# Vehicle Sketch Pad: A Parametric Geometry Modeler for Conceptual Aircraft Design

Andrew S. Hahn[1]
*NASA Langley Research Center, Hampton, VA, 23681*

**The conceptual aircraft designer is faced with a dilemma, how to strike the best balance between productivity and fidelity? Historically, handbook methods have required only the coarsest of geometric parameterizations in order to perform analysis. Increasingly, there has been a drive to upgrade analysis methods, but these require considerably more precise and detailed geometry. Attempts have been made to use computer-aided design packages to fill this void, but their cost and steep learning curve have made them unwieldy at best. Vehicle Sketch Pad (VSP) has been developed over several years to better fill this void. While no substitute for the full feature set of computer-aided design packages, VSP allows even novices to quickly become proficient in defining three-dimensional, watertight aircraft geometries that are adequate for producing multi-disciplinary meta-models for higher order analysis methods, wind tunnel and display models, as well as a starting point for animation models. This paper will give an overview of the development and future course of VSP.**

## Nomenclature

*ACSYNT* = aircraft synthesis design program
*ASCII* = American Standard Code for Information Interchange
*CAD* = computer-aided design
*FLOPS* = flight optimization system design program
*GUI* = graphical user interface
*HWB* = hybrid wing-body aircraft configuration
*RAM* = Rapid Airplane Modeler program
*STOL* = short takeoff and landing
*VSP* = Vehicle Sketch Pad program

## I. Introduction

THERE are three basic phases in aircraft design. They are: conceptual, preliminary, and detailed. Each design phase has characteristics that drive the tools and methods used as the design progresses. While it may be desirable to have the same suite of tools and methods spanning the design process, the widely varying characteristics of each of the phases makes this extremely difficult. For those organizations whose activities span all of these phases, there is a strong desire to have tools and methods that also span all of the phases, and this is most evident in the area of geometry definition. Many of the largest airframers have chosen to utilize computer-aided design (CAD) tools from the very start and use them all the way through to actual production. There are many benefits claimed for this approach, such as automated documentation of changes, reduction of errors, and improvements in communication with suppliers. It may also be argued that while this approach may be less optimal for any single design phase, the benefits over the lifecycle are worth the detriments in any individual phase. It is a matter of judgment as to whether this is true, but in those organizations whose activities do not span all of the design phases, this is certainly not true.

The conceptual design phase has characteristics that make the use of CAD for geometry definition a non-optimal choice. Many of the strengths of CAD that make it beneficial in the preliminary and detailed design phases actually become either of little benefit or even hindrances in the conceptual design phase.

After careful consideration of the available options, the Aeronautics Systems Analysis Branch at NASA Langley Research Center decided to embark on the development of a clean sheet approach to the problem of defining aircraft

[1] Aerospace Engineer, Aeronautics Systems Analysis Branch, Mail Stop 442, AIAA Member 193989.

geometries for use in the conceptual design phase. This paper highlights the development of Vehicle Sketch Pad (VSP), a geometry modeling program specifically designed for use in the aircraft conceptual design phase.

## II. Conceptual Design Phase Characteristics

The conceptual design phase is the earliest, and is in many ways the inverse of the detailed design phase. Conceptual design is very fluid, requiring numerous large geometry changes in order to explore a broad design space. In order to explore this space, the designer will need to interact with the model many times manually, as well as perhaps employ the use of optimization in order to quickly develop designs for comparison. Because of this, it is very important that the geometry be created and modified easily, with a minimum of parameters, in ways that do not yield nonsensical results.

Also, the conceptual design phase has the least amount of information available, making it difficult to specify many aspects of the geometry in great detail. Frequently, only basic geometry is publicly available for existing aircraft in a particular class of interest, requiring the designers to use their experience in filling in information gaps when building validation models. Moreover, generation of the information necessary to define many details can be very time-consuming.

Another characteristic of conceptual design is that the main goal is to perform analysis of alternatives in order to choose the best combination of configuration and technologies available. Because these choices generally have very large impacts on the design, the impacts of finer details are not discriminators. Spending a great deal of time or effort on characteristics that will not substantially change the outcome is not productive. The tools and methods generally used in this phase have a similar effect in that they do not operate on the high fidelity model directly, but instead require extraction of a meta-model, which is a new geometry of reduced complexity derived from the original. The very act of creating the meta-model tends to eliminate details that the analysis method is insensitive to, while retaining those details that it is sensitive to. This process tends to eliminate the impact of most fine details.

Generally, the characteristics of the conceptual design phase require that modifying the geometry be easily accomplished, the geometry be amenable to optimization, and that the geometry has sufficient fidelity to perform analysis of alternatives, but does not need the high fidelity necessary in later phases of the design process.

## III. Conceptual Design Geometry Options

At the outset, a small number of systems analysts came together to identify the requirements of a conceptual design geometry modeler and to review the available options for providing the desired capability. All of the options have pros and cons associated with them, making the final decision a matter of judgment and consensus. Different organizations may come up with different decisions, but the following is a summary of the thinking of this one group.

At the time, there were three basic options available. The first was to continue using text input. The second was to transition into using CAD software. The third was to develop custom modeling software in-house.

### A. Text Input

The standard method of defining geometry for conceptual design at the time was to have a parameter list as part of the input to a monolithic aircraft design synthesis program. This was what was used in both the ACSYNT and FLOPS programs (Fig. 1). This worked well given that the methods used at the time were essentially extensions of manual handbook methods. These methods were sensitive

```
#########################################################################
#                                                                       #
#                          GEOMETRY NAMELISTS                           #
#                                                                       #
##0#################0#################0#################0##############

 $WING
  AR=5.01           AREA=231.77,      DIHED=2.5         FDENWG=48.6163,
  LFLAPC=0.00,      SWEEP=13.0,       SWFACT=1.2576,    TAPER=0.507,
  TCROOT=0.09,      TCTIP=0.09,       TFLAPC=0.3,       WFFRAC=0.8631,
  XWING=0.4632,     ZROOT=-0.6095,    KSWEEP=1,
 $END
 $HTAIL
  AREA=54.0,        CVHT=0.635,       HTFRAC=-0.35,     SIZIT=F,
  AR=4.0,           SWEEP=25.0,       SWFACT=0.9954,    TAPER=0.469,
  TCROOT=0.08,      TCTIP=0.08,       XHTAIL=0.6,       ZROOT=2.00,
  KSWEEP=1,
 $END
 $VTAIL
  AREA=37.373,      CVVT=0.073,       VTFRAC=-0.35,     SIZIT=F,
  AR=0.798,         SWEEP=39.5,       SWFACT=1.9356,    TAPER=0.493,
  TCROOT=0.10,      TCTIP=0.10,       XVTAIL=1.0,       YROOT=0.0,
  ZROOT=0.5048,     KSWEEP=1,         VTNO=1.0,
 $END
 $FPOD
  X=0.563,          SOD=0.77,         THETA=15.0,       SYMCOD=0,
 $END
 $FUS
  BDMAX=5.25,       BODL=40.714,      DRADAR=1.0,       FRAB=3.9,
  FRN=1.9,          LRADAR=1.0,       SFFACT=0.8588,    WALL=0.2,
  WFUEL=5395.0,     ITAIL=1,
 $END
```

**Figure 1. ACSYNT example of text input.**

only to the most fundamental characteristics of the geometry, and to a great extent relied on the assumption that the final design was very similar to most of the designs that came before it, and that the myriad of detailed design decisions would be made at least as well as they had in the past. While computerization of these handbook methods was a major advance in terms of productivity, and application of optimization allowed for more consistent design in a multi-disciplinary and multi-dimensional design space, the underlying analysis methods were not well suited for the conceptual design of significantly different configurations. What was desired was a way of bringing higher order analysis results that are sensitive to new configurations into the synthesis process. This required both a greater degree of flexibility in defining geometry and the creation of meta-models from the geometry, catering to the needs of each higher order analysis. Doing so within the large, monolithic synthesis programs was thought to be undesirable. This was because the ability of the designer to both manage and visualize the design being described by the parameter list was already at its limit.

Early attempts to add automated graphics for visualization to these synthesis programs almost seem comical by today's standard. The earliest attempt was to produce top and side views using only ASCII characters. As primitive as this was, it was actually useful because large errors were caught easily. Still, smaller errors would go unnoticed. Understandably, these graphics were not useful for communicating with management or colleagues and were not shown outside of the Aeronautics Systems Analysis Branch. Unfortunately, more precise and presentable graphics required manual generation, which meant that they were done infrequently and usually at the end of the design process. These took the form of manually drawn three view drawings and artist's renderings. There was an attempt to do automated vector graphics, and it was quite a bit better than the old ASCII characters, but it was still a two-dimensional representation of a three-dimensional geometry.

In the end, it was felt that adding more parameters to the list would have made these programs too difficult to use and that a true three-dimensional capability was desirable.

### B. Computer Aided Design

The second option was the use of computer aided design (CAD) programs. These programs came out of the manufacturing world. Because of this, they have certain characteristics that make them well suited for manufacturing. These include very high precision, extreme flexibility, and the ability to easily represent machining operations. In general, the geometries are represented as surfaces defined by common primitive operations such as extrusion or radiusing. Given that the outer mold line of a typical aircraft is not produced by these processes, it takes a skilled user to produce aircraft geometries with CAD. Also, the CAD operations used to define the geometry internally are not similar to the way that aircraft designers define geometry. This requires either the operator or a programmer to translate traditional aircraft design parameters into operations native to CAD.

Given the resource realities of the Aeronautics Systems Analysis Branch, it was unrealistic to expect to have a skilled CAD operator on staff. Also, geometry description is done sporadically, with many changes in very short periods of time. Not only is there a time issue whenever manual intervention is required, but the learning curve of CAD is steep. While putting the effort in to learn CAD is acceptable for a specialist, the sporadic nature of conceptual design precluded the designers from becoming proficient with CAD, and having to repay some of the learning curve every time a new study was started was undesirable. It was felt that an automated process would be a much better fit to our requirements.

At the time, parametrically defined CAD interfaces were being developed. At first this seems like an attractive option. By programming these parametric interfaces, it might be possible to get the best of all worlds. It would be possible to specify the geometry in familiar terms, to have a limited number of parameters necessary for both productivity and optimization, to not have the time delay inherent with manual operation, to have the precision of industrial-strength CAD, and to have the benefit of sharing development costs across a large number of users. After closer examination, it was the consensus of the group that programming the interfaces would be nearly as difficult as programming a new custom geometry modeling program. In particular, it seemed very difficult to program generic parameterization and that there would be a constant need to develop new parameterizations for each new design. The time lag associated with having to program the new parameterizations would be unacceptable.

Also, many of the benefits of CAD really were not beneficial to conceptual design. For example, a very high degree of precision, which is a necessity in manufacturing, is not generally beneficial to conceptual design. This means that, while development costs were being spread across many users, most of the development focus and features were not actually useful in conceptual design, and that the cost of licensing CAD was not that much lower than developing a custom, in-house geometry modeler. Last, the main benefit of using CAD for airframers in conceptual design appears to be the ability to carry the geometry through the entire design process, trading detriments in individual phases for benefits across all phases. Since the Aeronautics Systems Analysis Branch mission does not span the entire design process, there is no opportunity to take advantage of this synergy.

### C. Parameterized GUI Geometry Modeler

The third option was to develop a custom geometry modeler. The main benefit of developing our own geometry modeler is that it can be made to address conceptual design geometry requirements specifically. What was needed was a tool that allowed for greatly expanded parameterization and model visualization as well as meta-model export to analysis methods and other software to facilitate testing and communication. It was felt that a parameterization along the lines of the lists used by synthesis programs was a good start, but it became so difficult to manage that some other form of interaction was needed instead of a simple text listing. A relatively simple graphical user interface (GUI) using forms and a three-dimensional interactive display was developed in order to prove the utility of this approach. This was the original Rapid Airplane Modeler (RAM). RAM showed that it was possible to provide many more parameters in a user-friendly environment, greatly enhancing the designer's ability to manage the definition of a model and to visualize the results of that definition. It also demonstrated a limited capability to provide export of the geometry in formats that were useful in creating meta-models for specific analysis methods.

Because the parameterization was based mostly on traditional aeronautical concepts, experience showed that the learning curve was very short, and more importantly, the relearning curve was extremely short. This meant that a designer could quickly become proficient with minimal effort, and then could come back after months of inactivity and be nearly as proficient as before.

Because a great deal of thought had been put into the expanded parameterization, it was both deceptively simple and unusually flexible. This meant that the parameterization did not have to be customized for every new configuration, making constant programming changes unnecessary and making the program useful immediately.

Also, we felt that the ability to create completely arbitrary shapes is not only not necessary, but somewhat problematic. For example, aerodynamics is sensitive to the second derivative of the curve of the skin. One of the features inherent in this parameterization and implementation is that the shapes produced are smooth and continuous to the second derivative. This is both liberating and limiting. Since the general desire of the designer is to produce aerodynamic vehicles, the result of manipulating the parameters always results in aerodynamic shapes. Of course, this meant that modeling non-aerodynamic shapes was not natural to the program. While it is possible to force non-aerodynamic shapes to be modeled, it is more difficult and generally sacrifices the benefits of parameterization.

There are drawbacks to developing a custom geometry modeling program. The most obvious is that all of the cost of development is borne by the Aeronautics Systems Analysis Branch and the programs that we serve. While this development cost has been significant in aggregate, it has actually been modest per year and has been spread out over several years. Also, cost per user has been high in the past because the initial user base was very small. However, with the improvements that are in our more advanced modeler, the user base has grown dramatically, greatly reducing the cost per user.

Another drawback is that this kind of modeler is no substitute for the capabilities in CAD. It is designed to work best in the conceptual design phase. While the modeler may be extended to where it may be useful in the preliminary design phase, the architecture makes it unsuitable for the detailed design phase. In order to carry a design through the entire design cycle, the model would have to be completely reworked in a CAD program. This is potentially a significant cost, but it is not entirely clear that this would make a conceptual design geometry modeler unattractive, even for organizations whose mission does span the entire design process. The reason is that this kind of geometry modeler is so efficient and productive, that it is very attractive during the conceptual design phase. By the time that the preliminary design phase begins, the geometry has settled down to the point where major changes are no longer likely to occur. Instead of manually modeling many disparate configurations during conceptual design in CAD, this expensive re-model need only be done once. It is a matter of judgment and preference by the organization as to what course makes the most sense.

Given the very positive experience with RAM, but also recognizing its limitations, a new aircraft conceptual design geometry modeler was developed that built upon the positive aspects of RAM, while greatly extending its capabilities. That modeler is called Vehicle Sketch Pad.

## IV. Vehicle Sketch Pad

Vehicle Sketch Pad (VSP) is a new implementation of a parameterized GUI geometry modeler that leveraged the experience of developing RAM (Fig. 2). The geometry components and their parameterizations have been significantly upgraded along with much more sophisticated export and meta-model creation capability. VSP has many of the same characteristics in that it builds a text file that is filled out through a series of forms presented graphically to the user, and the geometry is displayed in real-time in a three-dimensional display window.

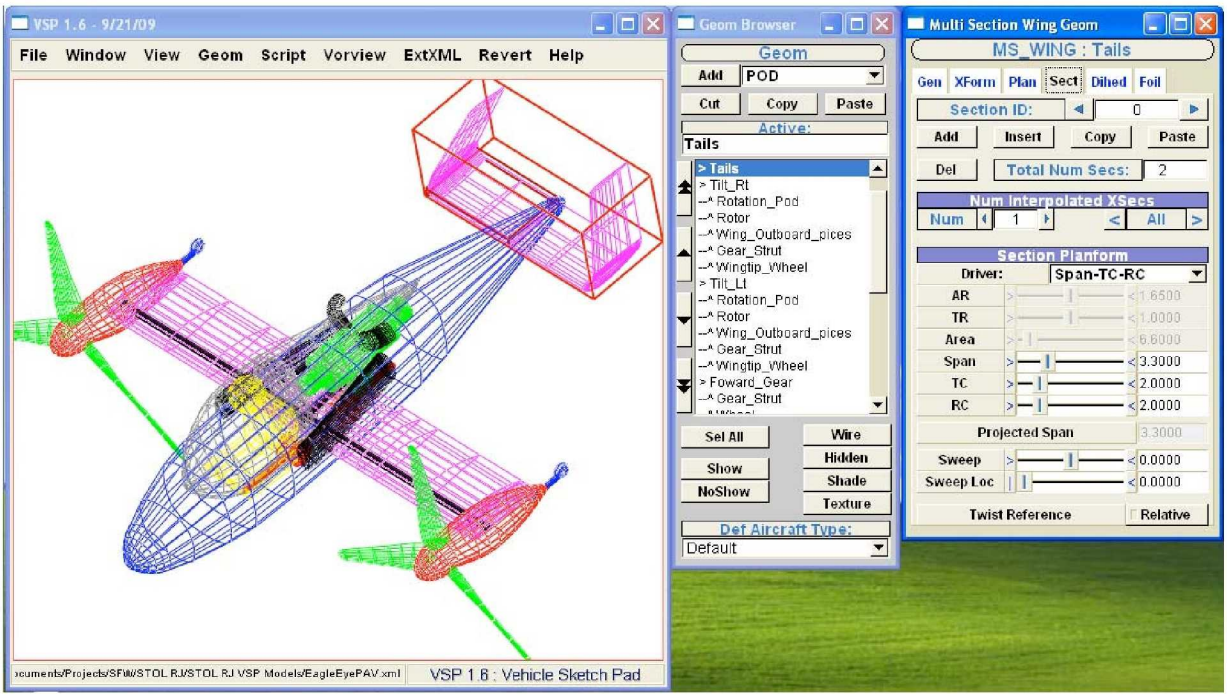


**Figure 2. Vehicle Sketch Pad example of parameterized GUI.**

## A. Flexible

While the parameterization has gotten considerably more extensive, losing some of the simplicity of the original RAM, care has been taken to present forms, which group parameters in a logical way, to facilitate typical workflows while hiding complexity when it is not needed. User interface techniques, such as the use of driver groups, help the designer by allowing input of whatever data is available and calculating dependent data internally (Fig. 3). This reduces the need for off-line calculation in order to translate from data that the designer does have into parameters that the modeler has available. A subtle use of the driver group feature is that the designer can change the group, specifying which independent variables will be held constant while changing the geometry. An example of this is in the wing section form. For the trapezoidal section, there are six parameters available to define the section, only three of which are required to uniquely define the section. The use of driver groups allows the designer to choose which three are to be independent, leaving the remaining three to be dependent.

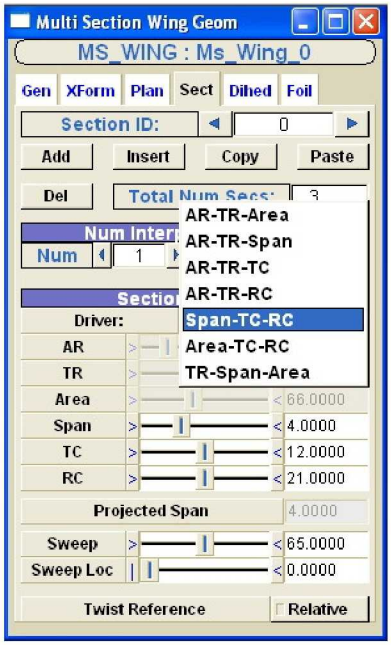


**Figure 3. VSP driver groups.**

An example of how VSP's deceptively simple parameterization can be used effectively is in the area of defining high-lift devices. The usual practice is to provide airfoil coordinates on a scale of zero to one chord, with the individual element chord line at Y=0. Doing this for all of the elements in a high-lift system would require the designer to scale, rotate, and translate each element independently of each other. It is very difficult to perform these operations while maintaining the proper relationship between the elements and if the cruise wing geometry is altered, then the designer would have to determine off-line what the new scales, rotations, and translations would have to be. Instead of this convention, VSP allows for arbitrary airfoil coordinates in relation to the traditional zero to one chord reference (Fig. 4). This means that the cruise configuration wing can be defined, and if the high-lift configuration airfoils are defined properly, then multiple copies of the cruise wing only have to differ by the airfoil file chosen in order to correctly size and position all of the elements. Any other high-lift flap settings need only to

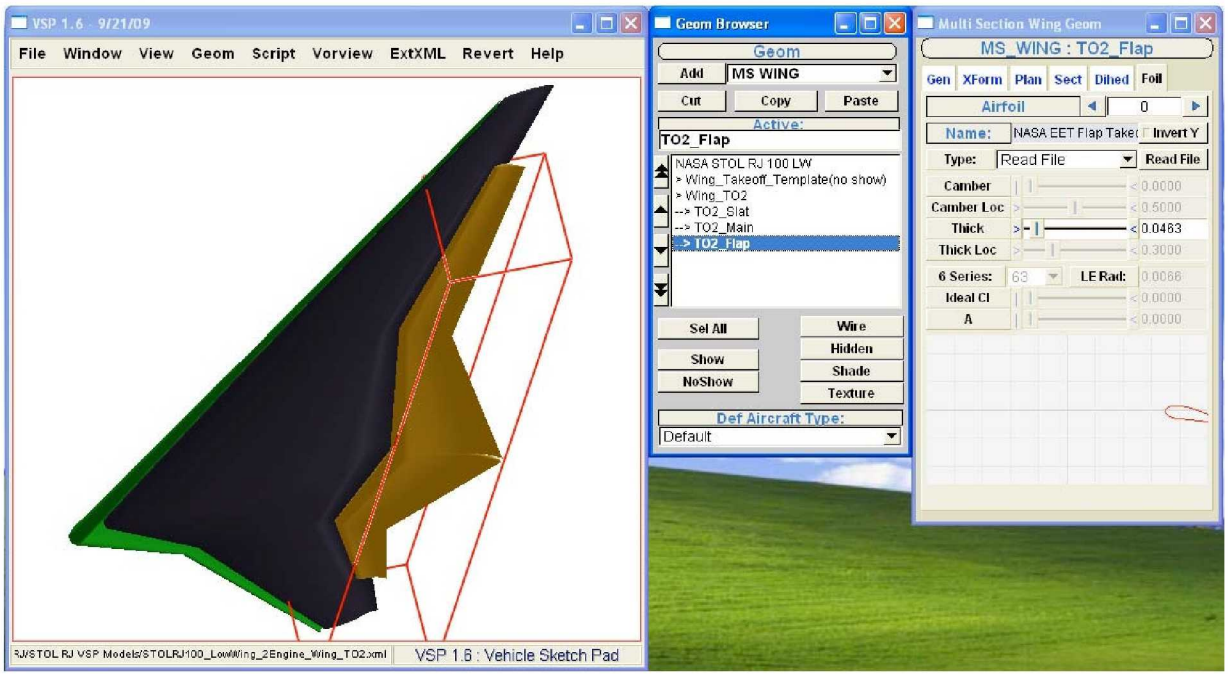


**Figure 4. VSP example of dependent high-lift flap elements.**

have different airfoil files read in to define them as well. The accuracy of the positioning is only dependent upon the accuracy of the airfoil file. If for any reason the cruise wing geometry is altered, then all that is needed is for the copies to be updated to the same values. While this update must be performed manually, it is relatively simple to do. A future feature may be to automate this process by allowing the designer to specify dependencies down to the individual parameter level. Currently, parent-child dependencies only extend to position and rotation.

VSP has some capability to do arbitrary shapes. In general, this is to be avoided because it negates the main advantage that VSP has over CAD, namely parametric input and the ease with which the geometry can be modified. Still, the capability is there when needed. Arbitrary fuselage cross-sections and wing airfoil sections may be read in from files. Manipulation is limited to scaling the height and width of these sections. Also, an unlimited number of hard cross-sections may be defined at any point along the fuselage in order to ensure that the surface goes through specific points.

### B. Real-time Interactive

The VSP user interface has both slider bars for real-time approximate parameter input as well as numeric input. The slider bar interface allows for the designer to change parameters while watching the 3-D display window, allowing for interactive and dynamic design in a trial and error way. Frequently, this is used to get parameters near to their final values, but the lack of precision may not be what's desired. To compensate for this, numeric input is also available for parameters. Any change is reflected immediately upon entry, still giving a degree of real-time interaction, while also giving a fine degree of control. This real-time capability can be used to quickly check options in "what if" scenarios and experience has shown it is particularly useful when positioning and sizing internal subsystems.

### C. Batch Operable

While VSP is primarily intended to be interactive with the designer, it does have the ability to run in batch mode. This enables VSP to be interfaced to other programs without the need for manual intervention. This has been very useful when running optimization. The design parameters are exposed so that a control program may change them at will and then take advantage of the services that VSP provides, such as calculation of trimmed and untrimmed wetted areas and volumes. Given that changing a small number of parameters may yield very large changes in the geometry, VSP is particularly well suited for manipulation by gradient methods or genetic algorithms. While this may lack the flexibility offered by other optimization schemes like adjoint methods that can perturb individual points in the mesh, it is in keeping with the capability desired in the conceptual design phase.

### D. Portable

VSP was initially developed with portability in mind. Early releases were made for both Windows and Linux platforms. The native file format is XML text, which, while optimized for machine parsing, is human readable and editable. Because it is an ASCII text file, it is not subject to issues concerning reading binary files on different platforms. Also, because the file is a list of parametric values, it tends to be very compact. Transfer of files through e-mail, regardless of complexity, is extremely easy. Generally the things that tend to drive up file size are external cross-section or airfoil section files that have been read in and these are completely under the control of the designer.

Currently, only development of the Windows version is ongoing. This is because there were no Linux users of VSP. It is still possible to port VSP to Linux or MacOS, but as of now the small user base that would need this does not justify continuing these developments, particularly because there are Windows emulators available and VSP works very well in these environments.

## V. Application of Vehicle Sketch Pad

While visualization of the geometry is important, it is of limited utility. The real utility of having an easy-to-use higher order geometry modeler is in being able to export information and meta-models to other analysis methods and modelers.

Initially, VSP was used simply to provide wetted areas and volumes so that tuning factors could be adjusted in synthesis codes, such as ACSYNT and FLOPS, in order to correct the more basic internal estimates. While this is still done, it is increasingly more common to supplant the internal estimates with VSP estimates by linking to the synthesis code in batch mode through a software integration framework, such as Phoenix Integration's ModelCenter®.

VSP has also been used to check fit, form, and function of subsystems in a variety of designs. Examples include packaging studies for high-altitude long-endurance aircraft (Fig. 5), a MarsPlane project (Fig. 6), and airliner landing gear operation with clearance constraint checks for rotation and collapsed nose gear conditions (Fig. 7).

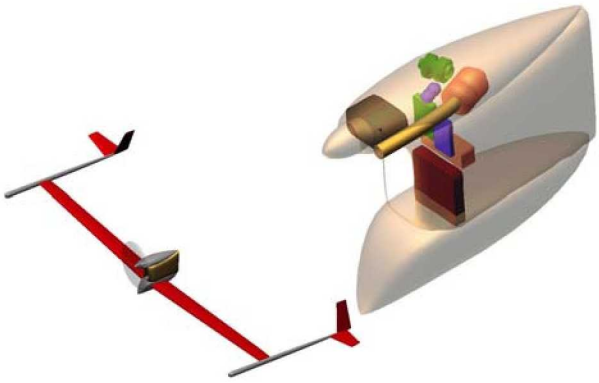

**Figure 5. High-altitude UAV systems.**

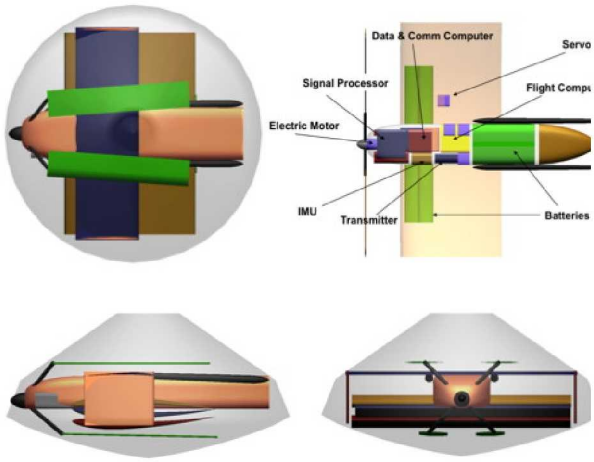


**Figure 6. MarsPlane packaging and systems.**

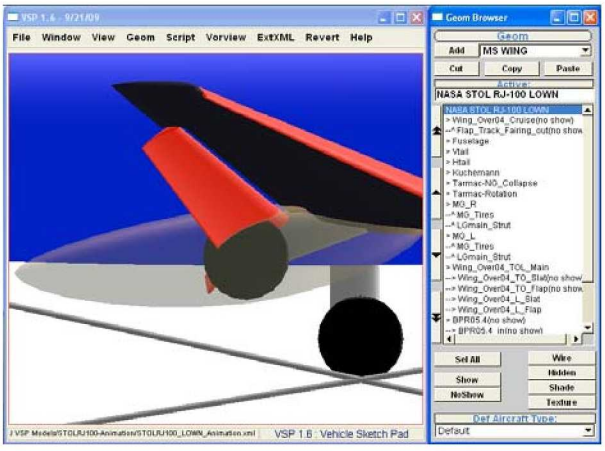


**Figure 7. Landing gear fit, function, constraints.**

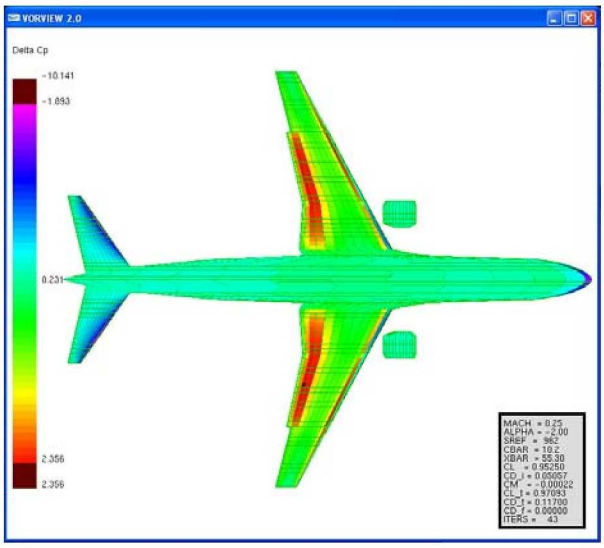


**Figure 8. VorView/VorLax high-lift analysis.**

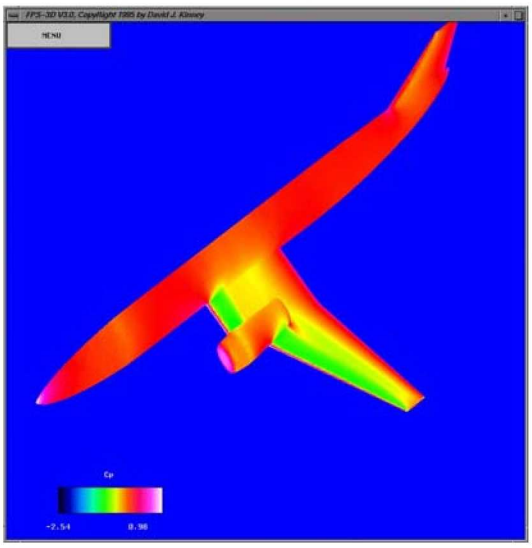


**Figure 9. FPS3D full potential flow analysis.**

As VSP developed, new capabilities to provide higher order analysis methods with meta-models were added. The earliest was the ability to output the averaged cross-sectional area distribution of the VSP model in a form that could be used directly by the AWAVE supersonic compressibility drag analysis program. This particular effort was interesting in that a great deal of effort was expended trying to create a Craidon format meta-model so that AWAVE could then create the averaged cross-sectional area distribution itself. This proved to be extremely difficult, and in exasperation the current method was tried and found to be both easier to implement and of higher fidelity.

Next, a file type referred to as Hermite was added to VSP's export capability, which was then used in VorView to create a meta-model compatible with VorLax, a well-known vortex lattice flow solver. While it is possible to run VorView in batch mode to generate the meta-model for VorLax automatically, this only works for simple geometries. For more complex geometries a hybrid approach is possible. If the designer understands the way that VorView works, it is possible to manually define certain characteristics on an initial model, and then have VorView automatically create similar meta-models after that. While not perfect, certain tasks such as sweep optimization may be performed effectively. An example of a recent use of these codes is that of a high lift aerodynamic assessment of a STOL regional jet design (Fig. 8).

An early example of exporting VSP geometry to computational fluid dynamics flow solvers is that of a jet airliner whose engines are mounted above the wing. This required that VSP be able to export geometry in a form needed by the FELISA surface and volume grid generation software. Support code used the exported geometry and default sourcing scheme to automatically

**Figure 10. Rapid prototyped wind tunnel model.**

**Figure 11. Gridlock Commuter display model.**

create intersection curves, surface grids, and volume grids for both a full potential flow solver (FPS3D) and an Euler flow solver (MUSEC) (Fig. 9). At the time, the productivity of this suite of tools was impressive. Literally, the designer could start with no geometry at all in the morning and be ready to submit full configurations for solution by the end of the day. Because of the high degree of automation, the majority of the day was actually spent waiting for the gridding to complete.

A stereolithography format (STL) was added to VSP's export capability to support rapid prototyping. This capability was exercised by using stereolithography to build a wind tunnel model for use in a small, low-cost test. We produced a very nice model; however, in our inexperience with this technology we had neglected to specify that the body should be made hollow, creating a model that was unfortunately too heavy for the force balance (Fig. 10). The stereolithography export found much more success when used to create display models for communication. This was especially true when models were modified by sculpting software to have cut outs and interiors (Figs. 11,12).

As the number of export formats grew, it became increasingly difficult to maintain them. After surveying the available options, we decided that it would be better to use a commercially available, low cost, relatively easy-to-use CAD program named Rhino3D® to handle the exploding number of export file formats. The native .3dm file format for Rhino3D® is now the preferred export format, because Rino3D®'s input/output capability is extensive, relieving us of this maintenance burden. This has proven to be a good choice because .3dm support in other programs appears to be getting better and there are some operations that can be done easily in Rhino3D® to prepare the geometry for other higher order analysis methods. Examples that have used this export capability are animations to communicate very complex and unusual vehicle concepts (Figs. 13,14,15), and the full viscous Navier-Stokes aerodynamic analysis using the USM3D flow solver at transonic flight speeds (Fig. 16).

In a recent development, internally generated high-quality triangulated surface meshes are being used with the Euler flow solver Cart3D on supersonic aircraft designs (Fig. 17). These supersonic configurations are being optimized for sonic boom mitigation while being constrained for equal volume and lift without increasing the drag of a baseline configuration.

## VI. Current Development

VSP has recently experienced a significant increase in development activity. There are multiple efforts underway to improve modeling and export capabilities.

**Figure 12. Civetta display model.**

**Figure 13. Dos Samara animation.**

**Figure 14. Tazenflugel animation.**

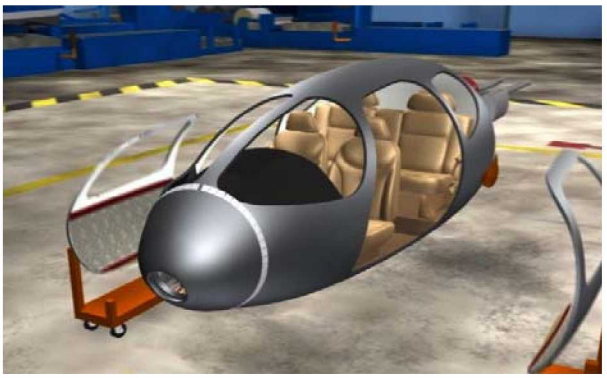

**Figure 15. Civetta assembly animation.**

The first is that Phoenix Integration has adapted both ModelCenter® and VSP to have compatible plug-in architectures (Fig. 18). This is far superior to the previous implementation because it removes bottlenecks that slowed integration and execution performance. In addition, designers will be able to monitor progress during optimization by having a VSP 3D display window embedded in the ModelCenter® process window.

The second development is that VSP now has the ability to generate high quality triangulated surface meshes automatically and internally for both aerodynamic and structural high order analysis methods (Fig. 19). Historically, VSP has been dependent upon other tools that work upon exported geometry. This has resulted in some loss of fidelity as files were manipulated, translated, and approximated. Also, most of these external tools were manual, significantly reducing productivity. VSP's surface grids are generated fully automatically and very quickly, greatly improving productivity, particularly when teamed with unstructured aerodynamic panel programs (e.g., CBaero), fully automated aerodynamic volume gridding and solution (e.g., Cart3D), and finite element structures programs (e.g., Nastran, Calculix).

The third development is a new capability to specify internal structural members (Fig. 20). The interface follows much the same philosophy as the rest of VSP. It uses a flexible parameterization that strikes a balance between ease-of-use and fidelity. While no replacement for fully generalized tools that are now available, this parameterization is flexible enough to define credible internal structures more easily than most other tools. This capability, teamed with the automated grid generation will provide a quantum leap in analysis productivity. While we have exported outer mold line geometry before, the analysis was performed on a pure monocoque structure, which was not representative of the actual structure.

The fourth development was a test case to see if VSP geometry could be automatically converted to a meta-model suitable for use in the aerodynamic panel program PMARC (Fig. 21). Initial results are encouraging for the multi-section wing component, which is appropriate for analyzing the hybrid wing-body (HWB) configuration. There is no plan to pursue this development for generalized configurations because it becomes extremely difficult to automatically panel intersecting bodies. Still, considering the ease with which one can analyze an HWB, which is also a configuration of high interest, this is a significant development.

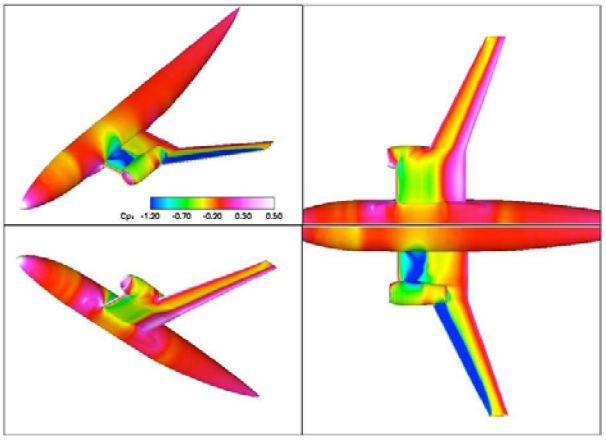

**Figure 16. USM3D Navier-Stokes flow analysis.**

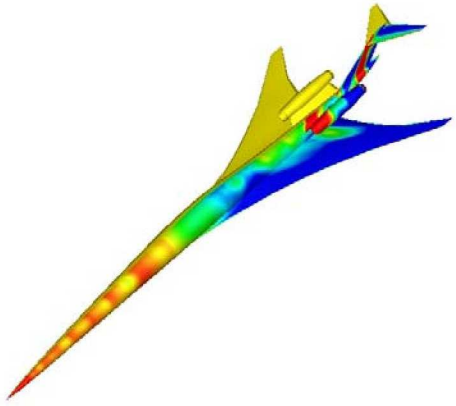

**Figure 17. Cart3D Euler supersonic flow analysis.**

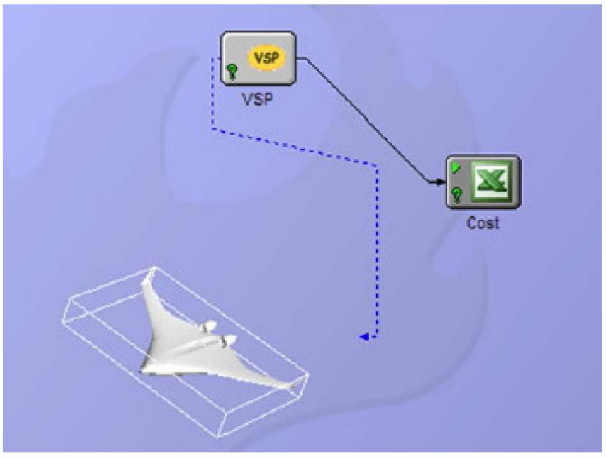


**Figure 18. VSP plug-in for ModelCenter.**

The last major development also relates to the HWB. Until recently, the HWB has been so different from traditional aircraft configurations, that our confidence in the analysis methods has been low. There has been a major investment in multiple disciplines to upgrade analysis

tools and methods to better estimate HWB characteristics. VSP is one of those tools, and a new parameterization allows much easier definition of typical HWB geometries.

## VII. Future Development

Despite the recent major increase in development, there is always a desire to do more. Certainly, the capabilities currently being developed will spur follow-on efforts. For example, the internal structural layout currently can only define elements that are full-depth (ribs, spars, bulkheads). Adding partial depth elements (formers, stringers, stiffeners) is desired. Also, adding an automated capability to define wakes for unstructured panel codes would improve ease-of-use and productivity with those analysis programs. Finally, there has been a long held desire to have Hermite file format read. While manipulation of this fixed geometry would be very limited, there are many possible uses for this capability. Right now, there is only a rudimentary capability and a much more general implementation would be of significant benefit.

Of course, the more that VSP is expanded and used, the more software bugs and interface issues will surface. VSP is not a commercial shrink-wrapped application, and so quality control is ad hoc. Still problems are tracked once identified and corrected as resources permit.

## VIII. Conclusion

Vehicle Sketch Pad is a parametric geometry modeler with a graphical user interface, designed to work best in the conceptual design phase. Ease-of-use and productivity were emphasized over the ability to produce completely arbitrary geometries with very high precision. Over time, the ability to export geometry to other geometry modelers and analysis programs has increased, making defining a geometry in VSP much more useful for design, analysis, and communication.


## Acknowledgments

The development of Vehicle Sketch Pad has been a team effort spanning many years. The list of contributors would be quite long if it were all-inclusive. Still, I would like to acknowledge the efforts of the principle author, J. R. Gloudemans for sticking with us through thick and thin, but mostly thin. I would also like to acknowledge Mark D. Moore for never giving up and keeping the embers glowing.


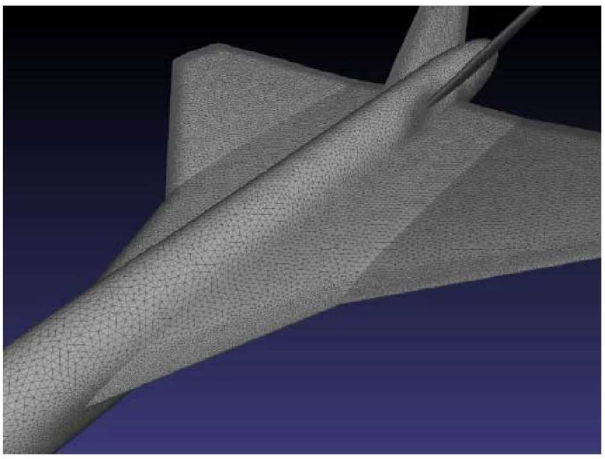

**Figure 19. High quality surface triangulation.**

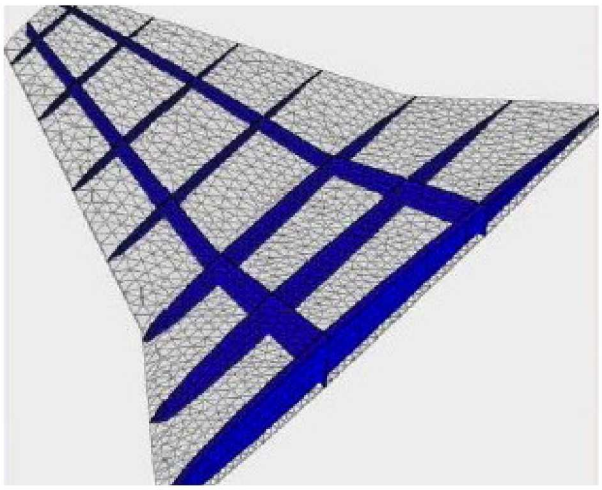

**Figure 20. Internal structure layout.**

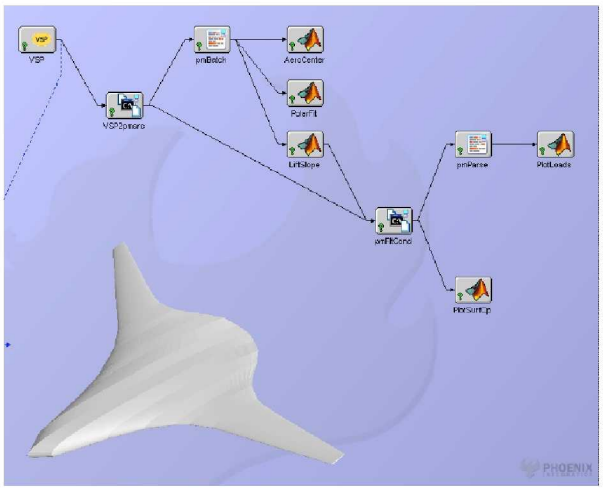


**Figure 21. PMARC meta-model generation.**