

Engineering Sketch Pad (ESP)



Training Session 10 Putting It All Together

John F. Dannenhoffer, III

jfdannen@syr.edu
Syracuse University

Bob Haimes

haimes@mit.edu
Massachusetts Institute of Technology
updated for v1.18

- During the design of an aircraft, various coupled models are needed
 - different disciplines
 - structures
 - controls
 - aerodynamics
 - ...
 - different fidelities
 - conceptual design
 - preliminary design
 - detailed design
- There needs to be communication between these models



Computational Aircraft Prototype Syntheses (CAPS)

- In order to support multi-fidelity and multi-disciplinary analyses, the **CAPS** program has been developed
 - funded by the AFRL
- **CAPS** uses geometries (and sensitivities) generated by **ESP**
- **CAPS** provides interfaces to many analysis programs, including:
 - aerodynamics (at various fidelities)
 - structures (at various fidelities)
 - ...
- There is a companion training course for **CAPS** that can be offered if there is sufficient interest

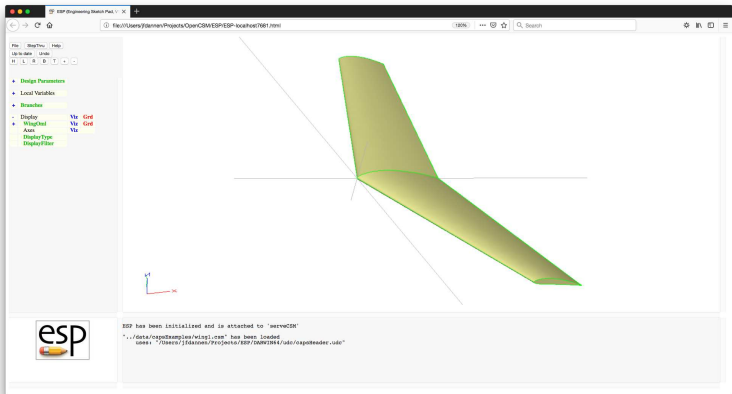
- One of the strengths of **ESP** is to be able to have multiple models of a single configuration
 - driven by a single set of Design Parameters
 - attributed so that “common” features could be linked together
- This capability has been used by the **CAPS** program
 - a set of “views” have been created, which can be used if the model is constructed and annotated in a consistent way
 - for AVL, SansLIP, SU2, Astros, ...
 - implemented as a series of UDCs

- Analysis of Simple Wing (`wing1`)
 - basic assumptions (orientation, ...)
 - required Bodys
 - required attributes (naming vs. meta-data)
- Analysis of wing with flaps (`wing2`)
 - required Bodys
 - required attributes (naming vs. meta-data)
- Analysis of wing structure `wing3`)
 - required Bodys
 - required attributes (naming vs. meta-data)
- Full aircraft model (`transport`)



wing1.csm

Isolated Wing: Outer Mold Line (OML) Only





Design Parameters for wing1

File can be found at `$ESP_ROOT/training/ESP/data/session10`

wing:area	10.0	wing area
wing:aspect	6.00	aspect ratio
wing:taper	0.60	taper ratio
wing:sweep	20.0	deg (of leading edge)
wing:thick	0.12	thickness ratio, frac of local chord
wing:camber	0.04	maximum camber, frac of local chord
wing:washout	5.00	deg (down at tip)
wing:dihedral	4.00	deg

- VIEW:Concept — conceptual design
- VIEW:VLM — vortex lattice method
- VIEW:CFDInviscid — inviscid CFD analysis
- VIEW:CFDVisous — viscous CFD analysis

- Configuration files defines the necessary Bodys
- Bodys are oriented such that:
 - x points out the tail
 - y points out the right wing
 - z points up

- Outer Mold Lines (OMLs) for each component
 - Fuse0m1 (a SolidBody)
 - Wing0m1 (a SolidBody)
 - Htail0m1 (a SolidBody)
 - Vtail0m1 (a SolidBody)

- Body
 - tagComp with value \$leftWing or \$riteWing
- Faces
 - tagComp with value \$leftWing or \$riteWing
 - tagType with value \$root, \$tip, \$upper, \$lower, or \$trailingEdge
- Edges
 - tagType with value \$root, \$leadingEdge (with supporting tagComp), or \$trailingEdge (with supporting tagComp)



Dissection of `wing1.csm` (1)

File can be found at `$ESP_ROOT/training/ESP/data/session10`

- Definition of VIEWS to be supported
- Definition of COMPOnents that are defined
- Definition of Design Parameters
- Call to `capsHeader`
- Construction of `Wing0m1` (with attributes)
- Call to `capsViews`

```
# wing1
# written by John Dannenhoffer

# define the views
CFGPMTR   VIEW:Concept      1
CFGPMTR   VIEW:VLM          0
CFGPMTR   VIEW:CFDInviscid  0
CFGPMTR   VIEW:CFDViscous   0

# define components to be used
CFGPMTR   COMP:Wing         1

# Design Parameters for OML
DESPMTR   wing:area         10.0    # wing area
DESPMTR   wing:aspect       6.00    # aspect ratio
DESPMTR   wing:taper        0.60    # taper ratio
DESPMTR   wing:sweep        20.0    # deg (of leading edge)
DESPMTR   wing:thickr       0.12    # thickness ratio at root
DESPMTR   wing:camber       0.06    # camber ratio at root
DESPMTR   wing:thickt       0.16    # thickness ratio at tip
DESPMTR   wing:cambert      0.02    # camber ratio at tip
DESPMTR   wing:alphat      -5.00    # setting angle at tip
DESPMTR   wing:diheedral    4.00    # deg
DESPMTR   wing:xroot        0.00    # xloc at root LE
DESPMTR   wing:yroot        0.00    # yloc at root LE
DESPMTR   wing:zroot        0.00    # zloc at root LE

# Define length units of the geometry
ATTRIBUTE capsLength      $ft
```

```
# convert VIEW:* variables into make* variables
UDPRIM    $/capsHeader

# wing local variables
SET       wing:span      sqrt(wing:aspect*wing:area)
SET       wing:chordr    2*wing:area/wing:span/(1+wing:taper)
SET       wing:chordt    wing:chordr*wing:taper
SET       wing:ytip      -wing:span/2
SET       wing:xtip      -wing:ytip*tand(wing:sweep)
SET       wing:ztip      -wing:ytip*tand(wing:dihedral)
SET       wing:mac       sqrt(wing:area/wing:aspect)

# make wing OML
IFTHEN    makeWingOml EQ 1
    # lay out left wing
    MARK
        # root
        UDPRIM    naca      thickness wing:thickr   camber   wing:camherr   sharpte   SHARP_TE
        SCALE     wing:chordr
        ROTATEX   90  0  0

        # left tip
        UDPRIM    naca      thickness wing:thickt   camber   wing:cambert   sharpte   SHARP_TE
        SCALE     wing:chordt
        ROTATEX   90  0  0
        ROTATEY   wing:alphat  0          0
        TRANSLATE wing:xtip    wing:ytip    wing:ztip
    RULE
        ATTRIBUTE tagComp $leftWing
    SET         ruledBody @nbody
```

```
SELECT    FACE ruledBody 1
          ATTRIBUTE tagType $root
SELECT    FACE ruledBody 2
          ATTRIBUTE tagType $tip
SELECT    FACE ruledBody 3
          ATTRIBUTE tagType $upper
SELECT    FACE ruledBody 4
          ATTRIBUTE tagType $lower
SELECT    EDGE ruledBody 3 ruledBody 4 1
          ATTRIBUTE tagComp $leftWing
          ATTRIBUTE tagType $leadingEdge
IFTHEN    SHARP_TE EQ 0
          SELECT    FACE ruledBody 5
                ATTRIBUTE tagType $trailingEdge
ELSE
          SELECT    EDGE ruledBody 3 ruledBody 4 2
                ATTRIBUTE tagComp $leftWing
                ATTRIBUTE tagType $trailingEdge
ENDIF
```



Dissection of wing1.csm (5)

```
# right wing too
STORE      LeftWing 0 1
RESTORE    LeftWing
    ATTRIBUTE tagComp $riteWing
    SELECT  EDGE  $tagType $leadingEdge
    IFTHEN  @iedge GT 0
        SELECT EDGE  $tagType $leadingEdge
        ATTRIBUTE tagComp $riteWing
    ENDIF
    SELECT  EDGE  $tagType $trailingEdge
    IFTHEN  @iedge GT 0
        SELECT EDGE  $tagType $trailingEdge
        ATTRIBUTE tagComp $riteWing
    ENDIF
    CATBEG   $edge_not_found
    CATEND
MIRROR     0    1    0
JOIN

SELECT     EDGE  ruledBody 3 ruledBody 3 1
    ATTRIBUTE tagType $root
SELECT     EDGE  ruledBody 4 ruledBody 4 1
    ATTRIBUTE tagType $root

STORE      WingOml
ENDIF

# now generate the needed views
UDPRIM     $/capsViews

END
```




New Design Parameters for wing2

File can be found at `$ESP_ROOT/training/ESP/data/session10`

<code>wing:hinge[i,1]</code>	deflection (degrees)
<code>wing:hinge[i,2]</code>	x/c at y -min end
<code>wing:hinge[i,3]</code>	$y/(b/2)$ at y -min end
<code>wing:hinge[i,4]</code>	z/t at y -min end
<code>wing:hinge[i,5]</code>	x/c at y -max end
<code>wing:hinge[i,6]</code>	$y/(b/2)$ at y -max end
<code>wing:hinge[i,7]</code>	z/t at y -max end
<code>wing:hinge[i,8]</code>	gap when cutting out for CFD
<code>wing:hinge[i,9]</code>	group (used to link controls in VLM)

- Outer Mold Lines (OMLs) for each component
 - FuseOml (a SolidBody)
 - WingOml (a SolidBody)
 - HtailOml (a SolidBody)
 - VtailOml (a SolidBody)
- Hinge lines for each control surface i on each component
 - WingHinge i (a WireBody)
 - HtailHinge i (a WireBody)
 - VtailHinge i (a WireBody)

- Body
 - (none required)
- Edges
 - `tagComp` with value `$wing`
 - `tagType` with value `$hinge`
 - `tagIndex` with value i
 - `deflect` with value equal to deflection angle (in degrees), positive according to right-hand rule
 - `xoverc1` with value equal to x/c at the y -min end
 - `xoverc2` with value equal to x/c at the y -max end
 - `gap` with value equal to gap size when cutting out control surface for CFD



Dissection of `wing2.csm` (1)

File can be found at `$ESP_ROOT/training/ESP/data/session10`

- Definition of VIEWS to be supported
- Definition of COMPOnents that are defined
- Definition of Design Parameters
- Call to `capsHeader`
- Construction of `WingOml` (with attributes)
- Construction of `WingHinges` (with attributes)
- Call to `capsViews`

```
# wing2
# written by John Dannenhoffer

# define the views
CFGPMTR  VIEW:Concept      1
CFGPMTR  VIEW:VLM          0
CFGPMTR  VIEW:CFDInviscid  0
CFGPMTR  VIEW:CFDViscous   0

# define components to be used
CFGPMTR  COMP:Wing         1
CFGPMTR  COMP:Control      0

# Design Parameters for OML
DESPMTR  wing:area         10.0    # wing area
DESPMTR  wing:aspect       6.00    # aspect ratio
DESPMTR  wing:taper        0.60    # taper ratio
DESPMTR  wing:sweep       20.0    # deg (of leading edge)
DESPMTR  wing:thickr       0.12    # thickness ratio at root
DESPMTR  wing:cambrerr     0.06    # camber ratio at root
DESPMTR  wing:thickt       0.16    # thickness ratio at tip
DESPMTR  wing:cambert      0.02    # camber ratio at tip
DESPMTR  wing:alphat      -5.00    # setting angle at tip
DESPMTR  wing:dihedral     4.00    # deg
DESPMTR  wing:xroot        0.00    # xloc at root LE
DESPMTR  wing:yroot        0.00    # yloc at root LE
DESPMTR  wing:zroot        0.00    # zloc at root LE
```



Dissection of wing2.csm (3)

```
# Design Parameters for controls
DIMENSION wing:hinge      2 9 1
#                          theta  ymin                ymax                gap  grp
DESPMTR   wing:hinge      "-10.0;  0.75; -0.90;  0.50;  0.75; -0.50; 0.50;  0.10; 1; \ left aileron
                          +10.0;  0.75;  0.50;  0.50;  0.75;  0.90; 0.50;  0.10; 2" #  rite aileron

# Define length units of the geometry
ATTRIBUTE capsLength      $ft

# convert VIEW:* variables into make* variables
UDPRIM      $/capsHeader

# wing local variables
SET      wing:span      sqrt(wing:aspect*wing:area)
SET      wing:chordr     2*wing:area/wing:span/(1+wing:taper)
SET      wing:chordt     wing:chordr*wing:taper
SET      wing:ytip       -wing:span/2
SET      wing:xtip       -wing:ytip*tand(wing:sweep)
SET      wing:ztip       -wing:ytip*tand(wing:dihedral)
SET      wing:mac        sqrt(wing:area/wing:aspect)
```

```
# make wing OML
IFTHEN    makeWingOml EQ 1
  # lay out left wing
  MARK
    # root
    UDPRIM    naca      thickness  wing:thickr  camber  wing:camherr  sharpte  SHARP_TE
    SCALE     wing:chordr
    ROTATEX   90  0  0

    # left tip
    UDPRIM    naca      thickness  wing:thickt  camber  wing:cambert  sharpte  SHARP_TE
    SCALE     wing:chordt
    ROTATEX   90  0  0
    ROTATEY   wing:alphat  0          0
    TRANSLATE wing:xtip    wing:ytip  wing:ztip
  RULE
    ATTRIBUTE tagComp $leftWing
  SET      ruledBody @nbody

  SELECT    FACE ruledBody 1
    ATTRIBUTE tagType $root
  SELECT    FACE ruledBody 2
    ATTRIBUTE tagType $tip
  SELECT    FACE ruledBody 3
    ATTRIBUTE tagType $upper
  SELECT    FACE ruledBody 4
    ATTRIBUTE tagType $lower
  SELECT    EDGE ruledBody 3 ruledBody 4 1
    ATTRIBUTE tagComp $leftWing
    ATTRIBUTE tagType $leadingEdge
```

```

IFTHEN    SHARP_TE EQ 0
    SELECT    FACE    ruledBody 5
        ATTRIBUTE tagType $trailingEdge
ELSE
    SELECT    EDGE    ruledBody 3 ruledBody 4 2
        ATTRIBUTE tagComp $leftWing
        ATTRIBUTE tagType $trailingEdge
ENDIF

# right wing too
STORE      LeftWing 0 1
RESTORE     LeftWing
    ATTRIBUTE tagComp $riteWing
    SELECT    EDGE    $tagType $leadingEdge
    IFTHEN    @iedge GT 0
        SELECT EDGE    $tagType $leadingEdge
        ATTRIBUTE tagComp $riteWing
    ENDIF
    SELECT    EDGE    $tagType $trailingEdge
    IFTHEN    @iedge GT 0
        SELECT EDGE    $tagType $trailingEdge
        ATTRIBUTE tagComp $riteWing
    ENDIF
    CATBEG     $edge_not_found
    CATEND
MIRROR      0    1    0
JOIN

```



```

SELECT    EDGE    ruledBody 3 ruledBody 3 1
      ATTRIBUTE tagType $root
SELECT    EDGE    ruledBody 4 ruledBody 4 1
      ATTRIBUTE tagType $root

      STORE      WingOml
ENDIF

# make wing hinge lines
IFTHEN    makeWingOml EQ 1 AND makeWingHinge EQ 1
  PATBEG    ihinge wing:hinge.nrow
    SET      y_ibd    wing:hinge[ihinge,3]*(-wing:ytip)
    BOX      -1000 y_ibd -1000 2000 0 2000
    RESTORE  WingOml
    INTERSECT
    SET      x_ibd    @xmin+wing:hinge[ihinge,2]*(@xmax-@xmin)
    STORE    .
    BOX      x_ibd y_ibd -1000 0 0 2000
    RESTORE  WingOml
    INTERSECT
    SET      z_ibd    @zmin+wing:hinge[ihinge,4]*(@zmax-@zmin)
    STORE    .

```

```

SET      y_obd    wing:hinge[ihinge,6]*(-wing:ytip)
BOX      -1000 y_obd -1000 2000 0 2000
RESTORE  Wing0m1
INTERSECT
SET      x_obd    @xmin+wing:hinge[ihinge,5]*(@xmax-@xmin)
STORE    .
BOX      x_obd y_obd -1000 0 0 2000
RESTORE  Wing0m1
INTERSECT
SET      z_obd    @zmin+wing:hinge[ihinge,7]*(@zmax-@zmin)
STORE    .

SKBEG    x_ibd y_ibd z_ibd
          LINSEG x_obd y_obd z_obd
SKEND
SELECT   EDGE 1
          ATTRIBUTE tagComp $wing
          ATTRIBUTE tagType $hinge
          ATTRIBUTE tagIndex !val2str(wing:hinge[ihinge,9],0)
          ATTRIBUTE deflect wing:hinge[ihinge,1]
          ATTRIBUTE xoverc1 wing:hinge[ihinge,2]
          ATTRIBUTE xoverc2 wing:hinge[ihinge,5]
          ATTRIBUTE gap wing:hinge[ihinge,8]
          ATTRIBUTE compIndex !val2str(ihinge,0)
STORE    WingHinge ihinge
PATEND
ENDIF

# now generate the needed views
UDPRIM   $/capsViews

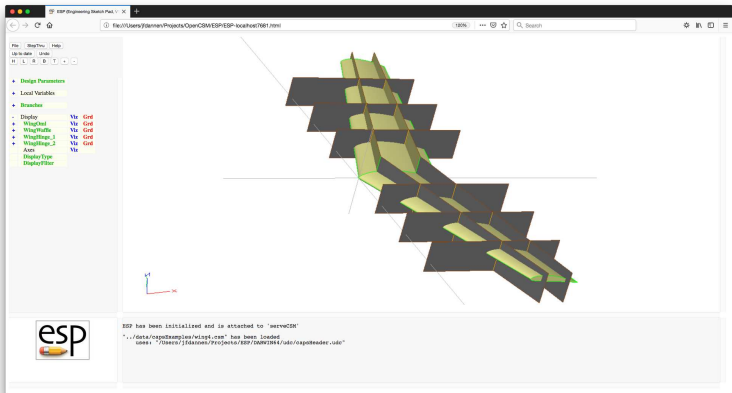
END

```



wing3.csm

Isolated Wing: OML and Structures





New Design Parameters for wing3

wing:spar1	0.20	location of fwrdr spar
wing:spar2	0.70	location of rwrdr spar
wing:nrib	3.00	number of ribs per wing



“Possible” Analyses (Views) for wing3

- VIEW:Concept — conceptual design
- VIEW:Structure — built-up element model

- Outer Mold Lines (OMLs) for each component
 - FuseOml (a SolidBody)
 - WingOml (a SolidBody)
 - HtailOml (a SolidBody)
 - VtailOml (a SolidBody)
- Waffle for each component
 - FuseWaffle (a SheetBody) — not yet supported
 - WingWaffle (a SheetBody)
 - HtailWaffle (a SheetBody) — not yet supported
 - VtailWaffle (a SheetBody) — not yet supported



Required Attributes of WingWaffle

- Body
 - (none required)
- Faces
 - `tagComp` with value `$leftwing`, `$riteWing`, or `$wing` (if on symmetry plane)
 - `tagType` with value `$spar` or `$rib`
 - `tagIndex` with different value for each spar and rib

```
# wing3
# written by John Dannenhoffer

# define the views
CFGPMTR  VIEW:Concept      1
CFGPMTR  VIEW:VLM          0
CFGPMTR  VIEW:CFDInviscid  0
CFGPMTR  VIEW:CFDViscous   0
CFGPMTR  VIEW:OmlStructure 0
CFGPMTR  VIEW:ClampedStructure 0
CFGPMTR  VIEW:SupportStructure 0
CFGPMTR  VIEW:BoxStructure  0

# define components to be used
CFGPMTR  COMP:Wing        1

# Design Parameters for OML
DESPMTR  wing:area        10.0    # wing area
DESPMTR  wing:aspect      6.00    # aspect ratio
DESPMTR  wing:taper       0.60    # taper ratio
DESPMTR  wing:sweep       20.0    # deg (of leading edge)
DESPMTR  wing:thickr      0.12    # thickness ratio at root
DESPMTR  wing:camber      0.06    # camber ratio at root
DESPMTR  wing:thickt      0.16    # thickness ratio at tip
DESPMTR  wing:cambert     0.02    # camber ratio at tip
DESPMTR  wing:alphat     -5.00    # setting angle at tip
DESPMTR  wing:dihedral    4.00    # deg
DESPMTR  wing:xroot       0.00    # xloc at root LE
DESPMTR  wing:yroot       0.00    # yloc at root LE
DESPMTR  wing:zroot       0.00    # zloc at root LE
```



```
# Design Parameters for structure
DESPMTR  wing:spar1    0.20      # location of fwd spar
DESPMTR  wing:spar2    0.70      # location of rwr spar
CFGPMTR  wing:nrib     3.00      # number of ribs per wing

# Define length units of the geometry
ATTRIBUTE capsLength  $ft

# convert VIEW:* variables into make* variables
UDPRIM   $/capsHeader

# wing local variables
SET      wing:span sqrt(wing:aspect*wing:area)
SET      wing:chordr 2*wing:area/wing:span/(1+wing:taper)
SET      wing:chordt wing:chordr*wing:taper
SET      wing:ytip   -wing:span/2
SET      wing:xtip   -wing:ytip*tand(wing:sweep)
SET      wing:ztip   -wing:ytip*tand(wing:dihebral)
SET      wing:mac    sqrt(wing:area/wing:aspect)
```

```
# make wing OML
IFTHEN    makeWingOml EQ 1
    # lay out left wing
    MARK
        # root
        UDPRIM    naca      thickness wing:thickr    camber    wing:camherr    sharpte    SHARP_TE
        SCALE     wing:chordr
        ROTATEX   90  0  0

        # left tip
        UDPRIM    naca      thickness wing:thickt    camber    wing:cambert    sharpte    SHARP_TE
        SCALE     wing:chordt
        ROTATEX   90  0  0
        ROTATEY   wing:alphat  0          0
        TRANSLATE wing:xtip    wing:ypip    wing:ztip
    RULE
        ATTRIBUTE tagComp $leftWing
    SET      ruledBody @nbody

    SELECT    FACE ruledBody 1
        ATTRIBUTE tagType $root
    SELECT    FACE ruledBody 2
        ATTRIBUTE tagType $tip
        ATTRIBUTE tagIndex $1
    SELECT    FACE ruledBody 3
        ATTRIBUTE tagType $upper
    SELECT    FACE ruledBody 4
        ATTRIBUTE tagType $lower
```

```

SELECT    EDGE    ruledBody 3 ruledBody 4 1
  ATTRIBUTE tagComp $leftWing
  ATTRIBUTE tagType $leadingEdge
IFTHEN    SHARP_TE EQ 0
  SELECT    FACE    ruledBody 5
    ATTRIBUTE tagType $trailingEdge
ELSE
  SELECT    EDGE    ruledBody 3 ruledBody 4 2
    ATTRIBUTE tagComp $leftWing
    ATTRIBUTE tagType $trailingEdge
ENDIF

# right wing too
STORE      LeftWing 0 1
RESTORE     LeftWing
  ATTRIBUTE tagComp $riteWing
  SELECT    FACE    $tagType $tip
  ATTRIBUTE tagIndex $2
  SELECT    EDGE    $tagType $leadingEdge
  IFTHEN    @iedge GT 0
    SELECT EDGE    $tagType $leadingEdge
      ATTRIBUTE tagComp $riteWing
  ENDIF
  SELECT    EDGE    $tagType $trailingEdge
  IFTHEN    @iedge GT 0
    SELECT EDGE    $tagType $trailingEdge
      ATTRIBUTE tagComp $riteWing
  ENDIF
CATBEG      $edge_not_found
CATEND
MIRROR      0    1    0
JOIN

```

```
SELECT    EDGE    ruledBody 3 ruledBody 3 1
          ATTRIBUTE tagType $root
SELECT    EDGE    ruledBody 4 ruledBody 4 1
          ATTRIBUTE tagType $root

STORE     WingOml
ENDIF

# make wing waffle
IFTHEN    makeWingWaffle EQ 1
  RESTORE  WingOml
  SET      xmin      @xmin-0.1
  SET      xmax      @xmax+0.1
  SET      ymin      0
  SET      ymax      @ymax+0.1
  SET      zmin      @zmin-0.1
  SET      zmax      @zmax+0.1
  STORE    .

  UDPARG   waffle     depth wing:nrib      # ensures rebuild
  UDPARG   waffle     depth wing:spar1
  UDPARG   waffle     depth wing:spar2
  UDPARG   waffle     depth zmax-zmin filename <<
```

```
# construction lines for spars
CPOINT A   AT      0+wing:spar1*wing:chordr 0
CPOINT B   AT  wing:xtip+wing:spar1*wing:chordt -wing:ytip
CPOINT C   AT      0+wing:spar2*wing:chordr 0
CPOINT D   AT  wing:xtip+wing:spar2*wing:chordt -wing:ytip

CLINE AB    A  B
CLINE CD    C  D

# rite spars
POINT E   ON  AB   YLOC  ymin
POINT F   ON  AB   YLOC  ymax
LINE  EF   E   F   tagComp=riteWing  tagType=spar  tagIndex=1

POINT G   ON  CD   YLOC  ymin
POINT H   ON  CD   YLOC  ymax
LINE  GH   G   H   tagComp=riteWing  tagType=spar  tagIndex=2

# rite ribs
PATBEG irib wing:nrib
    CPOINT I   AT  xmin  -wing:ytip*irib/(wing:nrib+1)
    CPOINT J   AT  xmax  y@I
    LINE  .    I   J   tagComp=riteWing  tagType=rib   tagIndex=!!val2str(irib,0)
PATEND
```

```
# root rib
CPOINT I AT xmin 0
CPOINT J AT xmax y@I
LINE . I J tagComp=rootWing tagType=rib tagIndex=0

# left spars
POINT E AT x@E -y@E
POINT F AT x@F -y@F
LINE FE F E tagComp=leftWing tagType=spar tagIndex=1

POINT G AT x@G -y@G
POINT H AT x@H -y@H
LINE HG H G tagComp=leftWing tagType=spar tagIndex=2

# left ribs
PATBEG irib wing:nrib
  CPOINT I AT xmin wing:ytip*irib/(wing:nrib+1)
  CPOINT J AT xmax y@I
  LINE . I J tagComp=leftWing tagType=rib tagIndex=!val2str(irib,0)
PATEND

>>
  TRANSLATE 0 0 zmin
  STORE WingWaffle
ENDIF

# now generate the needed views
UDPRIM $/capsViews

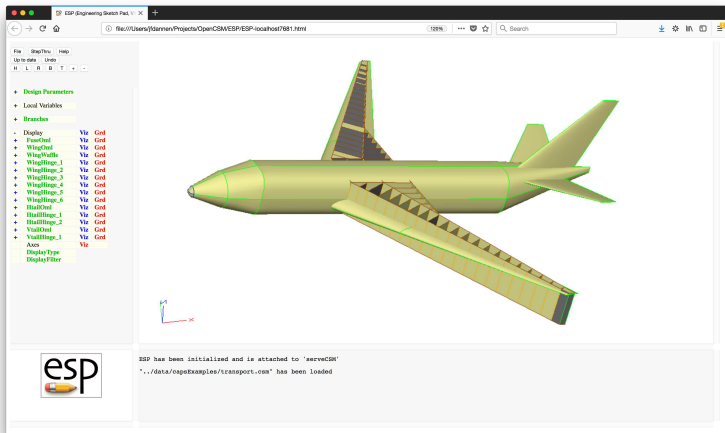
END
```



Full Transport Configuration

File found at `$ESP_ROOT/training/ESP/data/session10`

- Design Parameters associated with fuselage and tail
 - similar to wing
- Construction of fuselage and tail
 - similar to wing



- ESP is a powerful geometry-generating system that was designed for the analysis of complex configurations
 - supports multiple linked models
 - supports persistent attribution
 - provides sensitivities
 - can easily be coupled with other systems
- For CAPS, a set of “views” were defined; but these are only an example
- Each organization will want to develop a set of rules and conventions that are consistent with the rest of the organization’s design systems

- ESP is freely available for download from `acd1.mit.edu/ESP`
- Based upon user requests, new and improved features are added continually
- Send bug reports to `jfdannen@syr.edu` or `haimes@mit.edu`
- Also send success stories to `jfdannen@syr.edu` or `haimes@mit.edu`
- Thank you for attending; send comments about the course to `jfdannen@syr.edu` or `haimes@mit.edu`