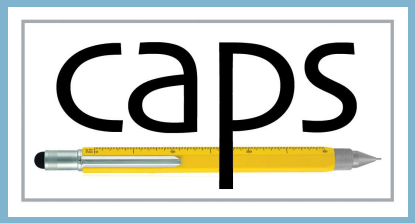
ESP/CAPS Training



Aircraft Design through Analysis

Some Thoughts, Observations and an Implementation

Bob Haimes haimes@mit.edu

Marshall Galbraith galbramc@mit.edu

Massachusetts Institute of Technology

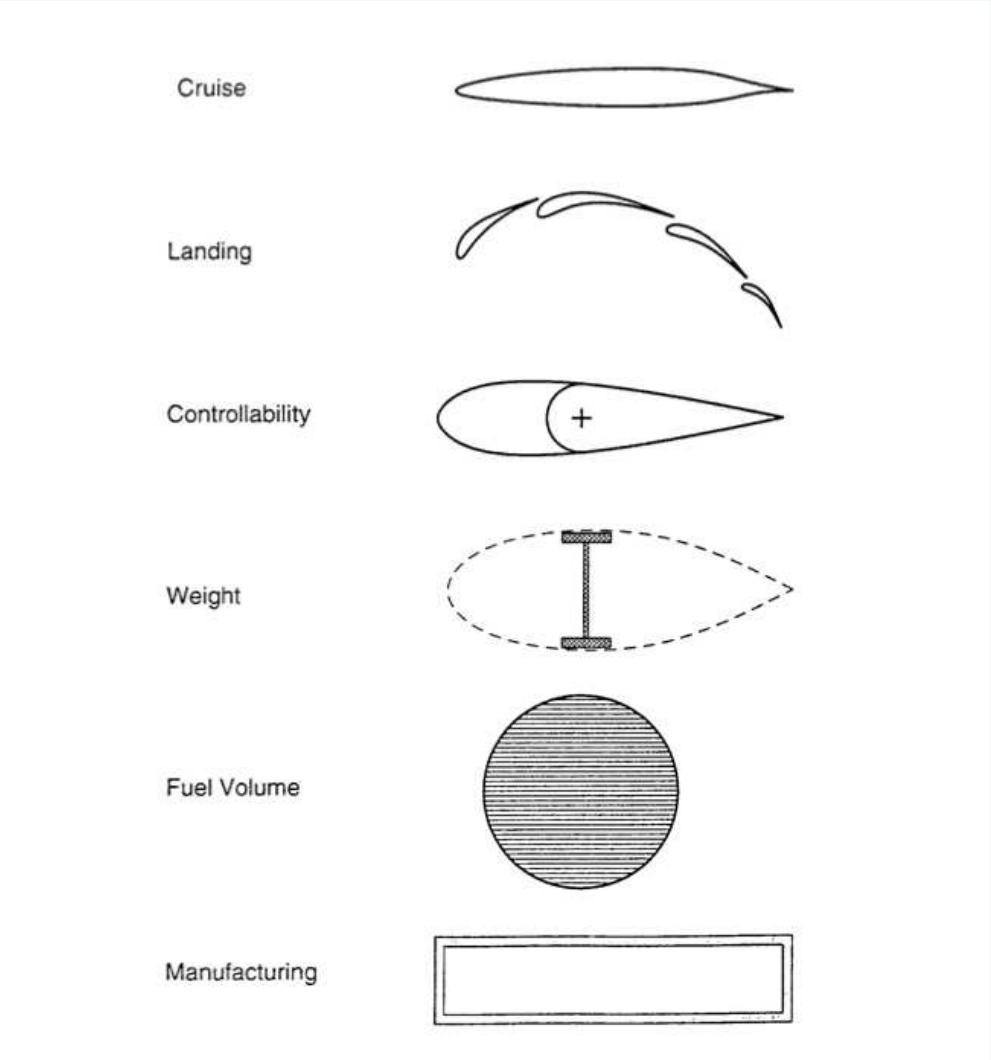
John F. Dannenhoffer, III

jfdannen@syr.edu

Syracuse University

A Discipline's View of Geometry

Dream Airfoils



Drela, *timeless*

Overview

- Introduction/Definitions
- Observations – Building a COTS Design System?
  - Commercial CAD / Geometry Generation
  - Multidisciplinary Analysis & MDO Frameworks
- Suggestions
  - *Design Intent*
  - Geometric Views
  - Parametric Sensitivities
  - Federated Software Architecture
  - Attribution Throughout
- The Engineering SketchPad (`ESP`) – An Implementation
  - `EGADS`
  - `OpenCSM`
  - `CAPS`

Introduction

In Analysis-based Parametric Studies,
Automated Design (through Analysis) and Design Optimization:

IT IS ALL ABOUT THE PROCESS!

Any module or component of the process that cannot be handled in a fully *hands-off* manner will be the bottleneck!

How can you do 1,000s of iterations, each requiring manual intervention?

In Architectural Design they say: *"Form follows Function"*
In Aircraft Design: **Form is Function**

Therefore how can one perform Aircraft Design without complete control over *form* (i.e., geometry)?

Introduction

Aircraft Design as a System Engineer Endeavor

- Given: Requirements & Mission Statement
- Multi-fidelity
- Multidisciplinary/Interdisciplinary
  - Aerodynamics
  - Structural
  - Controls
  - Manufacturing
  - ...

Parameterization – Art form

- Describes the *form* and how it can change
- Defines the *Design space* for Optimization
  - Will not be Orthogonal – Will not be Convex
- Should be in a Basis understood by a Practitioner in the Discipline
- Should NOT be *hard-coded*

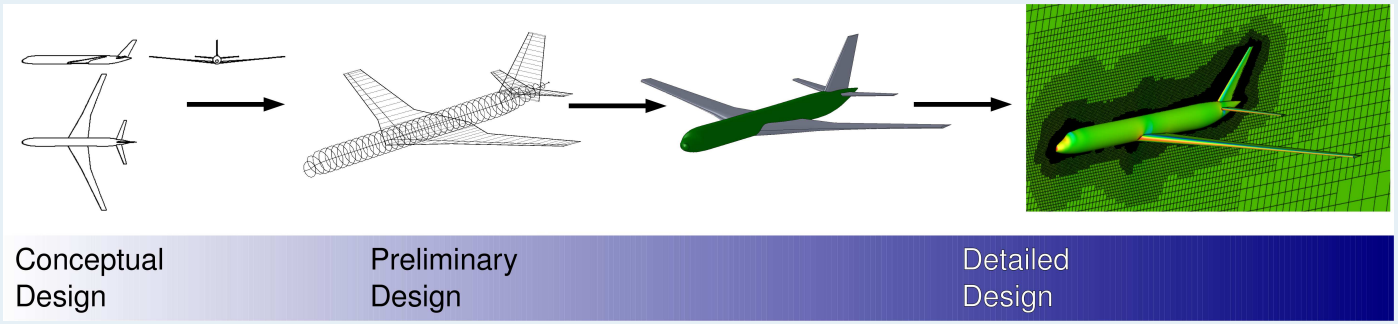
Introduction

Design Optimization

- Must be able to easily adjust the *Objective Function*
  - Not just L/D – what is the mission?
- Not about the Final Result
  - Optimizers focus on Bad or Incomplete Problem statements by producing *interesting* results
- Learn about the Problem
  - Examine the Optimum
  - Understand the Constraints & the Path taken
- Optimization SHOULD be an important tool for the Designer!

Introduction

Traditional Aircraft Design



This view is Artificial and Limiting!

- Huge Gap between *Conceptual* and *Preliminary*
- Artist renderings are used to bridge this Gap!
  - At the *Conceptual* level we know the form of the 3D Geometry
- No single *Model* or parameterization used throughout
- Need a Perspective that:
  - Can defer *down-select* as long as possible
  - Must be able to traverse these phases in either direction
    Should be a continuum

HiFi Analysis vs. Commercial CAD

CEA and CAD Systems have all been developed independently

- 3D Computational Engineering Analysis Requirements:
  - Closed watertight representation
  - Specification of materials/boundary conditions
  - Geometric fidelity depends on Analysis
- Current Geometric capture through Translation
  - Triangulation (STL – may not be closed or manifold)
  - IGES (Not a *Solid* and therefore requires intervention)
  - STEP (Can hold *Solids* but still may have closure issues)
- View that Geometry is only needed for the Mesh
- Analysis producers (except for Structural) have been CAD-phobic

Should CAD be fully embraced as our repository for Geometry?

Observation – Are MDO Frameworks Useful?

MDO Frameworks

- Provide numerous Optimization schemes
- Data-flow execution
- Cannot deal with *Rich data* – only knows scalar quantities
  - BReps
  - Mapped data (i.e., Pressures)
- Encodes, but has no ability to improve the Process
- Does not provide *Hooks* for Interdisciplinary Communication
  - User is responsible for *pair-wise* data transfers
  - Conservative data transfer is difficult to implement!

The answer is Yes and No!

Observation – We have a Broken Design Process

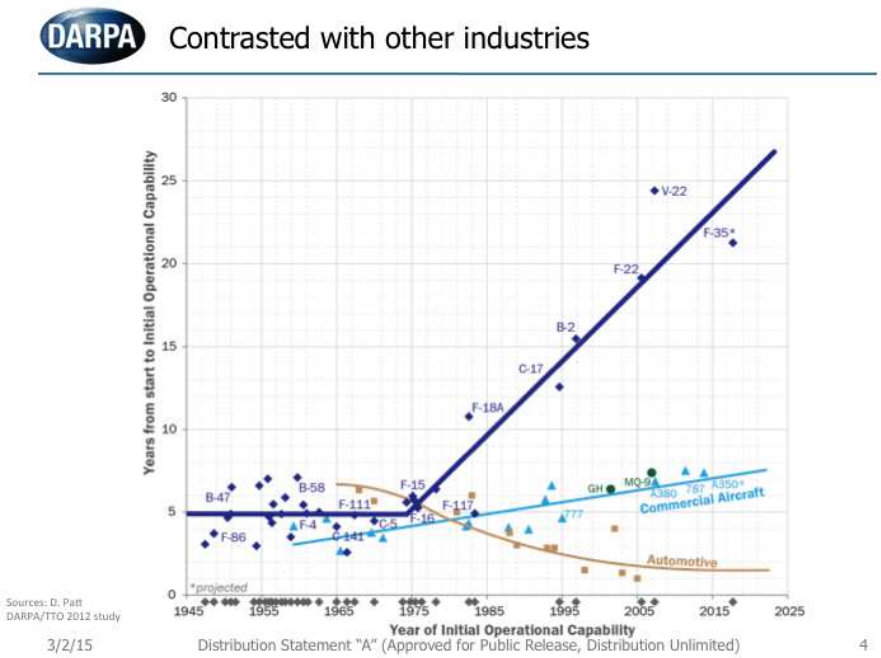
- Little Communication between Disciplines & with CAD
- CAD puts out one Geometric Representation (for Manufacturing)
- Making the Geometry ready for Analysis:
  - *Fixing* or *Healing* – Analysis doesn't deal with the BRep directly
  - Reverse-engineering of the Geometry – MAT
  - *Defeaturing* to remove aspects inappropriate for Analysis – Holes
- No Quantification of Errors maintained by the above
- Arrows between Process *Boxes* are individuals *munging* data

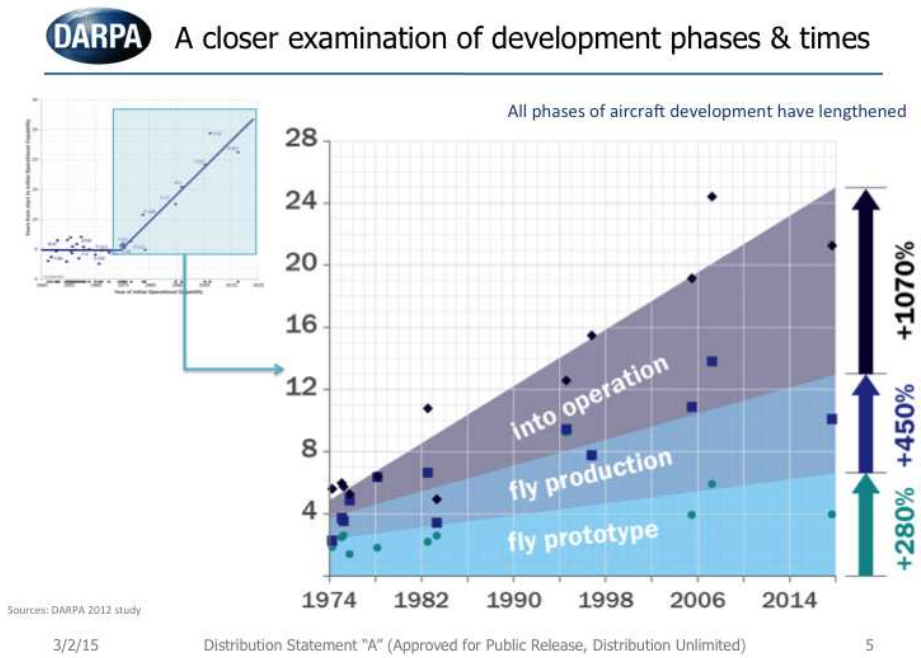
End Result

- No easy way to communicate the Design (hidden in a CAD File)
- Engineers are not doing Engineering & Designers not Designing
- All are at affect of the Process / Are we slaves to the computers?

We have a paperless paper-based process!

DARPA has Noticed the Problem





- What has computer power done over this timeframe?
- Is Analysis assisting in reducing the Design cycle time at all?

The Origins of CAD

TECHNICAL MEMORANDUM Copy No. 14

COMPUTER-AIDED DESIGN RELATED TO
THE ENGINEERING DESIGN PROCESS

by

S. A. Coons and R. W. Mann
(Mechanical Engineering Department)

8436-TM-5

October, 1960

Contract No. AF-33(600)-40604

The work reported in this document has been made possible through the support extended to the Massachusetts Institute of Technology, Electronic Systems Laboratory, by the Manufacturing Methods Division, AMC Aeronautical Systems Center, United States Air Force, under Contract No. AF-33(600)-40604, M. I. T. Project No. 8436. Part of the work is being performed by the Mechanical Engineering Department, Design and Graphics Division, under M. I. T. Project No. 8477. The report is published for technical information only and does not represent recommendations or conclusions of the sponsoring agency.

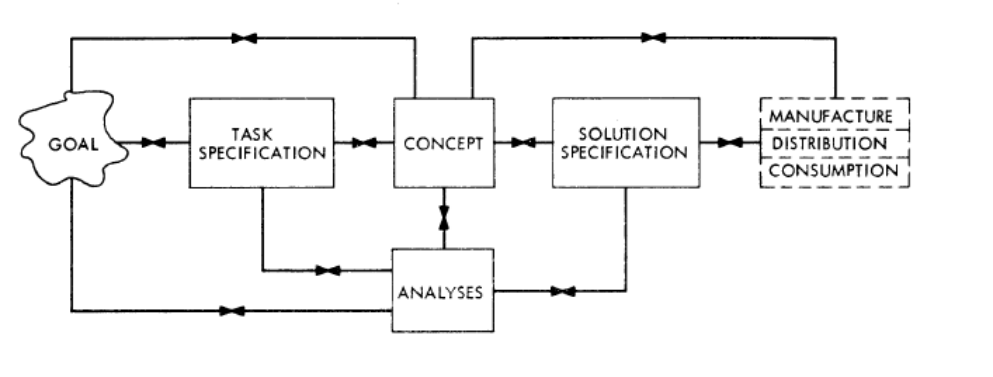
Approved by:

Douglas T. Ross, Project Engineer
Head, Computer Applications Group

Electronic Systems Laboratory
Department of Electrical Engineering
Massachusetts Institute of Technology
Cambridge 39, Massachusetts

The Origins of CAD

The Engineering Design Process – Figure 1



1960 – State of Computing

- Speed
- Memory
- Languages
- Video Output – See Sutherland's `Sketchpad`
  `https://www.youtube.com/watch?v=495nCzxM9PI`

How did this all go so wrong?

Suggestions

- Redesign the Design System!
- We need to go back to Coons' original CAD vision
  Apply current software and hardware technologies / abstractions
- We need a system that is:
  - Fundamentally a recasting / unifying CAD with MDO frameworks
    With the Addition of Support for Interdisciplinary Communication
  - Easy Accessible by any Software component via well crafted APIs
  - Open Source
  - Extensible & Extendable
  - Supporting *Rich* data at the Infrastructure level
- Cannot Evolve – Just toss out the old Process
  The definition of Disruptive Technology!

`ESP/CAPS` **is an Implementation of these Ideas**

Suggestion – *Design Intent*

- The Repository where the Design is Articulated
- Drives the entire Design Process
- Must be (human) readable – Communication
  - Sketching
  - A *Language* that can also be parsed by Computers to generate an Instance of the Design (`OpenCSM` in `ESP`)
- This is NOT Geometry – Geometry is an Outcome
  - Describe via Geometry Construction (currently as in `ESP`) – or –
  - *Design Intent* **Compiled** to specify constructing the differing Geometric Views

Suggestion – Geometric Views

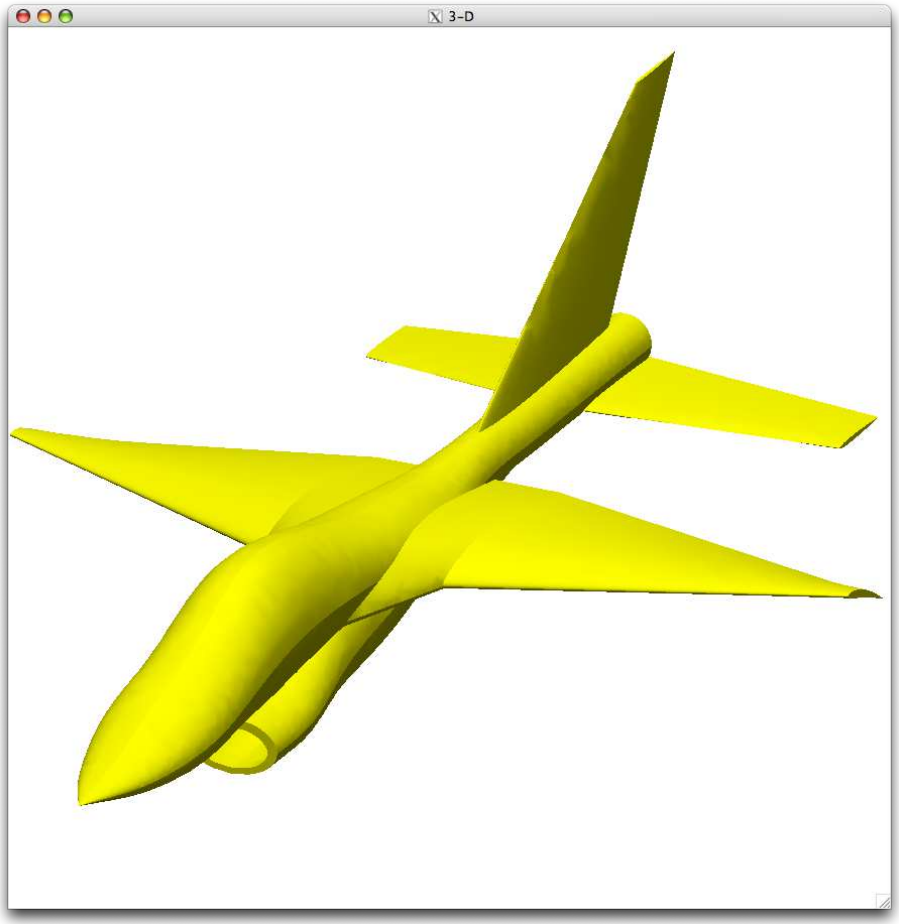
- *Design Intent* must be *Analysis Aware*
- Forward Engineer (e.g., don't take an OML to *slice* and *dice*)
- Parse the *Design Intent* to generate Geometry at the desired Fidelity and for the Discipline *at hand*
- Use a consistent (hierarchal) Parameterization
  Allows for a Rigorous Spectrum of Fidelities (at Each Discipline)

Geometry is constructed that can be directly used by the Discipline

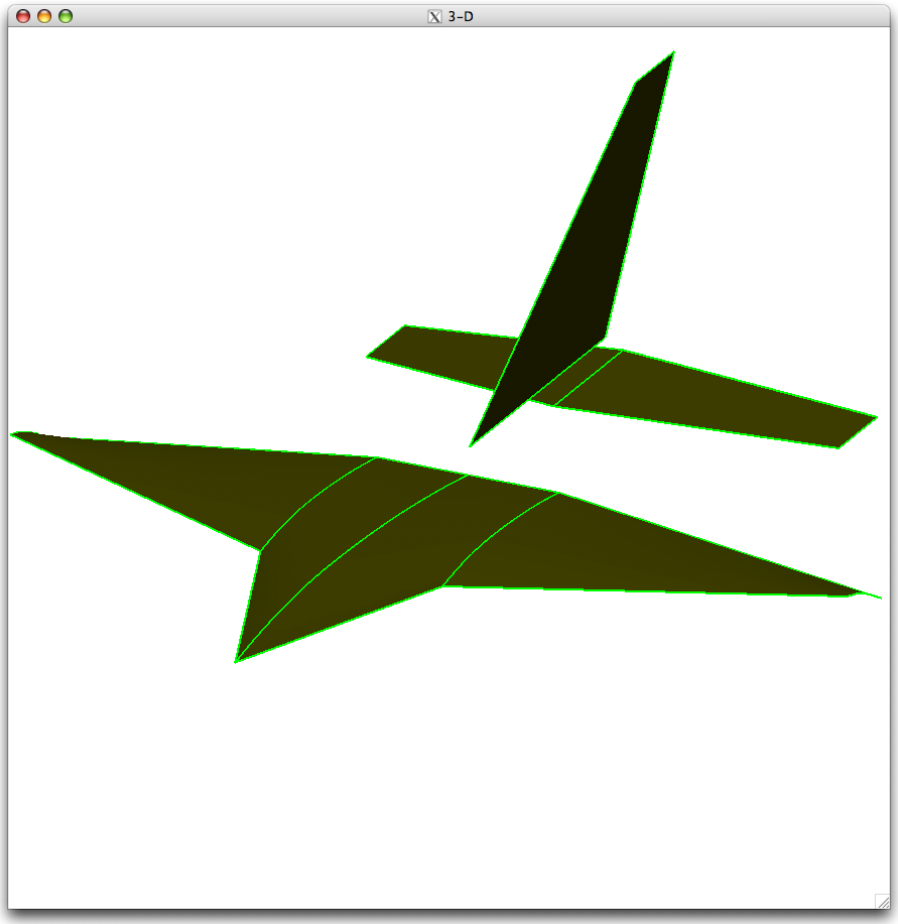
– or –

the Meshing Subsystem required by the Discipline

Multi-fidelity Geometry – Aerodynamics

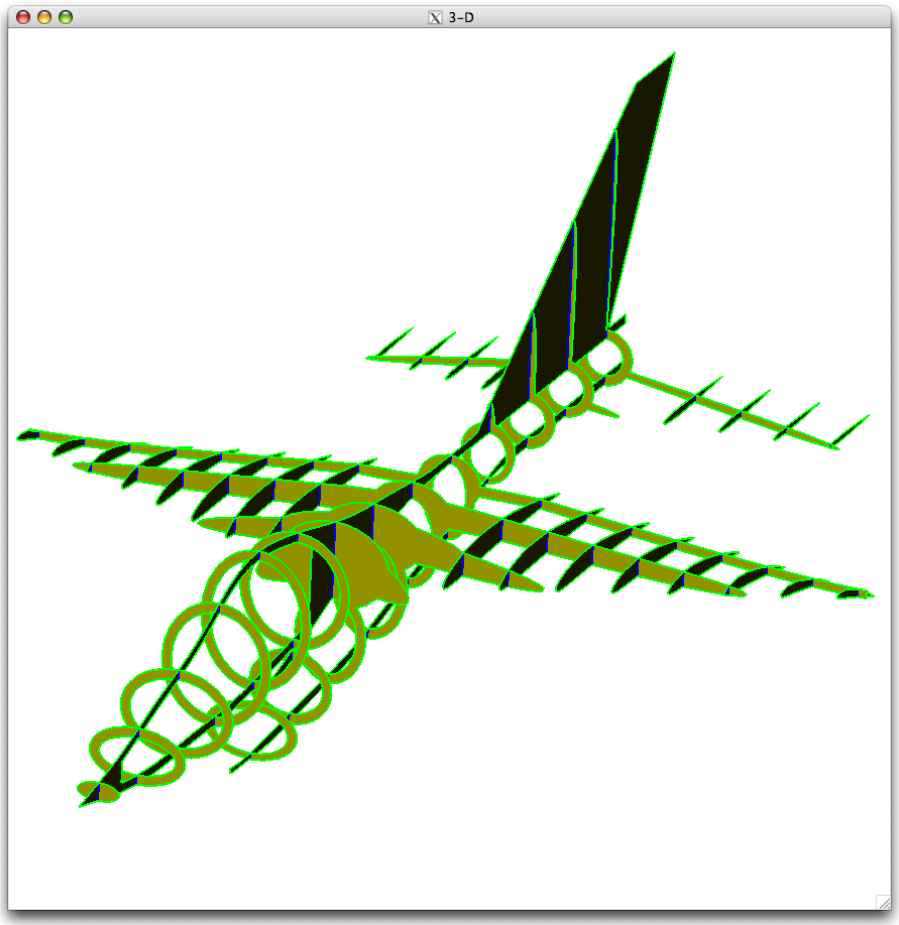


Outer Mold Line (OML)

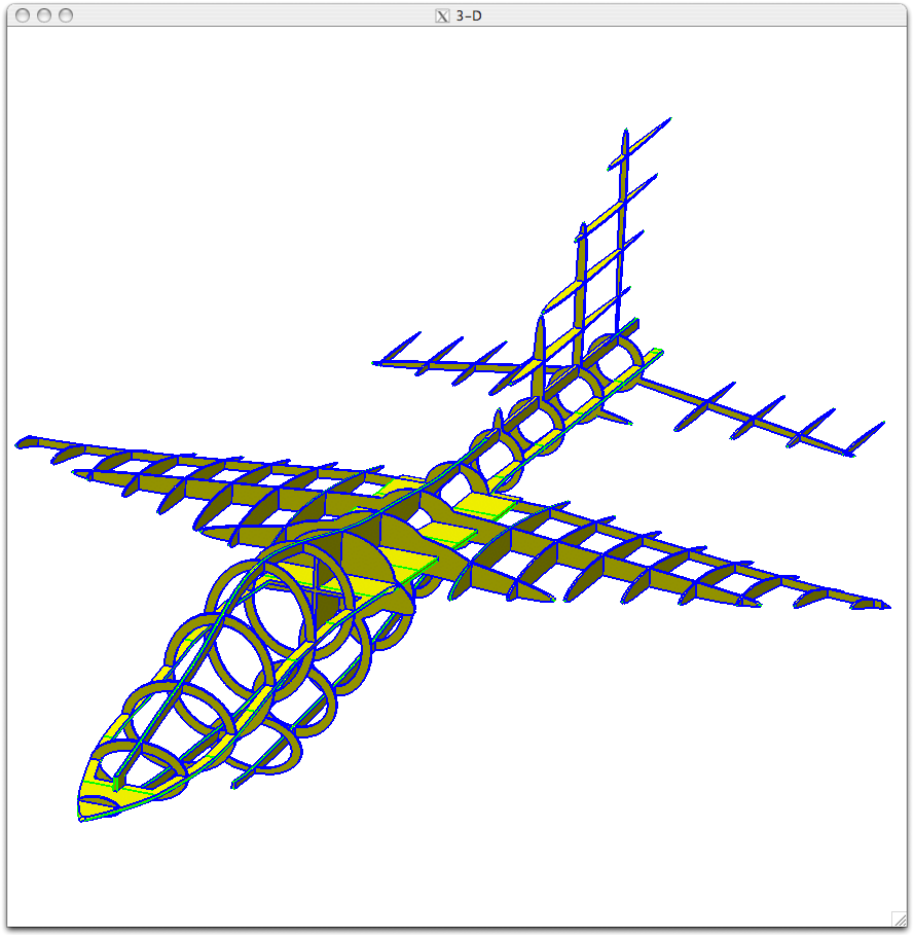


Mid-surface Aero

Multi-fidelity Geometry – Structures



Built-up Element Model



Solid Structural Model

Suggestion – Parametric Sensitivities

- Designer
  - Complex Models/Complex Parameterizations
    May not be obvious how the Parameter changes the *form*
- Gradient Based Optimization
  - An Adjoint-based solver can generate the complete suite of Sensitivities down to the Surface Mesh – Cost is 2 solves

$$\frac{\partial F}{\partial \boldsymbol{x}} = \underbrace{\frac{\partial F}{\partial \boldsymbol{U}}}_{1} \cdot \underbrace{\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{V}}}_{2} \cdot \underbrace{\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{S}}}_{3} \cdot \underbrace{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{x}}}_{4}$$

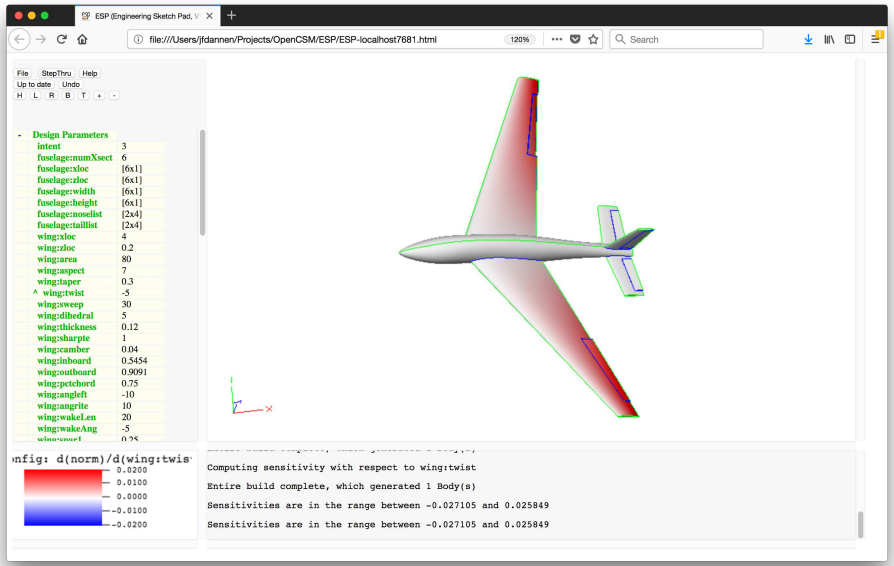
$F = \int_\Omega f d\Omega$
$f = f(\boldsymbol{x}, \boldsymbol{U})$
$\boldsymbol{U}$ = CFD Solution
$\boldsymbol{V}$ = Volume Mesh
$\boldsymbol{S}$ = Surface Geometry
$\boldsymbol{x}$ = Configuration Parameters

Sensitivity of:

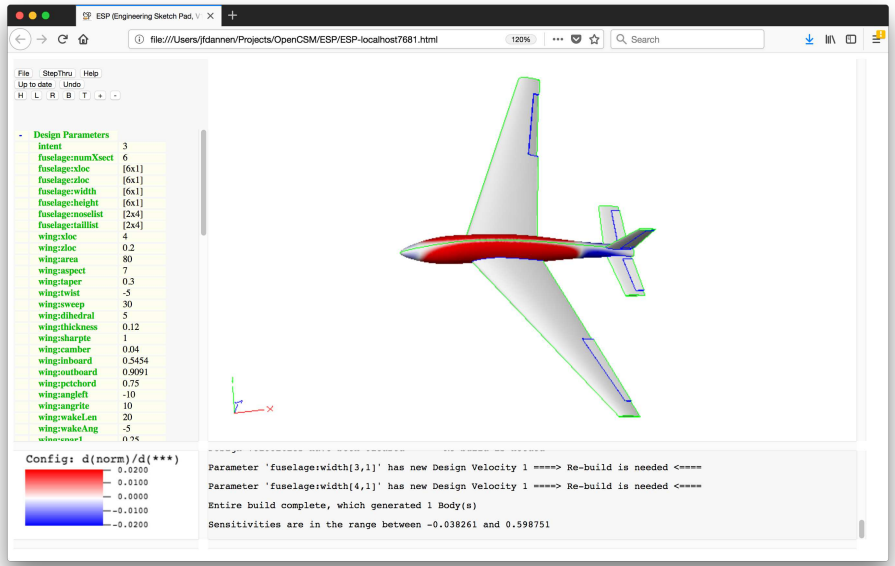
1. Objective Function to Solution
2. Solution to Volume Mesh
3. Mesh to Geometry
4. Geometry to Parameter

Why are Sensitivities NOT part of Commercial CAD Systems?

Parametric Sensitivities in ESP



twist



fuselage width

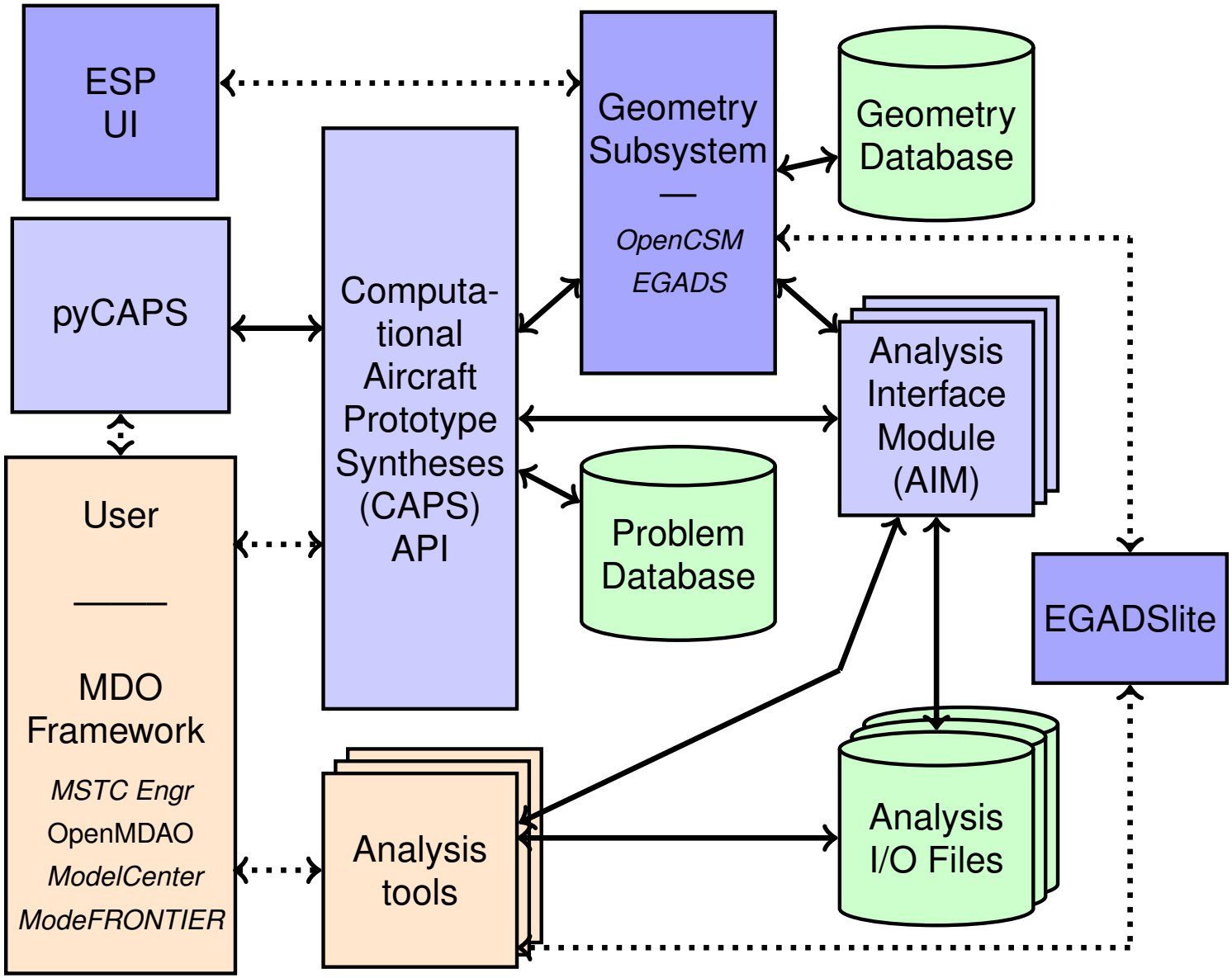
Suggestion – Federated Software Architecture

- Design Teams are Distributed – Use Web Services Overall
- "Sketching" on iPads/Tablets – Provide a Paper Replacement
- Discipline Solvers as **Services**
  - Solvers are **still** developed in Isolation – Need to Open this up!
  - Provide Integrated QoIs as a part of the Output
  - WSDL should be used to define both Inputs and Outputs
- UI using JavaScript and done in Web Browsers
- All of this makes the MDO Framework's Job simple!
  Can and should be done in a Browser
  - Software Installation is *lightweight* or nonexistent
  - Software Maintenance can be easily Managed
  - Licensed Software as Services – Manage the number of Licenses

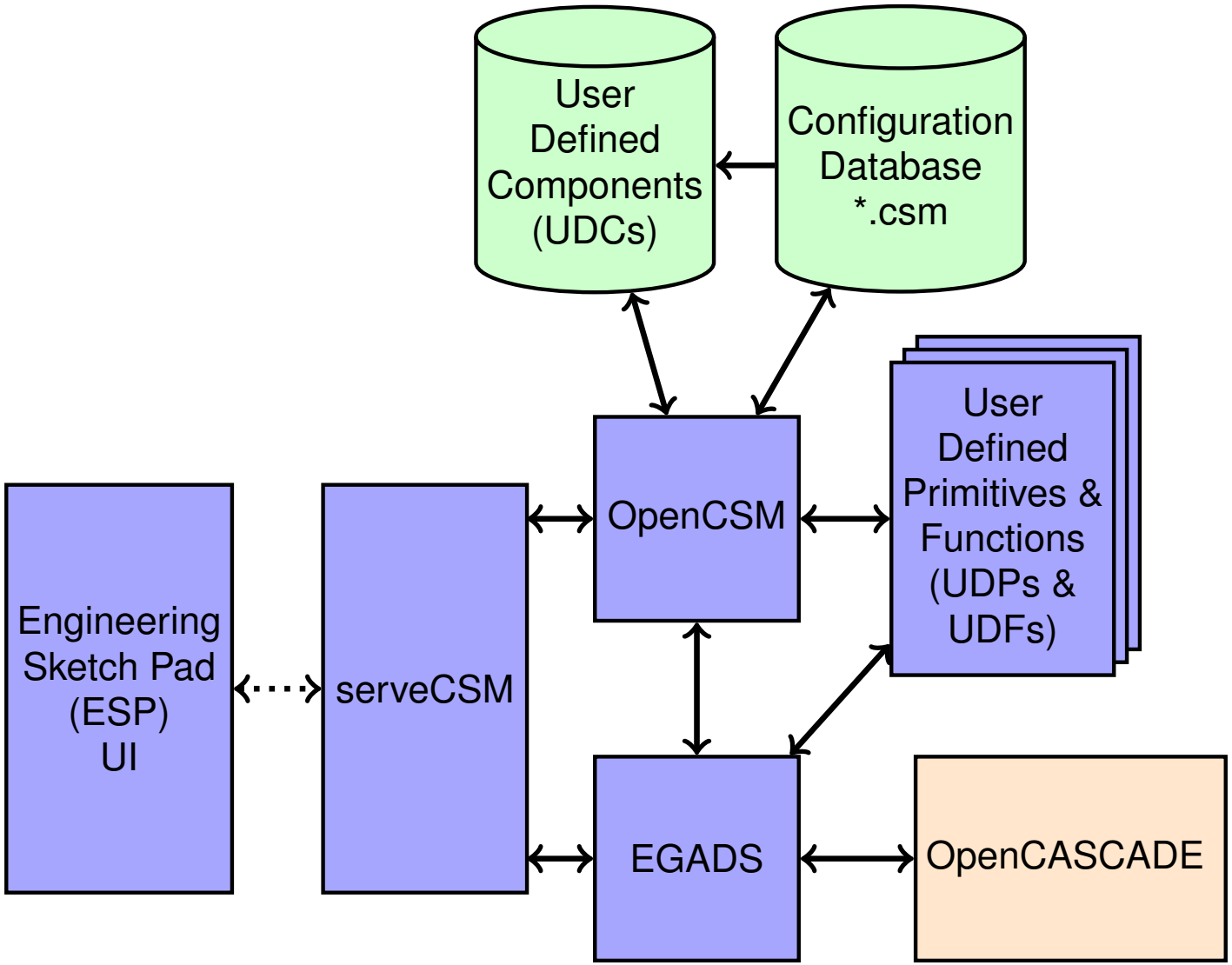
Suggestion – Attribution Throughout

- Metadata that ends up attached to the BRep
- The "Glue that Binds"...
  - Assign Solver Boundary Conditions
  - Material Properties
  - Interdisciplinary Data
  - Mesh spacings
- Air Force's *Digital Thread / Digital Twin*
- From *Design Intent* – not placed on Geometry directly
  - Commercial CAD provides: *name*, *layer*, *color*
  - Need general Name/Value(s) pairs
  - Must be applied as the Build progresses – Responsibility of the Geometry Subsystem to maintain the tagging through Operations

Engineering Sketch Pad (ESP) w/ CAPS



ESP's Geometry Subsystem Architecture



EGADS Overview

The Engineering Geometry Aircraft Design System (EGADS) is an open-source geometry interface (API) to OpenCASCADE

- reduces OpenCASCADE's 17,000 methods to less than 100 calls
  - Supports C, C++ & FORTRAN
- provides *bottom-up* and/or *top-down* construction
- geometric primitives
  - curve/p-curve: line, circle, ellipse, parabola, hyperbola, offset, bezier, BSpline (including NURBS)
  - surface: plane, spherical, conical, cylindrical, toroidal, revolution, extrusion, offset, bezier, BSpline (including NURBS)
- solid creation and Boolean operations (*top-down*)
- provides persistent user-defined attributes on topological entities
- adjustable tessellator (*vs* a surface mesher) with support for finite-differencing in the calculation of parametric sensitivities

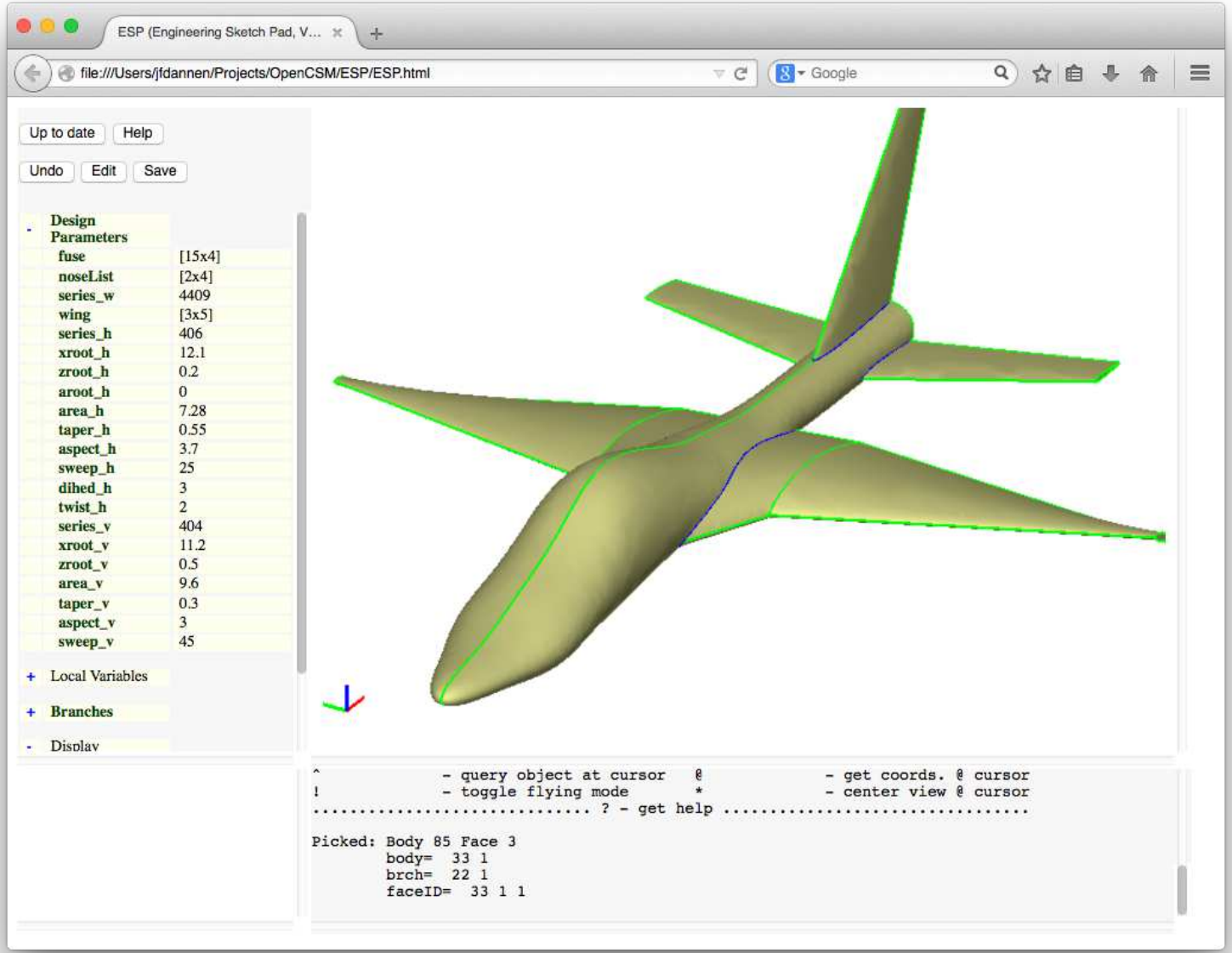
The dependency on OpenCASCADE is being reduced while the EGADS API is being maintained

EGADS Overview

EGADSlite – for HPC Environments

- No construction supported
- Same API and Object model as EGADS
  - Can use EGADS to prototype/build EGADSlite code
- Suitable for an MPI setup:
  - Data export from EGADS via a *stream*
  - Data import to EGADSlite from the *stream*
  - *Stream* setup to Broadcast (or write to disk)
- ANSI C – No OpenCASCADE – GPU friendly
- Tiny memory footprint
- Thread safe and scalable
  - EGADS' OpenCASCADE evaluation functions replaced with those written for EGADSlite

OpenCSM Overview



Screen Shot of ESP through serveCSM

OpenCSM Overview

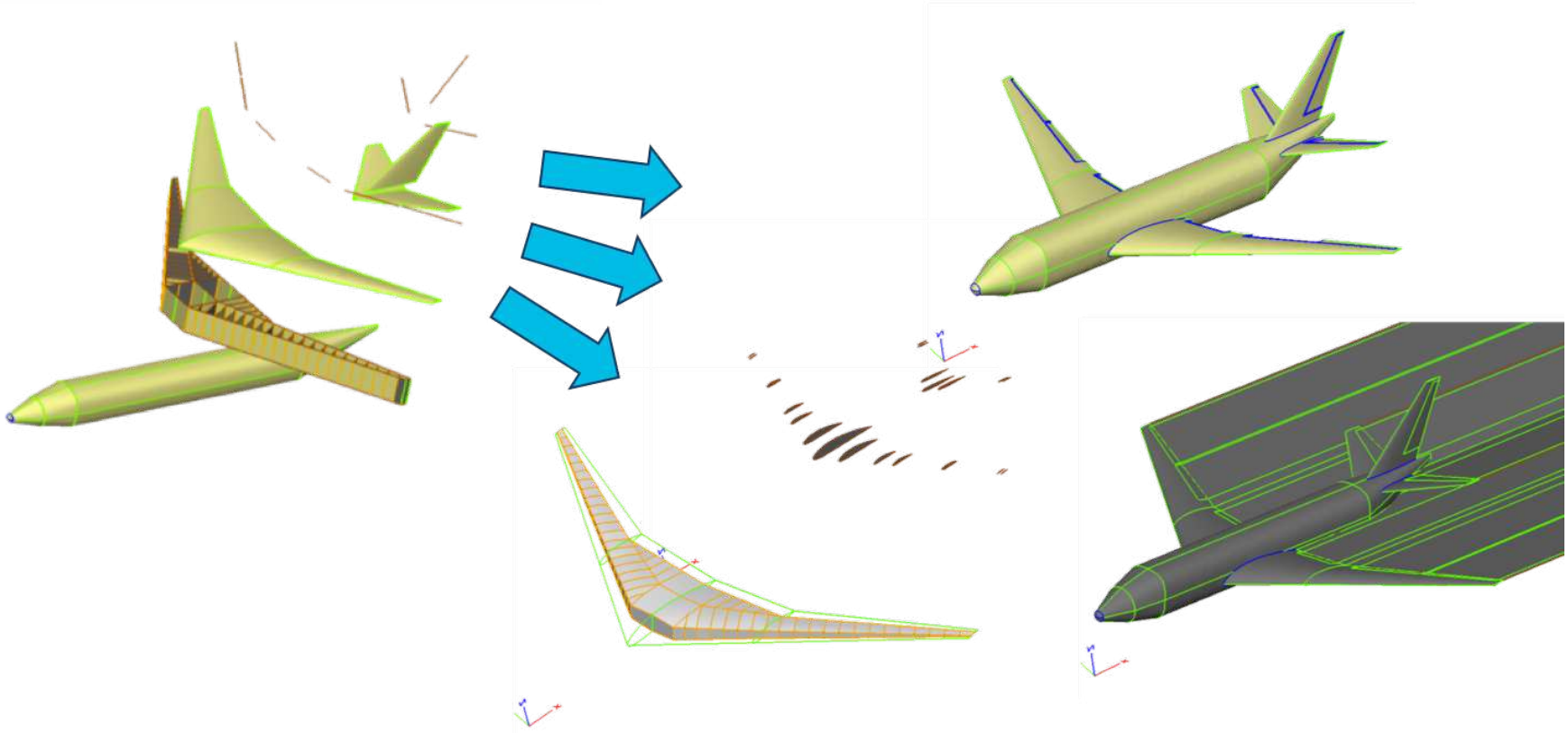
- Script driven – CAD like *Feature Tree* and *Parameters*
- Builds various *Analysis Views* from the same suite of *Parameters*
- Solid Modeling w/ *WireBody* and *SheetBody* support
- Access via API or GUI
- Attributes assigned during build (part of the language)
- Deployable
  - Back-end runs on: LINUX, OSX and Windows
  - UI runs in a browser: FireFox, Chrome, Safari
- Embeddable – example: Cart3D Design Framework
- Differentiated (analytically where possible)
- Extensible – UDPs & UDFs (and macros – UDCs)
  - EGADS applets written in C, C++ and/or FORTRAN
  - *Bottom-up* and/or *Top-down* build
  - Dynamically loaded at run-time

OpenCSM Overview – Multidiscipline Build

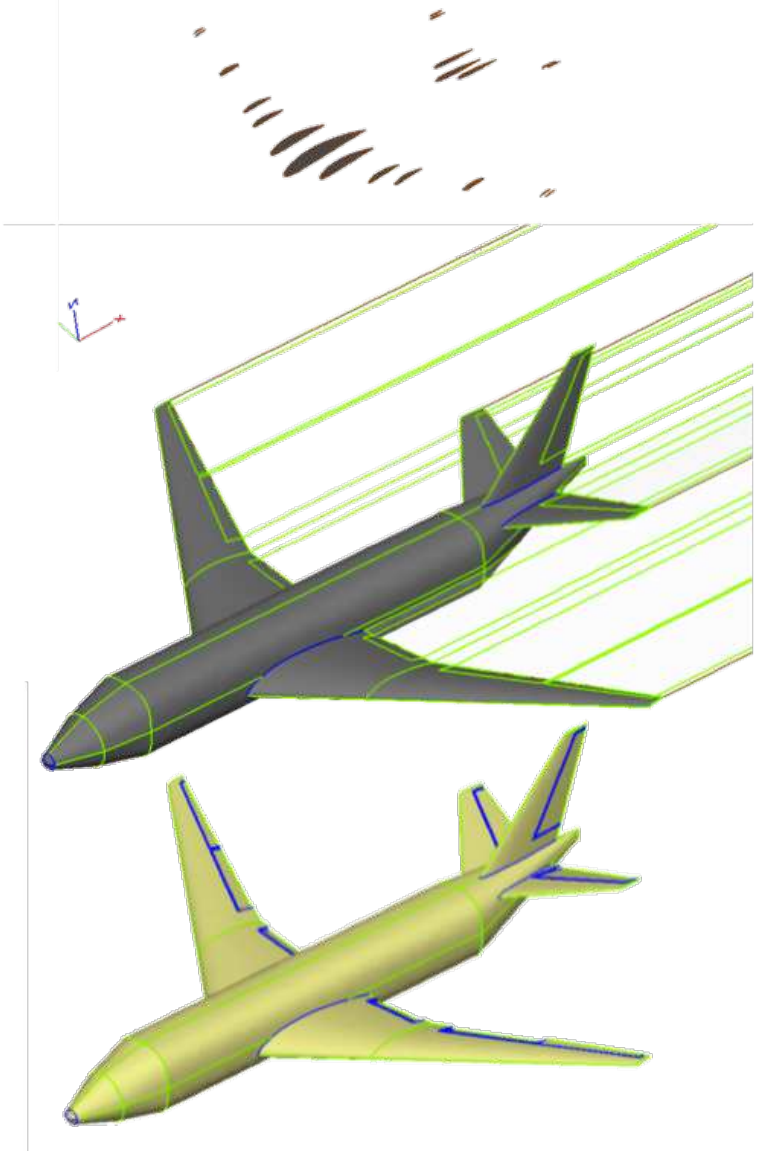
- The design can be component driven that conceptually comprises a configuration
- The *Design Intent* describes the behavior of the attributed model in response to design parameter changes

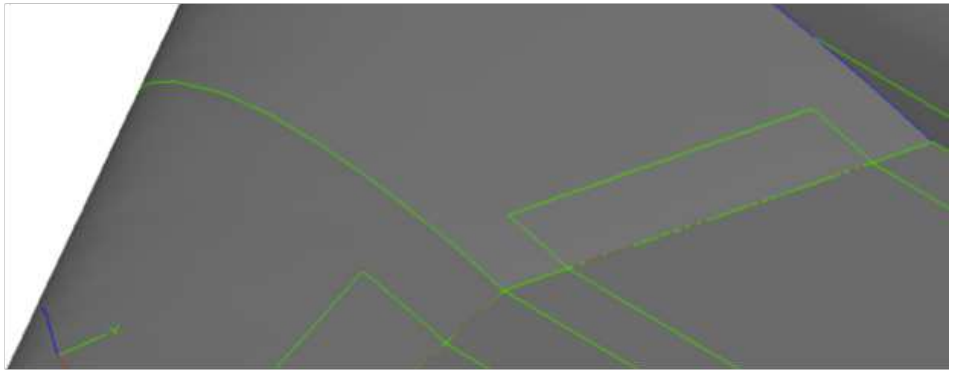
OpenCSM Overview – Views

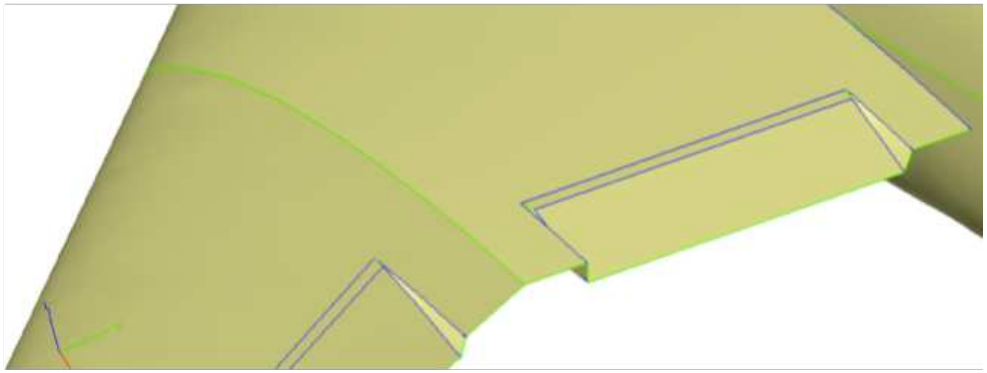
- ESP generates mathematically-consistent geometry, enabling multifidelity & multidisciplinary analysis

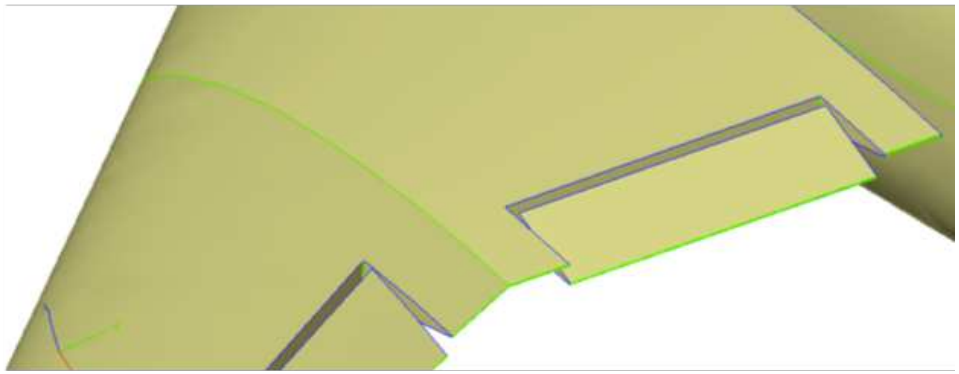


OpenCSM Overview – CFD Views









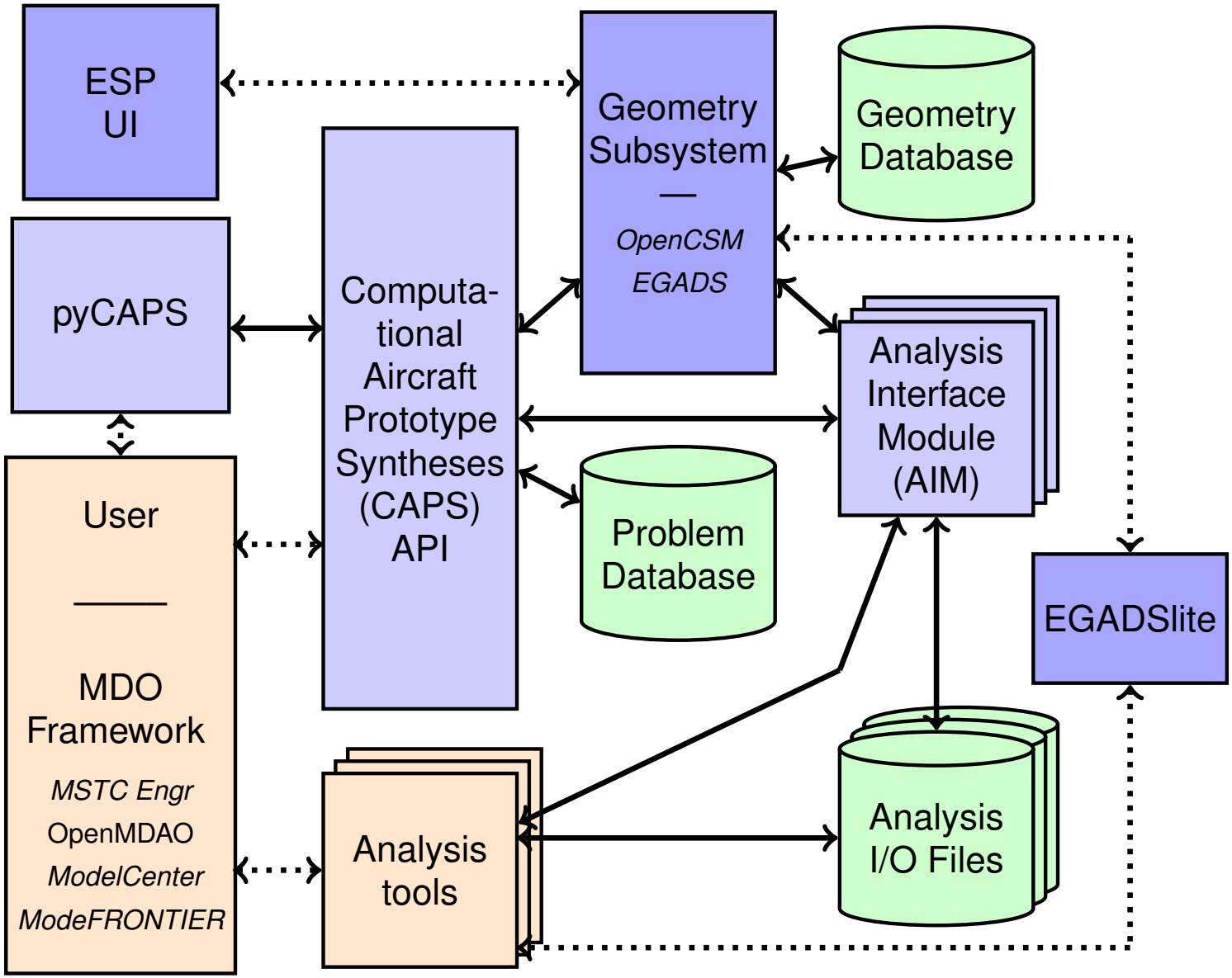
CAPS Goals

- Augment/fix MDO frameworks
- Provide the tools & techniques for generalizing analysis coupling
  - multidisciplinary coupling: aeroelastic, FSI
  - multi-fidelity coupling: conceptual and preliminary design
- Provide the tools & techniques for rigorously dealing with geometry (single and multi-fidelity) in a design framework / process
- Input and attribution driven automated meshing

CAPS Design Principals

- Automates high burden, error prone, repetitive touch labor tasks (e.g. mesh generation)
- A user should only have to define parameters and geometric construction once
  - geometry is driven by user-defined Design Parameters
  - geometry is attributed (tagged) during the construction process for analysis
- Analysis tools do not have to be modified
- Expandable so that new analyses can be added at any time
- System execution must be flexible enough to support nearly any design or analysis task

ESP with the CAPS Infrastructure



Some of the of AIMs shipped with ESP

Low Fidelity

- AWAVE
- Friction
- AVL
- XFoil

Structural Analysis

- masstran
- mySTRAN
- NASTRAN
- ASTROS
  - linear static & modal analysis
  - support for composites, optimization & aeroelasticity

3D CFD

- Cart3D
- Fun3D
- SU^2

Meshing

- Surface
  - Native EGADS
  - AFLR4
  - Pointwise
- Volume
  - TetGen
  - AFLR3
  - Pointwise

CAPS – Meshing

Automatic (not Automated) Meshing† is of Paramount Importance

CFD:

- Pointwise†
- AFLR
- TetGen

Structural Analysis (Built up Element Model):

- EGADS triangle or full quadrilateral tessellation

All from EGADS geometry directly!

What you will learn this week

Sessions:

- ESP Overview & Getting Started
- CSM Language
- Solids Fundamentals
- Attribution
- Sketcher Fundamentals
- Putting It All Together
- Python Tutorial
- CAPS Overview
- CAPS Geometry
- CAPS Analysis
- Aero Modeling
- Meshing for CFD
- CFD Analysis: Fun3D and SU2
- Meshing for Structures
- Structures Analysis: mASTROS/NASTRAN
- Data Transfer: Loosely-Coupled Aeroelasticity
- Wrapping Up