External Connections

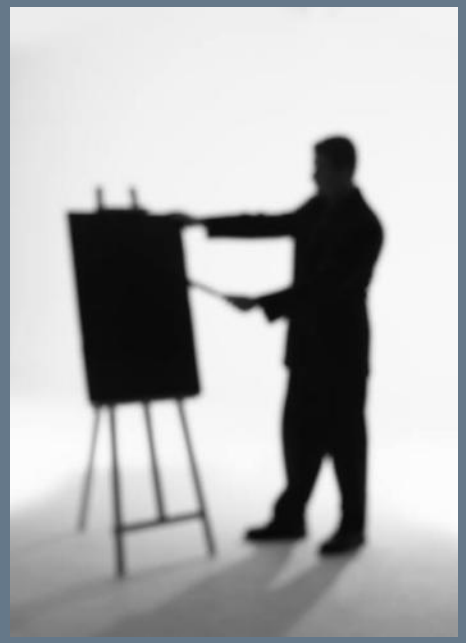
Eric Miller
Principal
Director, Simulation and Business Technologies

10/14/2011
PADT, Inc.

Agenda

- Note: This presentation is being recorded

- Introductions
- How it Works
- The XML File
- Example



Introductions

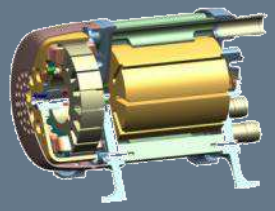
Upcoming Webinars

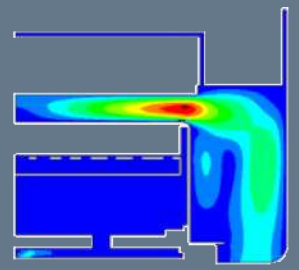
- Upcoming Webinars
  - Going to 2 a months instead of 3
  - ~~Aug 11, 2011: Parameter Management in Workbench~~
  - ~~Aug 25, 2011: Using Python with Workbench~~
  - ~~Sep 8, 2011: ANSYS Office Hours with PADT~~
  - ~~Sep 22, 2011: Using Excel with Workbench~~
  - ~~**Oct 14**, 2011: External Connections?~~
  - Oct 27, 2011: Important New Stuff in R14
  - Nov 10, 2011: Using Sub Modeling in ANSYS Mechanical
  - Nov 17, 2011: The In's and Out's of Load Steps with ANSYS Mechanical

- See upcoming and past webinars at:
  - padtincevents.webex.com
    - Click on ANSYS Webinar Series

About PADT

- PADT is an Engineering Services Company
  - Mechanical Engineering
  - 17 Years of Growth and Happy customers
  - 68 Employees
- 3 Business Areas
  - CAE Sales & Services
    - Consulting, Training, Sales, Support
  - Product Development
  - Rapid Prototyping & Manufacturing
- Learn More: www.PADTINC.com





"We Bring Dimension to Your Ideas"

Cube HVPC Systems

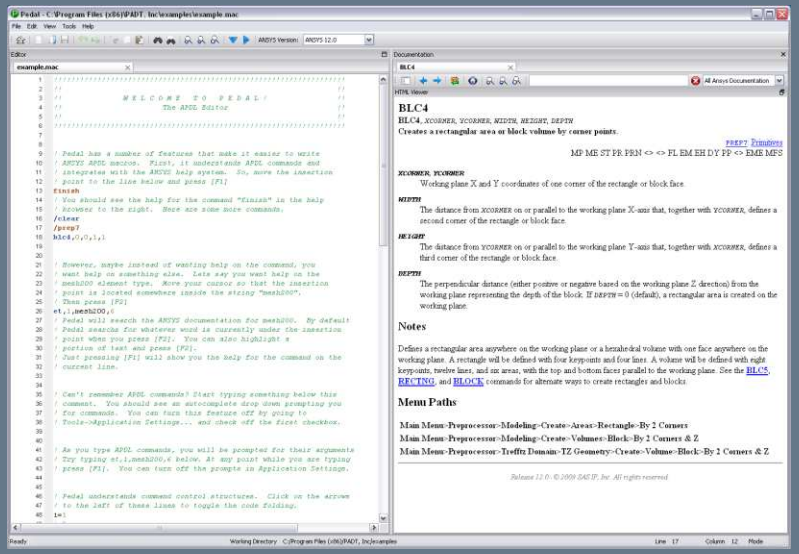
- Now Offering 12 (2x6) Core Intel Systems
- Balance between speed and cost
  - **Mini-Cluster**
    96 Cores / 256 GB RAM / 3.6 TB Disk
    Mobile Rack / UPS / Monitor / Keyboard
    $43,250
  - **Compute Server**
    32 Cores / 128 GB RAM / 3 TB Disk
    $12,300
  - **Simulation Workstation**
    12 Cores / 64 GB RAM / 1.5 TB D
    $5,800
  - **Simulation Fileserver**
    10 TB Disk / External eSATA
    $5,800

Cores	96 (2x4x12)	48 (4x12)	32 (4x8)	16 (2x8)	12 (2x6)	6 (1x6)	Fileserver (2x8)
System Name	c96SEi	c48	c32	w16	w12	w6	fs10
Price	$43,250	$15,500	$12,300	$11,600	$5,800	$5,400	$5,800
Configuration	2 x 2U Rack	1U Rack	1U Rack	Tower	Tower	Tower	2U Rack
CPU	2.3GHz AMD 6176	2.3GHz AMD 6176	2.4GHz AMD 6136	2.6GHz AMD 6140	2.8GHz AMD 4184	2.8GHz AMD 4184	2.0GHz AMD 6128
RAM (DDR3 1333)	256GB	128GB	128GB	128GB	64GB	32GB	16GB
OS Drive	256 GB SSD	256 GB SSD	256 GB SSD	256 GB SSD	256 GB SSD	320 GB Hybrid	-
Data Drives	3.6 TB 6 x 600 GB SAS2 15k RAID0	3 TB 3 x 1 TB 7.2K SATAII RAID0	3 TB 3 x 1 TB 7.2K SATAII RAID0	1.7 TB 3 x 600GB SAS2 15k RAID0	1.5TB 3 x 500 GB 7.2K SATAII RAID0	1.5TB 3 x 500 GB 7.2K SATAII RAID0	10 TB 8 x 1.5 TB 7.2K SATAII RAID60
NIC	2 x GigE	2 x GigE	2 x GigE	4 x GigE	2 x GigE	2 x GigE	2 x GigE
Video Card	Matrox 16MB	Matrox 16MB	Matrox 16MB	QuadroFX 580	QuadroFX 580	QuadroFX 580	Matrox 16MB
OS	CentOS Linux64	CentOS Linux64	CentOS Linux64	Windows 7 64	Windows 7 64	Windows 7 64	CentOS Linux64
Infiniband	QDR 40Gbps	-	-	-	-	-	-
Other	SAS2 Key Mobile Rack KVM Keyboard 17" LCD Monitor GigE switch 2x1.5 KW UPS's						External eSATA Dual Drive Bay w/ 3TB SATAII drive

- www.CUBE-HVPC.com

PeDAL – The APDL Editor

- Side-by-side editor and help viewer layout.
- Instant help on any documented APDL command by pressing F1.
- Full syntax highlighting for ANSYS v12 Mechanical APDL.
- Auto-complete drop downs for APDL Commands.
- APDL Command argument hints while typing commands.
- Search ANSYS help phrases and keywords.
- Multiple tabs for the editor and html viewer.
- Full capability web browser built in allows for rich web experience and web searches.



Connect with PADT

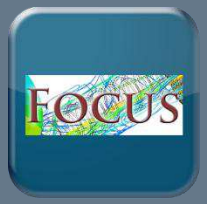
Facebook:
facebook.com/padtinc

Email Subscriptions:
www.padtinc.com/epubs

Twitter:
#padtinc

Web:
www.PADTINC.com

LinkedIn:
Search on PADT, Inc.



ANSYS User Blog:
padtinc.com/focus

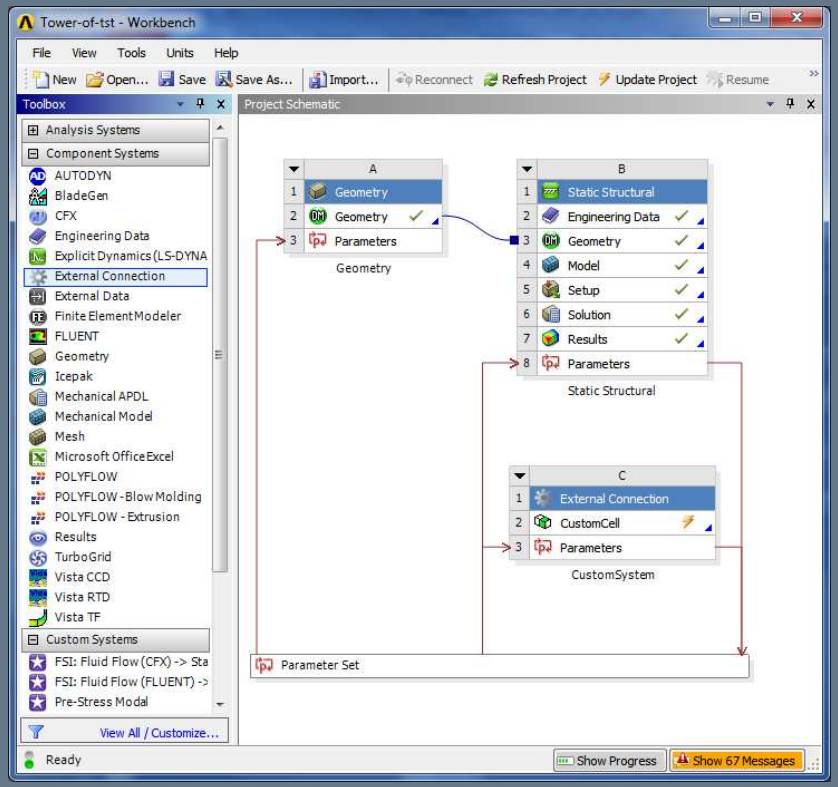
How it Works

Before We Begin

- Go to www.padtinc.com/thefocus
- Find Config file for External Connection Webinar posting
  - Click on the link for the file
  - Keep it up during webinar or print it out
- Thank you to ANSYS, Inc. Tech Support for providing a sample file that was modified for this webinar

What is an External Connection

- A way to add an external program to a project schematic through the exchange of parameters
  - The connection is NOT data compliant
- A way to add simple GUI commands to the Project Schematic



The Way It Works

- User creates an XML that defines things Workbench needs to know
  - More on this in coming slides
- User adds an External Connection system to their project
- User points the system to their XML file
- When project is updated
  - Workbench writes the requested input parameters into an input file
  - Workbench executes the external application
  - Looks to make sure execution was successful, error if not
  - Reads the output file for output parameters and updates the parameter manager.

What is in the XML File

- Required:
  - Information on Input and Output Parameters
    - Their name, type, and parsing rule for reading them
  - The name of a python script to be executed on an update
- Optional:
  - The name for the system and the cell
  - Version information
  - Working directory
  - Error File and error check string
  - Arguments passed to script
  - Interrupt and stop scripts
  - Various options
  - QUI Operations
    - C# code or python script to be executed

Documentation

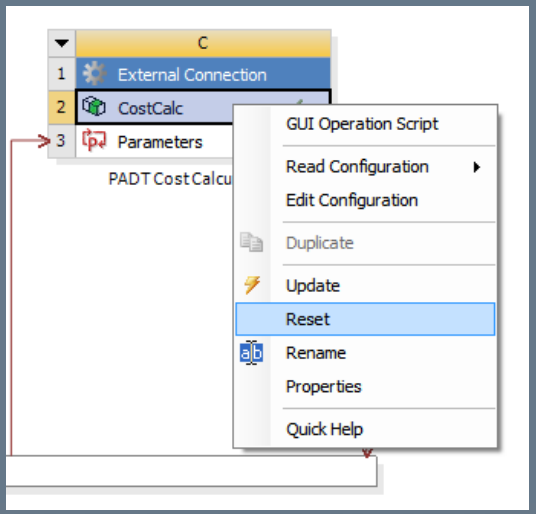
- Workbench // External Connection Add-in
- Fairly complete but written by a developer for a developer
  - If you don't know XML well, objects well, or how Workbench works well… a bit mystifying
  - Use example as starting point
  - Use Doc as reference

What You Will Need

- Sample program
  - A simple python script that reads and writes parameters for testing
- You real program
  - If it runs fast, use it to test and debug
- An XML editor
  - Too easy to make typos, you need something that highlights
  - You can use a text editor, but use one that recognizes XML
  - I have Adobe Dreamweaver and use that
  - VIM works just fine, but no “grouping”

Things to know

- To get input parameters from other systems
  - Set EC params equal to other system input params
- When you edit your file, the changes don't take effect till:
  - You RMB->Reset
  - RMB->Read Configuration



Outline of All Parameters

	A	B	C	D
1	ID	Parameter Name	Value	Unit
2	Input Parameters			
3	Geometry (A1)			
4	P1	W1	1	
5	P2	W2	1	
6	P3	Len	1.234	
7	PADT Cost Calculator (C1)			
8	P5	Length1	1.234	
*	New input parameter	New name	New expression	
10	Output Parameters			
11	Static Structural (B1)			
12	P4	Total Deformation Maximum	2.2544E-05	in
13	PADT Cost Calculator (C1)			
14	P6	OutputParameter	1.111	
*	New output parameter		New expression	
16	Charts			

Properties of Outline B8: P5

	A	B
1	Property	Value
2	General	
3	Description	
4	Error Message	
5	Expression	P3
6	Expression Type	Derived
7	Usage	Input

The XML File

Basic Structure

- All inside a Configuration node
- Three areas:
  - Instructions
    - Define input/output and execution
  - GUI Operations
    - What to add to the GUI
  - Properties
    - Custom data objects for Workbench
    - Ways for the user to specify values for the program besides input parameters

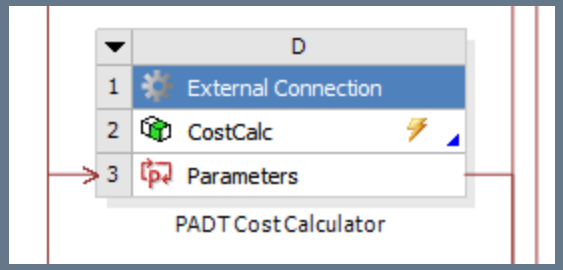
```
<Configuration>
   <Instructions>
      <Instruction>
            <Name></Name>

         <Args></Args>
         <ParameterParsingRules>
            <Parameter>
               <Rule></Rule>
               <Rule></Rule>
               ...
            </Parameter>
            <Parameter>... </Parameter>
         </ParameterParsingRules>
      </Instruction>... <Instruction>
      </Instructions>
      <GuiOperations>
      <GuiOperation>
         <Code> ... </code>
      </GuiOperation>
      </GuiOperations>
      <Properties>
      <DataEntity>
         <Object>
            <Property ... />
               </Object>
         </DataEntity>
         <DataEntity> ... </DataEntity>
      </Properties>
</Configuration>
```

Configuration Node

- <Configuration>
  - Main node that holds everything that Workbench needs to know about your external connection
  - Attributes are:
    - SystemName – This is the name assigned to your system block
    - CellName – The name for the cell in the block
    - Version – A Version for your file
    - ShowEditConfiguration (True/False) – Flag allowing edit ability of variables in the configuration file in the Workbench editor
- Example:

```
<Configuration
        CellName="CostCalc"
        SystemName="PADT CostCalculator"
        Version=""
        ShowEditConfiguration="True">
…
</Configuration>
```



Instructions

- <Instructions>
  - Holds the instructions that Workbench needs to deal with parameters and run your program
  - Attributes are:
    - Working Directory: The path to where the files used by the external connection are located
      - Very important
      - Makes the application of an external connection not so general
- Example

```
<Instructions
     WorkingDirectory=
            "C:\AAA_Work\seminars\ExternalConnections\ext1"
>
```

Instruction

- <Instruction>
  - Tells program what to do at initialization and during an update
  - Attributes are:
    - Type: Either Init or Update. When the instruction is to be executed
  - You can have multiple instruction nodes
  - Init's usually define the parameters input/output
  - Update's usually define the code to be executed
- Example:

```
<Instruction Type="Init"> … </Instruction>
<Instruction Type="Update"> … </Instruction>
```

Parameter Definitions

- Inside <Instruction Type="Init">
- <ParameterParsingRules>
  - No attributes, holds all the parameters
- <Parameter>
  - The definition for the parameter
  - Attributes:
    - Name: The Name for your Parameter, it shows up in the parameter manager
    - Type: Input or Output. Workbench uses this to put it in the right place
- <Rule>
  - Tell workbench how to handle the parameter
  - Attributes:
    - Type:
      - File: The name of the file. Can contain full path but by default looks in WorkingDirectory from <Instructions>
      - StartLine: The line number to start reading or writing data. This lets the program skip any headers you may have
      - PreString: The text preceding the value you want to read or write. Most commonly it is a parameter name and an equal sign (pval1=)
      - DataType: The type for the parameter. Must be: float, double, quantity (contains units)
      - SkipOccurences: The number of PreString values to skip before you get to the one you want
  - Value goes between Rule tags

Parameter Definitions

- Example:

```
<ParameterParsingRules>
     <Parameter Name="cLength" Type="Input">
            <Rule Type="File">costcalc.inp</Rule>
            <Rule Type="StartLine">1</Rule>
            <Rule Type="PreString">Length=</Rule>
            <Rule Type="DataType">Double</Rule>
     </Parameter>
     <Parameter Name="OutputParameter" Type="Output">
            <Rule Type="File"costcalc.out</Rule>
            <Rule Type="StartLine">1</Rule>
            <Rule Type="PreString">Cost=</Rule>
            <Rule Type="DataType">Double</Rule>
     </Parameter>
</ParameterParsingRules>
```

Update Definitions

- The other type of instruction: <Instruction Type="Update">
- Specify the script that gets run when you update
- Several tags, not nested
- Non-Critical tags inside <Instruction Type="Update">
  - <ExePath>: The path to the script, if different from the working dir.
  - <InterruptScript>: A script that will handle an interrupt from Workbench
  - <StopScript>: A script that will stop the main script if a stop is issued in Workbench
  - <CheckOnError>: A string to search for in the ErrorFile. If Workbench finds that line it will echo the line in the error message it displays
- Critical
  - <Name>: A name for the plugin, must be unique in case you have multiple
  - <Script>: The python script to be executed
  - <args>: Any command line arguments to pass into the script
  - <ErrorFile>: A file that, if it exists, tells workbench that things did not work.
    - Make sure you remove the error file, if it is there, WB will stop updating

Update Definitions

- Example:

```
<Instruction Type="Update">
    <Name>ScriptTest</Name>
    <Script>costcalc.py</Script>
    <ErrorFile>costcalc.err</ErrorFile>
    <CheckOnError>---ERROR---</CheckOnError>
 </Instruction>
```

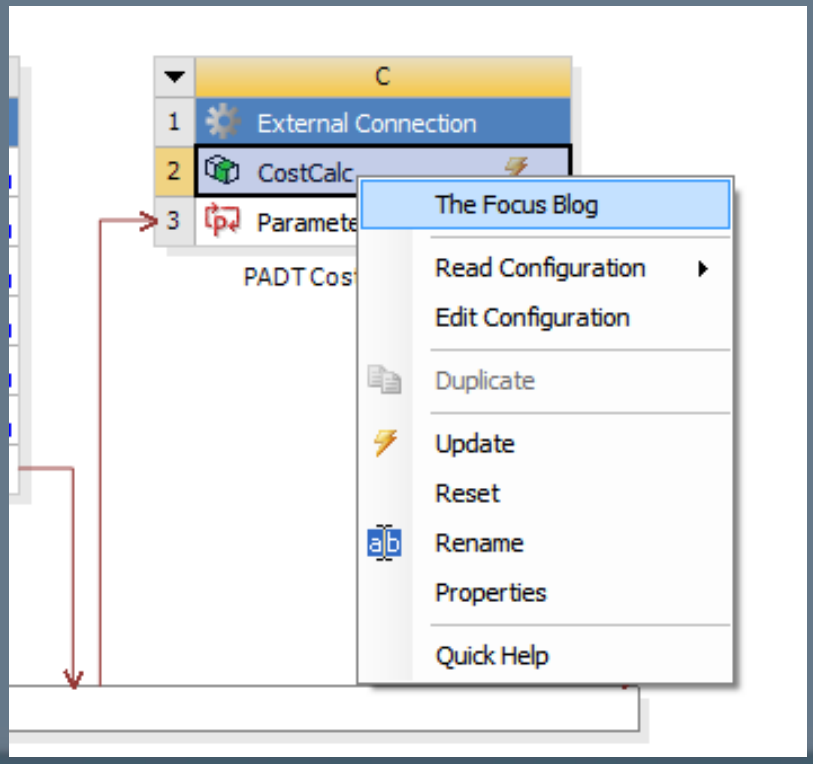
GuiOperations

- Allows you to put C# or python executable into the context menu or into the toolbar
  - I only got the context menu to work…
  - To get toolbar or menu to work you have to add it to the GUI, not in the XML file
    - That is a whole other topic…
- <GuiOperations>
  - Holder for multiple Gui additions
- <GuiOperation>
  - One for each addition to the menu

<GuiOperation>

- Required Attributes
  - Name: The text you want in the menu
  - Priority="2"
  - SourceType="Python"
  - ScriptFile: The name of the script you want run
- Option Attributes
  - See docuementation
  - You can execute C# commands if you want… I couldn't figure it out
  - Lots of other options
- Example:

```
<GuiOperations>
     <GuiOperation
     Name="The Focus Blog"
     Priority="2"
     SourceType="Python"
     ScriptFile="mtest.py" />
</GuiOperations>
```



Properties

- <Properties>
  - Allows user to input additional information, stuff that doesn't go into parameters well
  - Access by RMB->Edit Configuration or double-click on system
- <DataEntity> for each item you want to have
- <Object> holds properties together
- <Property> tags that define the name, type and value for each thing you want to have

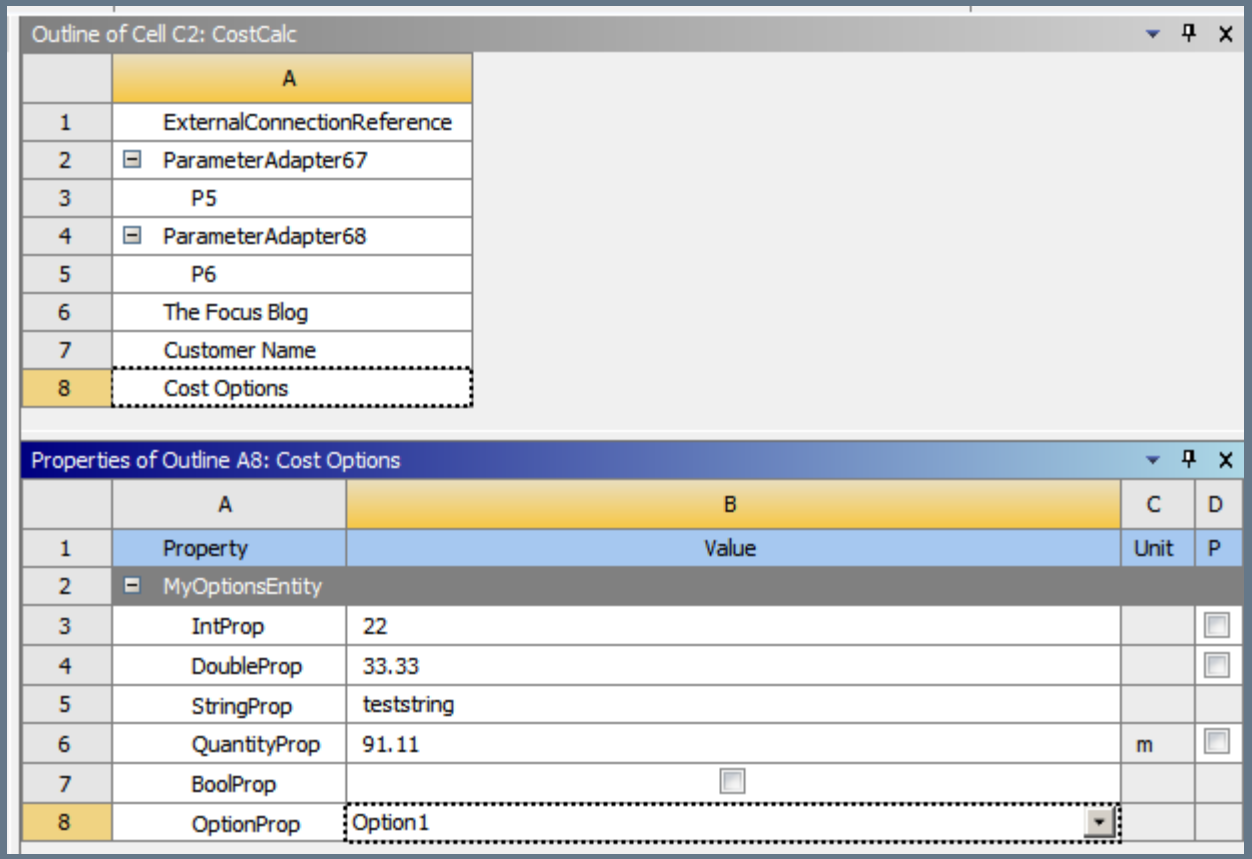
Properties

- Example:

```
<Properties>
     <DataEntity Name="CustNameEntity">
       <Object Name= "Customer Name">
         <Property Name="CustNameProp" DataType="string" Value="Enter Customer Name" />
       </Object>
     </DataEntity>
     <DataEntity Name="MyOptionsEntity">
           <Object Name= "Cost Options">
              <Property Name="IntProp" DataType="int" Value="22" />
              <Property Name="DoubleProp" DataType="double" Value="33.33" />
              <Property Name="StringProp" DataType="string" Value="teststring" />
              <Property Name="QuantityProp" DataType="quantity" Value="91.11 [m]" />
              <Property Name="BoolProp" DataType="bool" Value="False" />
              <Property Name="OptionProp" DataType="option" Value="Option1,Option2" />
            </Object>
        </DataEntity>
   </Properties>
</Configuration>
```

Properties

- Use: GetEntityProperties(Entity=configurationObject) in your python script to access properties.



Outline of Cell C2: CostCalc

	A
1	ExternalConnectionReference
2	⊟ ParameterAdapter67
3	P5
4	⊟ ParameterAdapter68
5	P6
6	The Focus Blog
7	Customer Name
8	Cost Options

Properties of Outline A8: Cost Options

	A	B	C	D
1	Property	Value	Unit	P
2	⊟ MyOptionsEntity			
3	IntProp	22		☐
4	DoubleProp	33.33		☐
5	StringProp	teststring		
6	QuantityProp	91.11	m	☐
7	BoolProp	☐		
8	OptionProp	Option1		

Example

Example

```
<Configuration CellName="CostCalc" SystemName="PADT Cost Calculator" Version="" ShowEditConfiguration="True">
   <Instructions WorkingDirectory="C:\AAA_Work\seminars\ExternalConnections\ext1">
      <Instruction Type="Init">
        <ParameterParsingRules>
           <Parameter Name="cLength" Type="Input">
              <Rule Type="File">costcalc.inp</Rule>
              <Rule Type="StartLine">1</Rule>
              <Rule Type="PreString">cLength=</Rule>
              <Rule Type="DataType">Double</Rule>
           </Parameter>
           <Parameter Name="OutputParameter" Type="Output">
              <Rule Type="File">costcalc.out</Rule>
              <Rule Type="StartLine">1</Rule>
              <Rule Type="PreString">Cost=</Rule>
              <Rule Type="DataType">Double</Rule>
           </Parameter>
        </ParameterParsingRules>
      </Instruction>
      <Instruction Type="Update">
         <Name>ScriptTest</Name>
        <Script>costcalc.py</Script>
                <ErrorFile>costcalc.err</ErrorFile>
                <CheckOnError>---ERROR---</CheckOnError>
      </Instruction>
   </Instructions>
   <GuiOperations>
       <GuiOperation Name="The Focus Blog" Priority="2"
           SourceType="Python"
                   ScriptFile="mtest.py" />
  </GuiOperations>
   <Properties>
     <DataEntity Name="CustNameEntity">
       <Object Name= "Customer Name">
         <Property Name="CustNameProp" DataType="string" Value="Enter Customer Name" />
       </Object>
     </DataEntity>
     <DataEntity Name="MyOptionsEntity">
           <Object Name= "Cost Options">
              <Property Name="IntProp" DataType="int" Value="22" />
              <Property Name="DoubleProp" DataType="double" Value="33.33" />
              <Property Name="StringProp" DataType="string" Value="teststring" />
              <Property Name="QuantityProp" DataType="quantity" Value="91.11 [m]" />
              <Property Name="BoolProp" DataType="bool" Value="False" />
              <Property Name="OptionProp" DataType="option" Value="Option1,Option2" />
            </Object>
        </DataEntity>
   </Properties>
</Configuration>
```

Example

- costcalc.py:

```
errFile = open('costcalc.err','w')

errFile.write ('---ERROR--- Damn!  Unknown Error in Execution')

outFile = open('costcalc.out',"w")
inFile = open('costcalc.inp','r')

for line in inFile:
   if 'cLength=' in line:
      theLength=line.lstrip('cLength=')

theCost = float(theLength)*34.52354


outFile.write('Cost='+str(theCost))

errFile.close()
outFile.close()
inFile.close()
os.remove('costcalc.err')
```

Thoughts

- Error files
  - You script should make an error file with maybe one line:
    - ---ERROR--- Program did not finish running
    - Use <CheckOnError>---ERROR---</CheckOnError>
  - If you encounter any other error, overwrite this with a more detailed error message
  - If your program runs successfully, then delete the file so Workbench will have a happy update
- Much more
  - This covered the basics
  - Once you are in python, there is much more you can do
  - You can use GuiOperations to bring up dialog boxes that prompt for more info or post process
  - Use a text file to store values for your program instead of parameters or properties

Thank You…

- PADT Enjoys doing these webinars…
- Please consider us as your partner
- ANSYS Related
  - Training, Mentoring
  - Consulting Services
  - Customization
  - Sales (if in AZ, NM, CO, UT, NV)
- Stratasys 3D Printers and Systems
- CUBE HVPC Systems
- Product Development
  - High-end engineering with practical, real world application
- Rapid Prototyping
  - SLA, SLS, FDM, PolyJet, CNC, Soft Tooling, Injection Molding
- Help us by letting us Help you