TOWARDS AN INTEGRATED MODELING ENVIRONMENT FOR HYPERSONIC VEHICLE DESIGN AND SYNTHESIS

Jeffrey V. Zweber*
Design Process & Environments Team, AFRL/VASD
Wright Patterson AFB, OH 45433

Hanee Kabis†
Engineering Methods Group, Lockheed Martin Missiles and Fire Control
Orlando, FL 32819

William W. Follett‡
Computational Fluid Dynamics, Boeing, Rocketdyne Division
Canoga Park, CA 91303

Narayan Ramabadran§
TechnoSoft, Inc.
Cincinnati, OH 45242

ABSTRACT

The US Air Force Research Laboratory, along with its contractor partners, is developing an integrated modeling environment for the conceptual and preliminary-level design and synthesis of airbreathing, hypersonic vehicles. This effort is built on the team's successful prototype of a similar environment for rocket-powered space access vehicles. The modeling environment under development will begin by developing a 3-4 level deep hierarchy of objects that represent a hypersonic vehicle. Initially, these objects will contain only conceptual-level representations of the geometry and mass properties of the vehicle and its components. This initial information will be used with a vehicle synthesis routine to develop a "closed" conceptual design. The second step in the design process is an initial analysis of the aerodynamic and propulsive characteristics of the vehicle. These analyses will be conducted in the environment and the geometric model developed in the initial hierarchy of objects will be of sufficient fidelity to support these analyses. Next, the mass properties, aerodynamic and propulsion analysis results will be used by a trajectory simulation code, also integrated into the environment, to determine if the initial vehicle design will meet the mission performance requirements. Finally, the results of the trajectory simulation will be used to iteratively resize the vehicle until the mission requirements are satisfied. Additionally, this paper will describe the modeling environment used for this effort, lessons learned from the development of the environment for rocket-powered vehicles, and the next steps planned to expand the capabilities of the integrated modeling environment.

INTRODUCTION

The US Air Force has a renewed interest in investigating airbreathing hypersonic vehicle concepts to meet its needs for future strike and reconnaissance systems [1]. In addition, NASA is continuing its investment in hypersonic airbreathing propulsion systems and vehicle concepts for space transportation applications. In recent years, these interests have been exemplified by NASA's X-43 (Hyper-X) program [2] and the Air Force's HyTech (Hypersonic Technology) program.

Like all engineering organizations, the Air Force Research Laboratory is interested in conducting its vehicle and technology forecasting studies as quickly as possible, with as high fidelity an analysis as is feasible and with a proven, repeatable design and analysis process. The approach that the Air Force Research Laboratory team has taken is to integrate its design, analysis and modeling tools into a collaborative, network-distributed design environment.

The benefits of using an integrated design environment to reduce the time and potential errors associated with the transfer of data between design and analysis codes are well documented [3, 4]. This paper will present the initial steps in the development of an integrated modeling and analysis application for hypersonic airbreathing vehicles. This application will be developed using a modern knowledge-based engineering environment and will incorporate the lessons learned from the development of a similar application for rocket-powered space access vehicles [5]. Furthermore, the current effort will demonstrate a significant reuse of much of the software that was developed for the launch vehicle application.

ADAPTIVE MODELING LANGUAGE

For this effort, the Adaptive Modeling Language® (AML™), developed by TechnoSoft, Inc., was selected as the design modeling environment. AML is a framework for Knowledge Based Engineering (KBE) that provides the ability to capture the vehicle design and analysis process and manage the data transfer between the various codes [6, 7]. The primary features of AML that led to its selection for this project are: its use of object-oriented programming and the Unified Part Model paradigm; its native understanding of geometric objects and features; and its support for multiple, simultaneous, network-distributed users.

Modeling Paradigm

The benefits of object-oriented programming (OOP) are well understood. OOP both increases the developer's ability to reuse code that was previously developed, and simplifies

---

* Aerospace Engineer, Senior Member AIAA
† Staff Research Engineer
‡ Engineer / Scientist
§ Software Applications Engineer

the initial development of the code. The Unified Part Model paradigm is an implementation of OOP in which the model of a given component, the fuselage for example, contains all the data about the fuselage. This paradigm helps ensure that the various models of the fuselage are consistent across the different disciplines.

To continue the fuselage example, this paradigm enforces the connection between the geometric model and the mass properties analysis. In the case of these disciplines, the mass estimating relationships (MERs), which are used to determine the vehicle's mass properties, are highly dependent on geometry (e.g., tank volume, fuselage surface area), as well as the overall vehicle weight and mission requirements. The Unified Part Model paradigm ensures that all weight items will be tracked by allowing the MERs to be included in the geometric objects. This can be visualized in the model tree shown in Figure 1. In this rocket-powered TSTO vehicle example, the tank-stack object contains both the geometry of the tanks shown and the weight and CG location of the tank.

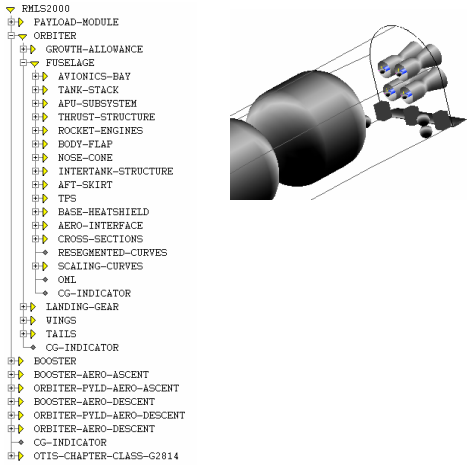

Figure 1: Unified Part Model Tree

In addition to being object oriented, AML has the ability to manage the relationships between the objects. As the model is being built; that is, as objects are being instantiated and as formulas are being coded to associate the parameters of one object with those of another object, AML is building a table of dependencies between the objects and properties of the model.

This "dependency tracking" is used to provide two computational benefits. First, dependency tracking allows AML to "smash" the values of all the variables that will change because of a change in a parameter upon which the variable is dependent. This feature means that the model will always be consistent. Once two parameters are related, the user can work with one portion of the model while not worrying that he is working with old data from another part of the model.

Related to AML's implementation of dependency tracking is its use of demand-driven computation. This feature means that for any requested calculation, only the calculations that are required will be computed, not the entire model. This feature is useful for complex models. For instance, in the fuselage model shown above, perhaps the tank geometry is calculated as an offset from the fuselage geometry. In this model, if the an aerodynamics engineer wanted to change the fuselage diameter and determine the external aerodynamic effects, the engineer would not have to wait for the tank geometry to be recomputed (or perhaps even the tank weights to be updated) because the aerodynamic calculation only requires information about the fuselage surface.

Geometric Modeling

In addition demonstrating the Unified Part Model paradigm, Figure 1 also shows AML's native geometric modeling capability. Included with AML's basic set of objects are classes to support a wide variety of commercial geometry engines. This feature means that geometric modeling is not handled as just another discipline. AML has a native understanding of solids and surfaces, can perform Boolean operations and has support for automatic mesh generation.

This capability is most easily demonstrated by considering AML's integration with Unigraphics and MSC.Patran. One of AML's supported geometry kernels is Parasolid, which is also the kernel for Unigraphics and MSC.Patran. This implementation means that AML can perform all the complex geometric calculations that are available in Unigraphics while natively transferring that geometry to MSC.Patran for meshing. Similar capabilities are available or under development for a variety of common CAD packages and commercial meshing programs.

Network-distributed (web-based) design modeling

The last major feature of AML that will be used for this application is AML's ability to allow multiple, simultaneous users to collaborate in a single engineering environment, even though they are distributed across a wide area network. AML supports two modes of operation (distributed-user collaboration and distributed-model collaboration), which can be used separately or together depending on the needs of the engineering team.

The distributed-user mode allows multiple users to interact simultaneously with a single model tree. In this mode, there is a single model that resides on a server with many users who have client interfaces on their local machine. Depending on the permissions granted, which can vary for each user, the users can view the model, change the values of model parameters, add objects to the model and even allow other users to see their current view of the model. This mode is useful for collaboration between engineers working in the same discipline. For instance, a novice aerodynamicist can receive help with the intricacies of a particular analysis code from a more senior engineer.

The second mode, distributed-model collaboration, is more useful for a multidisciplinary engineering project. In this mode, each engineer has an AML model that is tailored for his or her specific discipline. Then objects from the separate models can be connected through a central Object Request Broker (ORB). The use of an ORB allows disciplines to be added as needed. It can also allow models to connect and disconnect at will. This feature is useful for engineering teams that are spread across time zones. One discipline can start working; a second discipline can join (or rejoin) the collaboration and send and receive updates to the common objects; then the first discipline can disconnect from the collaboration.

The capability for simultaneous collaboration amongst multiple engineers will be useful for this project. Engineers from at least five different cities, spread across the United

States will be participating in the development of the hypersonic airbreathing vehicle design application.

ENGINEERING DISCIPLINES

The initial development of the integrated application will concentrate on conceptual design and synthesis of hypersonic airbreathing vehicle concepts. It is hoped that the design environment can then be expanded to model the preliminary and detailed levels of the development of a hypersonic vehicle. The authors believe this development strategy is feasible because of their experience with AML-based applications that are being used to capture and improve the detailed design of combustion engines [8].

Figure 2 shows the disciplines that are involved with the development of hypersonic vehicles. The figure also represents how the disciplines interact during the design process. Displayed in blue are the five disciplines involved in the conceptual-level synthesis of a hypersonic vehicle. Then, following sufficient iteration among these disciplines, the additional disciplines are added to refine the design of the vehicle through further iteration and studies.

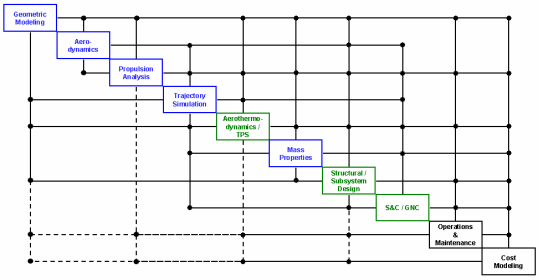

Figure 2: Hypersonic Airbreathing Vehicle Design Structure Matrix

For the effort documented here, the five disciplines that will be integrated into the application are: geometric modeling, aerodynamics, propulsion analysis (incl. flow path analysis), trajectory simulation and mass properties analysis. Additionally, this model development effort will take advantage of the inherent model and software reuse capability provide by AML's implementation of the object-oriented programming paradigm. This project will benefit greatly from the previous development of a design modeling application for rocket-powered launch vehicles [5]. Specifically, the aerodynamic analysis and trajectory simulation applications that were integrated for use on the previous effort will only require minor modifications to be applied to this application.

Synthesis and Geometric Modeling

The first step in developing a hypersonic vehicle design is to choose two configuration parameters, the vehicle class (e.g., 2-D lifting body, waverider, inward-turning or axisymmetric) and the design Mach number. Once these decisions are made, basic aerodynamic principles [9] can be used to define an initial propulsion flow path.

For the initial vehicle configurations that will be modeled in this effort, the FloGeo code from Boeing Rocketdyne will be used to generate propulsion flow path parameters for a 2-D lifting body configuration. FloGeo is method of characteristics based application that has been developed in Microsoft Excel. The application determines the necessary angles, lengths and other geometric properties of the inlet to end up with a "shock-on-lip" condition for the inlet flow at the specified design Mach number.

The next step in the design process is to develop an outer mold line (OML) model of the vehicle that is suitable for use with computational aerodynamic techniques. For this effort, a parametric OML model will be developed using AML's native geometric modeling capability. The process described here is for the 2-D lifting body class of vehicles, although a similar process could be used for the other classes of hypersonic vehicles.

The authors took this approach based on their previous experience with design modeling efforts for aerospace vehicles. They have found that developing different AML objects for different types of configurations is preferable to trying to develop a single parametric model capable of modeling the complete range of possible hypersonic vehicle configurations.

The first step in developing the OML is to model the propulsion flow path (i.e., keel line and cowl) as a set of parametric curves. The parameterization of these curves will match the output of FloGeo and may be used with an optimization procedure to improve the vehicle's performance at "off-design" Mach numbers.

Next, these curves are "extruded" to model the lower surface of the vehicle. The width of this extrusion will be determined from required internal volume of the vehicle. This required volume will be estimated initially, then verified by a mission and trajectory simulation.

Following the development the model of the vehicle's lower surface, a similar procedure is used to model the vehicle's upper surface. That is, first, a parametric curve is developed to control the shape of the upper surface, then that curve is extruded to create the upper surface model. The difference between the development of the upper surface and the lower surface is that the upper surface is designed solely based on aerodynamic and internal volume considerations, while the lower surface is strongly driven by its impacts on the performance of the propulsion system.

The last major step in developing the geometric model of the OML is to connect the upper and lower surfaces. This will be accomplished using the parametric surface modeling technique that was developed in AML for modeling aerospace surfaces. An example of this technique, used to build a parametric fuselage model, is shown in Figure 3. This fuselage geometry was built using two kinds of related profile objects termed "u" and "v" curves. These profile curves are controlled parametrically to shape and size each cross-section and the surface's behavior between the cross-sections at specified intervals. The external surface is then modeled as a nurb-surface that connects the various points that make up each profile curve.

For the OML of the 2-D lifting body class hypersonic vehicle, this profile curve paradigm will be used to connect the upper and lower surfaces. First, the edge of the upper and lower surface will be selected as the first and last v-curve for the side surface. Then points will be selected along each curve to form the starting points of a number of u-curves. Next, the u-curves will be developed freehand or with a simple mathematical formula. Finally, intermediate v-curves will be formulated or sketched freehand and the complete side surface will be automatically determined from these construction curves.

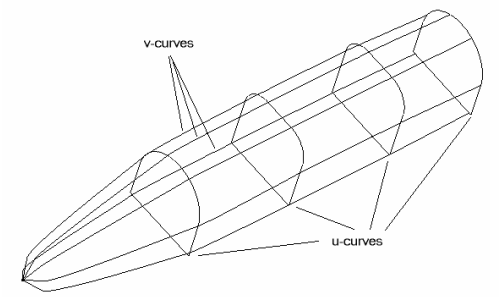

Figure 3: Fuselage Geometry Object

The final step in developing the geometric model, which will be used for computational aerodynamic and propulsion analysis, is adding control surfaces to the basic 2-D lifting body shape. There are two types of control surfaces that need to be developed, lifting surfaces and surfaces to control the propulsion flow path.

The models of the lifting surfaces (i.e., horizontal tails, vertical tails and canards) are developed in the same manner as the geometry model shown in Figure 3. Except in this case, the u-curves are airfoil sections and the v-curves are used to define the taper and twist of the lifting surface. Then, once the basic lifting surface has been modeled, it can be positioned relative to the 2-D lifting body shape and trimmed or joined to make a complete "watertight" surface.

To develop propulsion flow path control surfaces (e.g., a moving cowl lip or a moving inlet door for the turbine engine flow path of a TBCC engine), the OML that was generated by using basic aerodynamic principles, the FloGeo code in this case, must be modified.

Two methods are available to modify the OML. One option is to use Boolean operations on the OML. Using the cowl lip as an example, the process would require cutting the cowl to create the moving lip geometry; rotating the lip it to the desired angle; then trimming, extending and joining the rotated cowl lip to the remaining fixed portion of the original cowl. While this method is effective, and is commonly used by the aerospace industry, the method does not easily lend itself to design automation or optimization. The author's preferred method is to include parameters for these moving propulsion flow path surfaces in the original flow path curves that were modeled in AML. Then, a design model can be developed to automatically regenerate the lower OML surfaces when the parameters that change the propulsion flow path are varied. For the cowl lip example, the procedure would be generating a new cowl curve, extruding that curve to create the cowl surface and modifying the engine sidewalls to attach the cowl to the rest of the vehicle.

Mass Properties Estimation

The second discipline that is needed for the conceptual design and synthesis of a hypersonic vehicle is mass properties analysis. For this project, the authors have taken two approaches to integrate mass properties analysis into the design environment. The first, and simplest, method is to link an Excel spreadsheet-based weights model to AML. The second, and preferred, method is expand the AML geometry objects that were developed in the previous section so that these objects contain properties, objects and methods that will calculate estimates of the object's mass properties.

Spreadsheet-based Weights Model

The spreadsheet-based weights model that was used for this project was developed at the NASA Langley Research Center. This model consists of a main, system-level sheet with links to five discipline specific sheets (i.e., propulsion, structures, subsystems, landing gear and the thermal protection system for the airframe). The main sheet is the only one that needs to be linked to AML, with the other sheets being connected through the main spreadsheet.

The main, system-level spreadsheet takes two types of input, geometric parameters and design parameters. The geometric parameters will come from the OML model that was described in the previous section. Examples of the geometric information that is needed as inputs to the mass properties spreadsheet are: vehicle internal volume; wetted and planform areas for the fuselage and tails; surface areas covered by the various types of thermal protection systems (TPS); fuselage length; and combustor length.

The second type of inputs that are needed for mass properties analysis are determined either from mission requirements or from other, non-geometric, design decisions. Examples of these design inputs are: payload weight and volume; number and thrust level of the rocket engines, if needed for single stage to orbit vehicles; TPS unit weights, which are based on the type of TPS selected; vehicle design g-limit; and propellant fraction required.

Once the input design parameters and geometric parameters are determined, a set of mass estimating relationships (MERs) is used to determine an initial estimate of the mass properties of the vehicle. For this project, the team will use two types of MERs; one type based on component geometry and a second type based on system similarity.

An example of a geometry-based MER is Equation 1. This equation estimates the weight of the vertical tail based on the surface area of the tail. This equation was developed by fitting a curve to the historical data shown in Figure 4.

$$WT = 5 * S_{vt}^{1.09} * 0.89 \qquad [1]$$

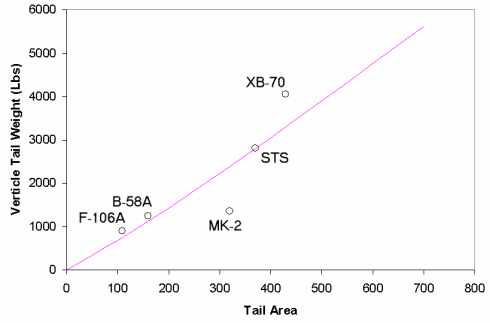

Figure 4: Vertical Tail Weight vs. Area

An example of the system similarity type of MER is the one used to estimate the mass of the vehicle's electrical system. In this case, the team estimated that the weight of the electrical system power supply will be 770 lbs, because that is the weight of the system on the Space Shuttle.

While some MERs, like the two examples presented above, are explicit, other MERs require and implicit solution method. For instance, the MER for the landing gear mass

strongly depends on the overall vehicle weight, which in turn is weakly dependent on the landing gear mass. This type of relationship leads to the need for an iterative process to ensure that the sized vehicle has a consistent set of mass properties.

The iteration process used with the spreadsheet-based weights model is based on determining a photographic scaling factor that makes the internal volume available match the internal volume required. This process requires the extra step of estimating component volumes as well as masses. The volume estimates are developed using a set of relationships that are conceptually similar to the MERs described above. The result of the iteration process is a scaling factor that can be used to scale or redraw the geometric model described above.

AML-based Weights Model

The major limitation of the spreadsheet-based weights model is that it is limited to photographic scaling of the vehicle that was created using the geometric modeling process. A few of the potentially useful vehicle changes that photographic scaling does not allow are: changing the vehicle's length and width independently (e.g., constraining the vehicle's length or width) or modifying the vehicle's upper surface shape to change the volume while keeping the vehicle's planform fixed.

The photographic scaling limitation will be eliminated in the second mass properties estimation method, the AML-based weights model. In this model, the mass estimating relationships will be incorporated into the geometric objects that were used to develop the OML model and geometric objects used to model the subsystems and internal components. This model will take advantage of AML's unified part model paradigm that was described above and is shown in Figure 1.

The major difference between the AML-based weights model and the spreadsheet-based weight model can be seen by examining the implementation of the vertical tail weight MER shown in Equation 1 above. In the case of the spreadsheet-based weights model, to calculate the vertical tail weight, the tail area (or the tail span and chords) must be extracted from the AML-based geometry model of the OML and input as parameters to the spreadsheet. However, in the AML-based weights modeling approach, a property called mass is added to the AML object that was already developed to model the geometry of the vertical tail. This new property can be programmed to calculate the weight of the vertical tail using the tail area or other parameters that are already available in the vertical tail object. These parameters are already available because they were needed for the geometric model.

The vertical tail weight model also exemplifies the paradigm that the methods for calculating the weights of each component of the vehicle will be developed as part of the object that also contains the top-level, conceptual geometric information about vehicle. Note however, that the fidelity of the geometric model and mass properties calculation may vary from vehicle component to vehicle component. For instance, the power system may be represented by boxes of fixed volume and weight, while the landing gear model may be made up of many additional geometric pieces (i.e., tires, brakes, struts, etc.) and have a complex MER based on takeoff weight, landing speed and runway surface.

Once MERs are integrated with a sufficient number of the AML objects to represent all of the major weight items, an iterative procedure must be developed in AML to ensure that the MERs are consistent. Like the spreadsheet-based weights model, this iteration will involve changing the size of the vehicle. However, because the mass properties estimate is tightly tied to the geometric components, design engineers will be able to develop more complex sizing routines. For instance, tail areas could be sized based on stability considerations and the vehicle's width could be constrained to accommodate an integer number of fixed sized airbreathing engines. This flexibility should allow the design team to synthesize feasible vehicles more easily.

Propulsion Flow Path Analysis

The bulk of the effort associated with this project will be the integration of ramjet and scramjet design and analysis tools into the AML environment. The first decision that needs to be made is what fidelity of propulsion flow path analysis is needed for conceptual-level design and synthesis of a hypersonic vehicle. Above, we described the FloGeo code, which will predict the on-design performance of the propulsion system. While this may be sufficient for the synthesis of cruise vehicles, a better prediction of the vehicle's off-design performance is needed for the design of accelerator configurations for access-to-space applications.

For this effort, the low-fidelity method that the authors have chosen to determine the performance of propulsion system is a combination of three codes. Together, Rocketdyne's FAST code, along with MCIA and L1IA from Lockheed Martin, provide tip-to-tail analysis of the propulsion flow path. MCIA will analyze the vehicle's inlet from its nose to the beginning of the isolator, FAST is an "engine deck" that will model the isolator and combustor, while L1IA will complete the analysis from the end of the combustor through the nozzle.

In addition to integrating these codes into AML, objects and methods will need to be developed to ensure that the flow conditions are consistent between MCIA and FAST as well as between FAST and L1IA. Finally, an AML procedure will be developed to create a table of propulsive forces and moments at various flight conditions (i.e., Mach Number, dynamic pressure and angle of attack). This table is needed for use with the trajectory simulation tool.

An alternate method was considered for calculating the off-design performance of the propulsion system. This method has two advantages. First, it is a single analysis code and second, it was already integrated into AML under a previous effort. The code that was considered is SRGULL [10] from NASA's Langley Research Center.

SRGULL uses the same approach for analyzing propulsion flow path as was described above. Namely, the flow path is divided into a forebody/inlet region, a combustor section and the nozzle. In SRGULL, the program is divided into subroutines to handle each section as opposed to the separate codes that were described above. As for the fidelity of the code, the calculations in the forebody/inlet and nozzle regions are 2-D or axisymmetric with 3-D corrections and the combustor flow is calculated using a 1-D method.

The inputs required by SRGULL are the flow path geometry, the flight conditions, fuel type and throttle setting. When the code was integrated with AML, a user interface

was also developed. A sample of the user interface is shown in Figure 5.

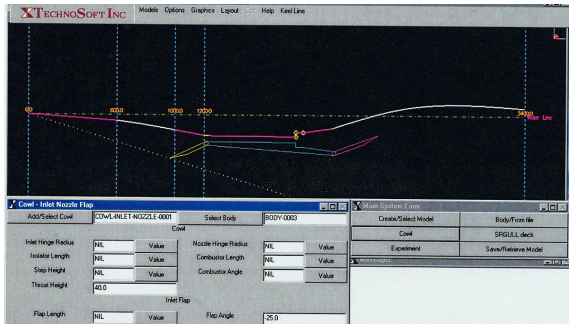

Figure 5: Example of the User Interface for Keel Line Modeling for the SRGULL Analysis Code

Even though SRGULL was already integrated into AML, it was not used for this project. This decision was made because the team was not comfortable with the code, whereas the team has extensive experience with the other codes described above. The authors have found that this situation occurs often. Typically, engineers have invested a considerable amount of time and effort in learning and improving the software tools that they use regularly and they are not willing to change software just because it is already part of an integrated process. The authors believe that each organization's design process is unique and that any integrated design environment must be tailored to support the process that already exists within an organization.

Aerodynamic Analysis

The next discipline considered, aerodynamic analysis, does allow the authors to take advantage of codes that were previously integrated into AML. For this effort, the team was comfortable with using Missile DATCOM [11], PANAIR [12] and S/HABP to determine the conceptual-level aerodynamic characteristics of the vehicle. Missile DATCOM is a semi-empirical code that can determine the forces and moments on a cylindrical or nearly cylindrical body, with small protuberances and axisymmetric finsets, over a wide range of Mach numbers. Some error will occur in using Missile DATCOM to analyze hypersonic vehicle shapes, however the team will use PANAIR to correct the calculation. PANAIR is a general-purpose aerodynamic code that uses a linear panel method. PANAIR is capable of determining the pressures on bodies and surfaces of arbitrary shape at subsonic and supersonic speeds. The final aerodynamic code that will be used is S/HABP (Supersonic/Hypersonic Arbitrary Body Program). S/HABP uses first order methods to calculate the pressures on arbitrarily shaped bodies and lifting surfaces at supersonic and hypersonic speeds.

Along with integrating these codes into AML, a limited visualization capability has also been developed in AML. For example, Figure 6 shows a simple plot of aerodynamic data and Figure 7 illustrates a typical body pressure distribution.

The final step in developing the aerodynamic analysis objects for hypersonic vehicles is implementing a method for determining or describing which portions of the OML are associated with the propulsion flow path and which areas are outside the propulsion flow path. This is very important for hypersonic vehicles because of the vastly different analyses that are used in each area. For instance, for a 2-D lifting body shape, the pressures, forces and moments on the entire lower surface of the vehicle are calculated using the propulsion analysis tools and the loads on the sides and upper surface of the vehicle are determined using regular aerodynamic analysis tools. Finally, the forces and moments on the external sidewalls and cowl of the engine need to be calculated by either the propulsion or aerodynamic analysis tools. Ultimately, what is needed is a procedure for determining a single set of resultant aeropropulsive forces and moments that vary as a function of flight condition and engine throttle setting.

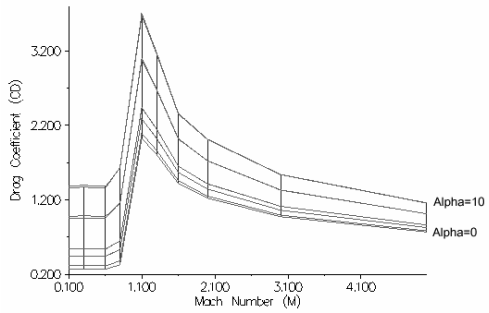

Figure 6: C_d for a Typical Wing-Body Vehicle as a Function of Mach Number as computed by Missile DATCOM

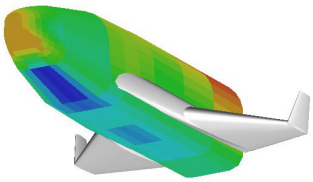

Figure 7: Pressure Distribution on a Fuselage as Predicted by PANAIR

Trajectory Simulation

The final discipline needed to "close" the design of a hypersonic vehicle is trajectory simulation. Trajectory simulation is used to determine and verify that the amount of fuel available in the vehicle is sufficient perform the desired mission (i.e., cruise a specified distance or accelerate to a desired staging point).

The trajectory simulation is used in an iterative process to determine the final, "closed", conceptual design. An iterative process is needed because the fuel fraction is a required input to the mass properties analysis. The vehicle closure process that will be used this hypersonic vehicle sizing application is:

1. Guess a required fuel fraction for the vehicle.
2. Specify a design for the propulsion flow path and OML of the hypersonic vehicle. (Note: both steps 1 and 2 are highly dependent on the design engineer's experience and personal preferences.)
3. Calculate the vehicle's mass properties based on the geometric model, the fuel fraction required and other mission parameters. (Remember that another iterative process is needed here to ensure that the MERs are consistent.)
4. Calculate the aeropropulsive forces on the vehicle for the flight conditions of interest.

5. Use a trajectory simulation code to calculate the fuel fraction that would be required for this vehicle to perform the desired mission.
6. Compare the results of step 5 with the guess from step 1 to determine if the fuel fraction required is consistent for this vehicle. If not, modify the vehicle's OML from step 2, adjust the guess of the fuel fraction required and repeat the process. (Note: depending on the type and magnitude of the changes to the OML, the aeropropulsive force calculations from step 4 may not need to be repeated.)

For this effort, two commonly used trajectory simulation and optimization codes, POST [13] and OTIS [14], will be considered. Both POST and OTIS have been integrated, to some level, in AML under previous efforts.

The methods that were used to integrate POST and OTIS exemplify the tradeoffs that are possible in all code integration problems. The main consideration that needs to be made when planning a code integration project is how much of the original functionality of the code will be available to the user of the integrated design application. Typically, as more functionality is made available in the integrated application, more discipline specific, or code specific, experience is required of the user. However, limiting the functionality of the integrated code usually also limits the range of designs that can be examined and limits the code reuse benefits of the object oriented programming paradigm.

Under previous efforts, POST was integrated using a "variant" approach, while the integration of OTIS was more comprehensive.

The variant approach required that a trajectory simulation engineer develop a complete simulation input file for the problem using existing procedures. Then, the engineer identified which parameters and data tables would change (and could change) when the baseline vehicle is redesigned. Finally, a method was developed to create a new input file for the simulation code by automatically changing the few parameters and tables that were previously identified.

A more comprehensive level of integration was developed for the OTIS 3.0 simulation code. The goal of this integration was to ensure that the complete functionality of the code was available in the integrated application. For OTIS, this also required the creation of a complex user interface. This is because the code has many settings available and these settings may be changed in each phase of the input. Note: phases can refer to changes in flight objective (e.g., minimum time climb or best altitude cruise) or changes in vehicle configuration (e.g., after launch vehicle staging or after weapons release).

Because of the complete level of integration of OTIS, it was expected that the main user of this portion of the integrated application would be an experienced OTIS user. For this reason, the developers chose to develop the user interface in pages that correspond to sections of the OTIS input file. A portion of the main user interface form is shown in Figure 8.

Furthermore, the AML programmers developed a basic online help interface for OTIS. The main feature of this help system is easy access to the definitions of the variables that are defined in OTIS. The developers have found that the cryptic abbreviations used by OTIS are one of the main concerns for new OTIS users. A sample of this help interface is shown in Figure 9.

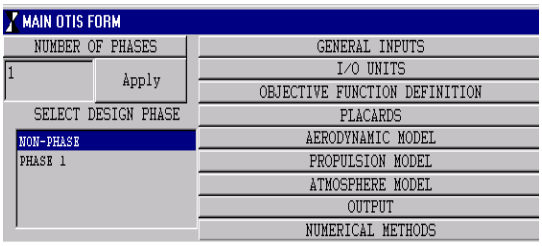

Figure 8: Main User Interface Form for OTIS in the Integrated Application

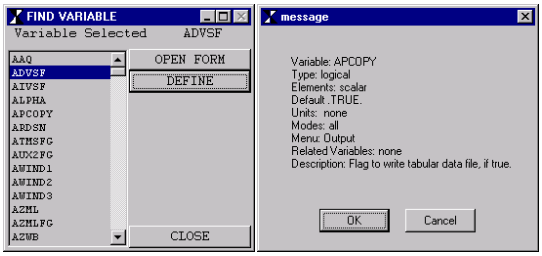

Figure 9: Help Screens for Defining OTIS Variables

The final significant part of the OTIS integration project was the development of a simple trajectory plotting object. This plotting object was developed mainly for use by the experienced OTIS user while debugging their setup of the trajectory simulation. A sample plot that was created using this capability is shown in Figure 10.

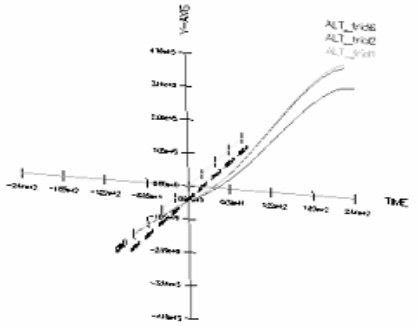

Figure 10: AML Trajectory Plotting Example

Based on their experience with these two integration efforts, the authors hope that a "happy medium" between the two approaches can be found. The POST integration is too limited and the OTIS integration exposes functions of the program that are very rarely, if ever, used for the simulation of hypersonic, airbreathing or rocket-powered launch or cruise vehicles. The team hopes to develop a user interface that is both robust and relatively easy to use by an engineer who is new to the trajectory simulation discipline.

INITIAL APPLICATION

The authors are currently working with two different hypersonic, airbreathing vehicle configurations. One configuration is a 2-D lifting body class vehicle, similar to NASA's X-43 (Hyper-X) configuration, shown in Figure 11. The second configuration is an axisymmetric, rocket-based combined cycle (RBCC) powered, SSTO vehicle, similar to the GTX configuration [15], shown in Figure 12.

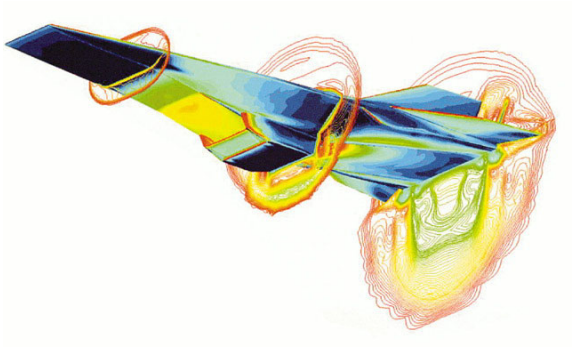

Figure 11: Hyper-X Configuration

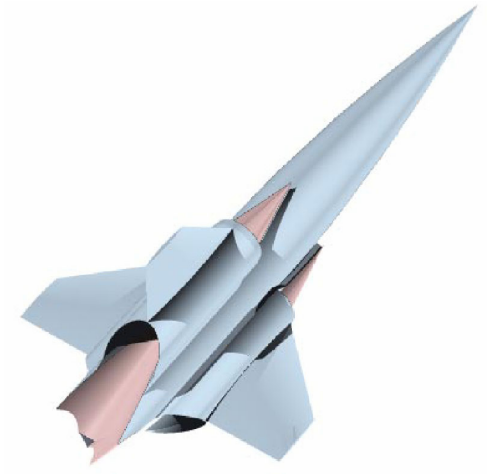

Figure 12: GTX Configuration

The geometric modeling efforts have been split between the two configurations. As mentioned above, many distinct OML modeling objects were developed for each configuration because they are such significantly different configurations. Also, the MERs that are integrated with the geometric objects will have to be tailored for each configuration.

For instance, the fuel tank models for each configuration will be significantly different. For the 2-D vehicle, the tanks will be conformal. That results in a more complex geometric model, even at the conceptual-level, and a MER which will estimate that the conformal tank is heavier than a cylindrical one for the same surface area, volume and/or pressure. However, for the axisymmetric vehicle, the fuel tanks will be mostly conical. These tanks can be modeled with simpler geometric objects and require MERs that predict a tank that weighs less than the conformal tank for similar design conditions.

For the other disciplines (i.e., propulsion flow path analysis, aerodynamic analysis and trajectory simulation), most of the author's effort to date has concentrated on the 2-D lifting body configuration. This decision was based on the author's previous experience in modeling these configurations as well as the availability of analysis codes and experimental results for configurations of this class.

ACKNOWLEDGEMENTS

Bob Pegg of the NASA Langley Research Center deserves recognition for his support of the AVS spreadsheet-based weights model. Additionally, the authors wish to thank Carrie Clewett of AFRL's Air Vehicles Directorate and Alicia Hartong & John Livingston of ASC's Air Vehicle Design branch for the concept and implementation of the AML-based weights model.

CONCLUSIONS

This effort has resulted in an integrated design modeling application for the conceptual-level design and analysis of hypersonic airbreathing vehicle configurations. Additionally, the authors believe that this application can form the basis of a more complex design environment that is capable of refining the design of hypersonic vehicles to the preliminary level.

To be called a preliminary-level design and analysis application, additional disciplines are needed. Specifically, two disciplines, aerothermal analysis and finite element structural analysis, are needed to provide physics-based mass properties information. Aerothermal analysis will provide the temperatures and heat fluxes to which the vehicle will be exposed during its mission. This information can then be used to size the thermal protection system of the vehicle, which makes up a significant portion of the total vehicle weight. Ongoing efforts [5] could be expanded to integrate aerothermal analysis tools into AML. Finite element methods (FEM) can be used to size the main structural components (including the tanks) based on the loads they will see during a mission. The weights of these size components can then be estimated more accurately. The integration of FEM into AML is being explored under a separate effort [16].

Another discipline that is encountered during preliminary design is flight control. The major tasks of this discipline are the development of flight control laws, sizing of the elements of the flight control system, and the development of a six degree-of-freedom (6DOF) model of the vehicle. Codes like MATLAB or MATRIXx are commonly used for these tasks and because they are modern software products, information can easily be linked between these programs an AML.

Finally, higher fidelity aerodynamic and propulsion analysis methods will be needed during the preliminary design process to generate the pressures, forces and moments to feed into the FEM and flight control disciplines.

REFERENCES

1. Tirpak, John A., "Mission to Mach 5", *Air Force Magazine: Journal of the Air Force Association*, Vol. 82, No. 1, January 1999.
2. Engelund, W. C., "Hyper-X Aerodynamics: The X-43A Airframe-Integrated Scramjet Propulsion Flight-Test Experiments", Journal of Spacecraft and Rockets, v. 38, no. 6, Nov-Dec 2001, pp 801-802.
3. Kowal, D., Zarda, P. R. and Baum, F. "Web-based Design and its Impact on the Engineer", presented at the 3rd International Conference on the Intelligent Processing and Manufacturing of Materials, Richmond, BC, CANADA, July 29-August 3, 2001.
4. Zweber, J. V., Stevenson, M. D., Bhungalia, A. A., Catron, S. J., Hartong, A. R. and Grandhi, R. V., "A Web-based Collaborative Application for Aerospace Vehicle Design and Analysis ", presented at the 3rd International Conference on the Intelligent Processing and Manufacturing of Materials, Richmond, BC, CANADA, July 29-August 3, 2001.

5. Bhungalia, A. A., Zweber, J. V. and Stevenson, M. D., "Thermal Protection System Analysis in an Integrated Design Environment for Launch Vehicles", AIAA paper 2002-0595, presented at the 40th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, January 14-17, 2002.
6. *Adaptive Modeling Language Reference Manual,* Version 3.1.3, TechnoSoft Inc., Cincinnati, OH, 1999.
7. The Adaptive Modeling Language. A Technical Perspective, TechnoSoft, Inc., Cincinnati, OH, http://www.technosoft.com/docs/brochure99.pdf
8. TechnoSoft, Inc. web site, http://www.technosoft.com/gallery/11.htm
9. Anderson, John D., Jr., *Modern Compressible Flow with Historical Perspective, 2nd Ed.,* McGraw-Hill, New York, 1990.
10. Pinckney, S. Z, Ferlemann, S. M., Mills, G. and Takashima, N., "Program Manual for SRGULL, Version 1.0: Second Generation Engineering Model for the Prediction of Airframe-Integrated Subsonic / Supersonic Combustion Ramjet Cycle Performance", NASA Langley Research Center Report HX-829, July 2000.
11. Blake, W., "Missile DATCOM Users Manual" AFRL-VA-WP-TR-1998-3009, February 1998.
12. Public Domain Aeronautical Software web site, http://www.pdas.com/panair.htm
13. Powell, R. W., Striepe, S. A., Desani, P. N., Braun, R. D., Brauer, G. L., Cornick, D. E., Olson, D. W., Peterson, F. M. and Stevenson, R., "Program To Optimize Simulation Trajectories (POST): Volume II, Version 5.2, Utilization Manual", NASA Langley Research Center, Hampton, VA, October 1997.
14. Paris, S. W. and Hargraves, C. R., *OTIS 3.0 Manual*, Boeing Space and Defense Group, Seattle, WA, 1996.
15. Trefny, C., "An Air-Breathing Launch Vehicle Concept for Single-Stage-to-Orbit", AIAA paper 99-2730, presented at the 35th AIAA / ASME / SAE / ASEE Joint Propulsion Conference and Exhibit, Los Angeles, CA, June 20–23, 1999.
16. Blair, M. and Canfield, R. A., "A Joined-Wing Structural Weight Modeling Study", AIAA paper 2002-1337, presented at the 43rd AIAA / ASME / ASCE/ AHS/ ASC Structures, Structural Dynamics & Materials Conference, Denver, CO, April 22-25, 2002.