the design cycle (Winer and Bloebaum, 1997). In addition, coupling suspension strategies – means for identifying and responding to couplings between subsystems which are found to be comparatively weak, in the interest of reducing overall design time and cost (English and Bloebaum, 1998). Eventually, technologies developed by completely exterior researchers shall also be incorporated. Secondly, more optimization possibilities should be added to the framework. Because optimization is the backbone of a majority of the research that FACETS is involved with, alternative options to ADS should be provided. These might include heuristic optimization techniques such as simulated annealing, or more recent traditional optimization methods found in Vanderplaats' DOT software package. Finally, the authors plan to incorporate some degree of WWW technologies within FACETS. Many of the initial (Java-based) aspirations for this framework have not yet come to pass. The authors feel that at the very least, web technologies can be utilized during the result visualization (post processing) phase of the framework. The automated creation of HTML pages as output files can allow for quick and easy global access of result data attained through the usage of the framework.

ACKNOWLEDGEMENT

The authors would like to acknowledge the partial funding support of The National Science Foundation from the following Presidential Faculty Fellow (PFF) grant: NSF DMI9553210.

REFERENCES

Balling, R.J., and Sobieszczanski-Sobieski, J., "Optimization of Coupled Systems: A Critical Overview of Approaches." AIAA Paper 94-4339, September, 1994.

Bloebaum, C.L., Hajela, P., and Sobieszczanski-Sobieski, J., "Non-Hierarchic System Decomposition in Structural Optimization." Third USAF/NASA Symposium on Recent Advances in Multidisciplinary Analysis and Optimization, San Francisco, CA, September, 1990.

Cramer, E.J. et al., "On Alternative Problem Formulations for Multidisciplinary Design Optimization." Fourth AIAA /NASA /ISSMO Symposium on Multidisciplinary Analysis and Optimization, Cleveland, OH, September, 1992.

English, K., and Bloebaum, C.L., "Development of Multiple Cycle Coupling Suspension in Complex System Optimization", Seventh AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, September 1998.

Haftka, R.T., "Simultaneous Analysis and Design." *AIAA Journal*, Volume 23, Number 7, July, 1985.

Haftka, R.T., Sobieszczanski-Sobieski, J., and Padula, S.L., "On Options for Interdisciplinary Analysis and Design Optimization." *Structural Optimization*, Volume 4, Number 2, June 1992. pp. 65-74.

Heller, D., and Ferguson, P.M., "Motif Programming Manual – Volume 6A." O'Reilly & Associates, Inc, California, 1994.

Hulme, K.F., and Bloebaum, C.L., "Development of a Multidisciplinary Design Optimization Test Simulator." *Structural Optimization*, Volume 14, Number 2-3, October, 1997.

Hulme, K.F., and Bloebaum, C.L., "A Comparison of Solution Strategies for Simulation-based Multidisciplinary Design Optimization." Seventh AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, Missouri, September, 1998.

Hulme, K.F., and Bloebaum, C.L., "A Comparison of Formal and Heuristic Strategies for Convergence of a Coupled Multidisciplinary Analysis. Accepted for presentation at the 3rd World Congress on Structural and Multidisciplinary Optimization, Amherst, NY, May, 1999.

McCulley, C., Hulme, K.F., and Bloebaum, C.L., "Simulation-based Development of Heuristic Strategies for Complex System Convergence", Applied Mechanics Review, Vol. 50, November 11, 1997.

Rogers, J.L., "DeMAID — A Design Manager's Aide for Intelligent Decomposition: User's Guide." NASA Technical Memorandum 101575, March, 1989.

Rogers, J.L., Salas, A.O., and Weston, R.P., "A Web-Based Monitoring System for Multidisciplinary Design Projects." Seventh AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, Missouri, September, 1998.

Salas, A.O., and Townsend, J.C., "Framework Requirements for MDO Application Development." Seventh AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, Missouri, September, 1998.

Sobieszczanski-Sobieski, J., "A Linear Decomposition Method for Optimization Problems - Blueprint for Development," NASA Technical Memorandum 83248, 1982.

Sobieszczanski-Sobieski, J., "The Sensitivity of Complex, Internally Coupled Systems," AIAA Journal, Volume 28, No. 1, 1990, pp. 153-160.

Steward, D.V., "System Analysis and Management." Petrocelli Books, New York, 1981.

Turner, P.J. "XMGR – XY plotting tool for workstations using X." (internet documentation) Portland, OR, 1991-1995.

Vanderplaats, G., "ADS - A FORTRAN Program for Automated Design Synthesis Version 1.10" (user's manual), University of California, Santa Barbara, California, 1985.

Winer, E., and Bloebaum, C.L., "Design Visualization by Graph Morphing for Multidisciplinary Design Optimization", Conference Proceedings of First International Conference on Engineering Design and Automation (EDA '97), Bangkok, Thailand, 1997.