

Engineering Sketch Pad (ESP)



Training Session 5 CSM Language (2)

John F. Dannenhoffer, III

jfdannen@syr.edu
Syracuse University

Bob Haimes

haimes@mit.edu
Massachusetts Institute of Technology
updated for v1.19

- Looping
 - PATBEG, PATBREAK, PATEND
- Logic
 - IFTHEN, ELSEIF, ELSE, ENDIF
- Signal Handling
 - THROW, CATBEG, CATEND
- Homework Exercises

- Patterns are like “for” or “do” loops
 - the Branches between the **PATBEG** and **PATEND** are executed a known number of times
 - at the beginning of each “instance”, the pattern number is incremented (from 1 to the number of copies)
 - one can break out of the pattern early with a **PATBREAK** statement
 - patterns can be nested within other patterns

- Example pattern (indentation optional):

```

PATBEG      i      3
      SET      j      i-1
      BOX      j      0  0  1  1  1
      ROTATEX  j*10  0  0
PATEND

```

- is the same as:

```

BOX      0  0  0  1  1  1
ROTATEX  0  0  0

BOX      1  0  0  1  1  1
ROTATEX  10 0  0

BOX      2  0  0  1  1  1
ROTATEX  20 0  0

```

- If/then constructs are used to make a choice within a `.csm` script
 - start with `IFTHEN` statement
 - has zero or more `ELSEIF` statements
 - has zero or one `ELSE` statement
 - has exactly one `ENDIF` statement
- The `IFTHEN` and `ELSEIF` statements have arguments, some of which can be specified in lowercase or UPPERCASE
 - `val1` — an expression
 - `op1` — can be `lt`, `le`, `eq`, `ge`, `gt`, `ne`, `LT`, ...
 - `val2` — an expression
 - `op2` — can be `or`, `xor`, `and`, `OR`, ... (defaults to `and`)
 - `val3` — an expression (defaults to 0)
 - `op3` — can be `lt`, `le`, `eq`, `ge`, `gt`, `ne`, `LT`, or ... (defaults to `eq`)
 - `val4` — an expression (defaults to 0)

- Example (indentation optional):

```
IFTHEN    a  eq  4  or  b  ne  2
          BOX  0  0  0  1  1  1
ELSEIF    c  eq  sqrt(9)
          BOX  2  2  2  2  2  2
ELSE
          BOX  3  3  3  3  3  3
ENDIF
```

- Note that only one of the BOX commands will be executed

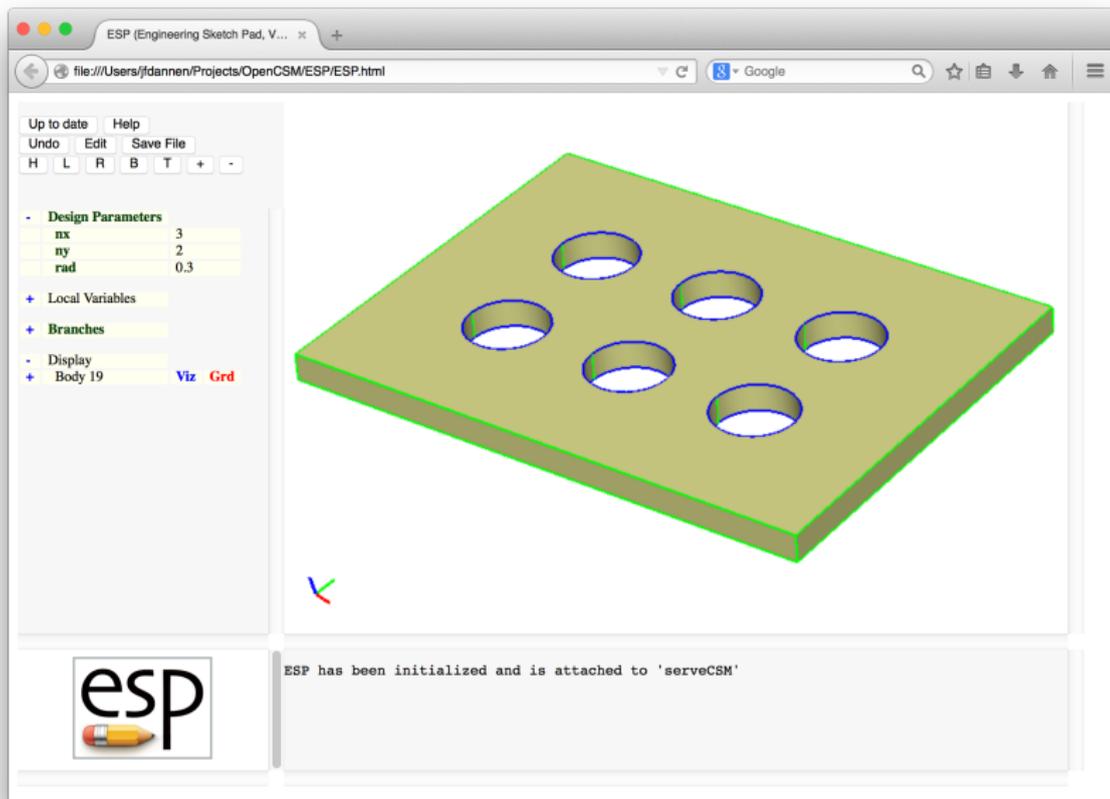
- Throw/catch constructs are used to generate and react to signals (errors)
- Signals can be generated by
 - executing a **THROW** command
 - ESP uses negative signal numbers, so users should generally use positive signal numbers to avoid collisions
 - a run-time error encountered elsewhere (see “help” for more info)
- When a signal is generated, all Branches are skipped until a matching **CATBEG** statement is encountered
 - the signal is cancelled
 - processing continues at the statement following the **CATBEG**
- If a **CATBEG** statement is encountered when there is no pending signal (or the pending signal does not match the **CATBEG**)
 - all Branches up to, and including the matching **CATEND** statement, are skipped

```
1: BOX      0 0 0 1 1 1
2: THROW   99
3: SPHERE  0 0 0 1
4: CATBEG  98
5:   SPHERE 0 0 0 2
6: CATEND
7: SPHERE  0 0 0 3
8: CATBEG  99
9:   BOX      1 0 0 1 1 1
10: CATEND
11: CATBEG  99
12:   SPHERE 0 0 0 4
13: CATEND
14: END
```

- BOX in line 1 is generated
- SPHERE in line 3 is skipped (since there is an active signal)
- CATBEG/CATEND in lines 4–6 are skipped (since they do not match 99)
- SPHERE in line 7 is skipped
- BOX in line 9 is generated
- CATBEG/CATEND in lines 11–13 are skipped (since the signal was cancelled when it was caught in line 8)

- Programming Blocks are delineated by
 - PATBEG and PATEND
 - IFTHEN, ELSEIF, ELSE, and ENDIF
 - SOLBEG and SOLEND
 - CATBEG and CATEND
- Any programming Block can be nested fully within any other programming Block (up to 20 levels deep)

- Rectangular plate with holes
- Round plate with holes
- Determine if two Bodies overlap
- Files in `$ESP_ROOT/training/ESP/data/session05` will get you started



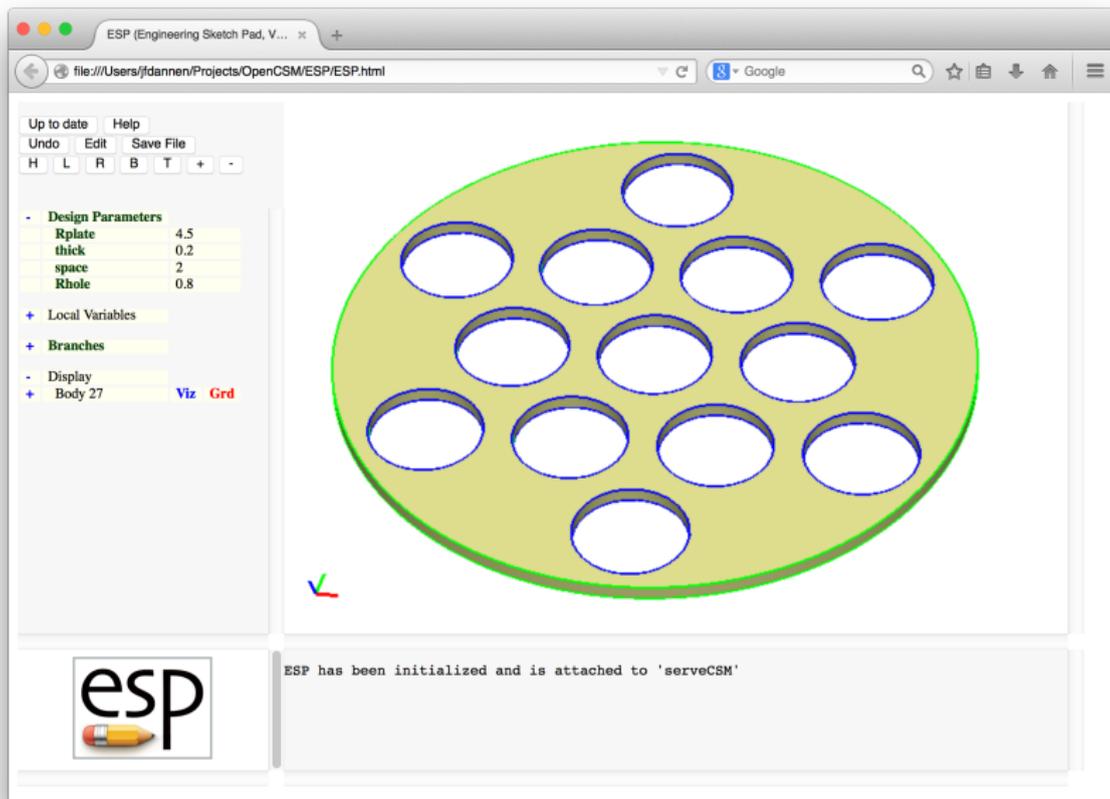
The screenshot displays the ESP (Engineering Sketch Pad) software interface. The main window shows a 3D model of a rectangular plate with six circular holes. The interface includes a menu bar with options like 'Up to date', 'Help', 'Undo', 'Edit', and 'Save File'. A toolbar below the menu bar contains icons for 'H', 'L', 'R', 'B', 'T', '+', and '-'. On the left side, there is a design tree with sections for 'Design Parameters', 'Local Variables', 'Branches', and 'Display'. The 'Design Parameters' section lists:

Design Parameters	Value
nx	3
ny	2
rad	0.3

The 'Display' section shows 'Body 19' with 'Viz' and 'Grd' options. The status bar at the bottom indicates 'ESP has been initialized and is attached to 'serveCSM''. The 'esp' logo is visible in the bottom left corner of the interface.

nx	number of holes in X -direction	3.00
ny	number of holes in Y -direction	2.00
rad	radius of each hole	0.30
	distance between hole centers	1.00

- Can you make a single hole in the center of the plate?
- Can you change your solution to have the holes spaced so that they fill the plate?
- What if you make the radius of the hole too big?



ESP (Engineering Sketch Pad, V...)

file:///Users/fldannen/Projects/OpenCSM/ESP/ESP.html

Google

Up to date Help

Undo Edit Save File

H L R B T + -

- Design Parameters

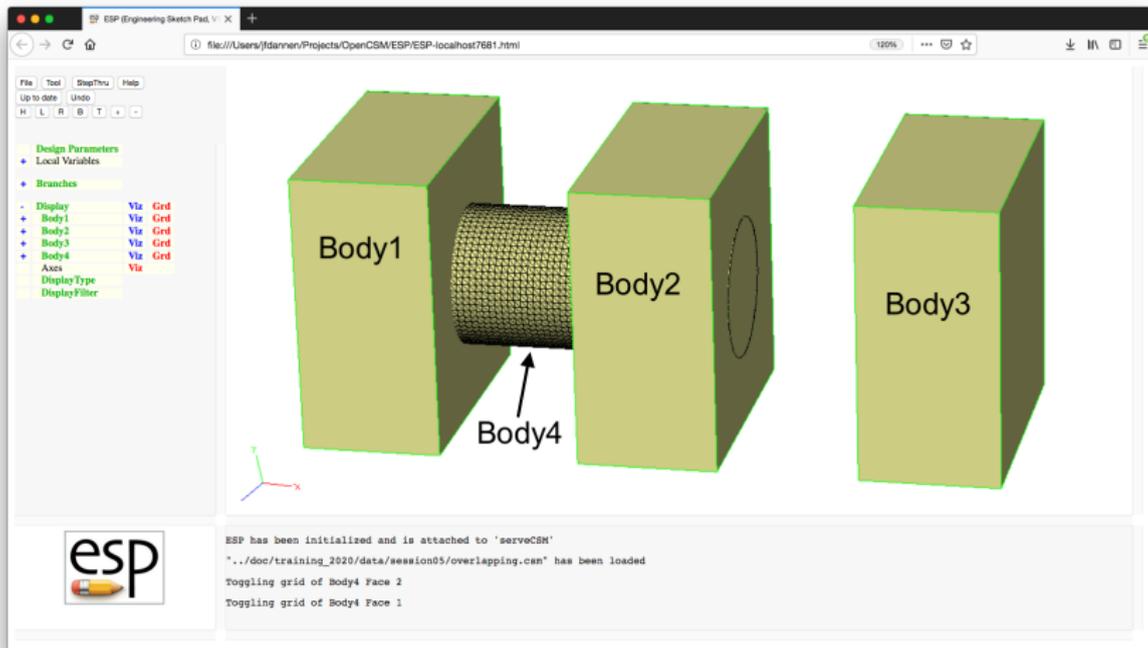
Rplate	4.5
thick	0.2
space	2
Rhole	0.8
- + Local Variables
- + Branches
- Display
- + Body 27 Viz Grd

ESP has been initialized and is attached to 'serveCSM'



Round Plate with Holes (2)

Rplate	radius of plate	4.50
thick	thickness of plate	0.20
space	distance between hole centers	2.00
Rhole	radius of holes	0.80
	number of holes selected automatically	



- Write `.csm` file to:
 - set `overlap1` to 1 if Bodys 1 and 4 overlap, otherwise set it to 0
 - set `overlap2` to 1 if Bodys 2 and 4 overlap, otherwise set it to 0
 - set `overlap3` to 1 if Bodys 3 and 4 overlap, otherwise set it to 0
- Try to use a pattern to do this compactly