Customizing ANSYS Workbench

By,
Sudharshan Renganathan
Technical Service Engineer
Outline

Understanding the Workbench Framework

WB Journaling and Scripting

Different Customization Methods

Conclusion
Workflow Streamlining Using Design Points and Optimization

All of this and much more without any customization!
Need for Customization

- Capture the existing simulation process
- Make repetitive operations automatic
- Integrate CAE with other in-house analysis processes
- Make the technology available to a wider group (non CAE experts)
Component applications, covering various phases of the simulation process, sitting on top of a common framework
Native and Data-Integrated Applications

Native applications
- Built entirely on WB2 Framework
- Embedded within the “Workbench” window
- Project Schematic, Design Exploration, Engineering Data
- Fully supported by Workbench scripting
- Scripting language: Python

Data-integrated applications
- Share data and parameters with Workbench, native applications, and other data-integrated applications
- Independent UI, window
- E.g., Mechanical, Mechanical APDL, CFX, FLUENT, DesignModeler
- Scripting Language: JScript, Scheme, APDL …
Application-level Scripting
• For task automation at the application level
  – Mechanical, DM, Meshing: JScript
  – CFX: CCL
  – FLUENT: Scheme
  – MAPDL: APDL

Workbench Scripting
• For task automation at project level
  – Creating project, performing parameters simulations, optimization etc.
• Works “hand-in-hand” with scripting in DIAs
  – Can embed JScript, CCL, Scheme, APDL
Manually Record Journal
Edit Journal

Define all variables at top

```python
# Define all the variables
rootPath = "D:/Documents and Settings/rrath/Desktop/"
init_file = "initial.wbpj"
final_file = "solved.wbpj"
Paral = 15
Paral2 = 8

# Set the current session default folder
SetUserPathRoot(DirectoryPath=rootPath)

# Open the initial file
Open(filePath=AbsUserPathName(init_file))

# Access the 1st Design Point (DP0)
designPoint1 = Parameters.GetDesignPoint(Name="0")

# Access and Update the Parameter values
parameter1 = Parameters.GetParameter(Name="P1")
designPoint1.setParameterExpression(
    Parameter=parameter1,
    Expression=str(Paral))
parameter2 = Parameters.GetParameter(Name="P2")
designPoint1.setParameterExpression(
    Parameter=parameter2,
    Expression=(str(Paral2) + " [N]")
)

# Update the project with new parameters
Update()

# Save the project
Save(filePath=AbsUserPathName(final_file), Overwrite=True)
```
Replay Journal/Run Script
# Scripting in Data Integrated Applications

You can insert the ‘SendCommand’ call into your ANSYS Workbench scripts to drive these data-integrated applications.

<table>
<thead>
<tr>
<th>Application</th>
<th>Journaling</th>
<th>Scripting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mechanical APDL</td>
<td>APDL</td>
<td>Yes</td>
</tr>
<tr>
<td>Mechanical</td>
<td>JScript</td>
<td>Yes</td>
</tr>
<tr>
<td>DesignModeler</td>
<td>JScript</td>
<td>Yes</td>
</tr>
<tr>
<td>Meshing</td>
<td>JScript</td>
<td>Yes</td>
</tr>
<tr>
<td>FE Molder</td>
<td>JScript</td>
<td>Yes</td>
</tr>
<tr>
<td>AQWA</td>
<td>JScript</td>
<td>Yes</td>
</tr>
<tr>
<td>CFX</td>
<td>CCL</td>
<td>Yes</td>
</tr>
<tr>
<td>CFD Post</td>
<td>CCL</td>
<td>Yes</td>
</tr>
<tr>
<td>FLUENT</td>
<td>Scheme</td>
<td>Yes</td>
</tr>
<tr>
<td>PolyFlow</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>IcePak</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>AUTODYN</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

Recording WB journal will record the actions applied during the set up in these data integrated apps.
WB Journal with FLUENT

```
# encoding: utf-8
SetScriptVersion(Version="13.0")
template1 = GetTemplate(TemplateName="FLUENT")
system1 = template1.CreateSystem()
with Transaction():
    setup1 = system1.GetContainer(ComponentName="Setup")
    setup1.Import{
        FilePath="E:/Temp/cylinder.msh",
        FileType="Mesh"
    }
    component1 = system1.GetComponent(Name="Setup")
    component1.Refresh()
    fluentLauncherSettings1 = setup1.GetFluentLauncherSettings()
    fluentLauncherSettings1.SetEntityProperties(Properties=Set(EnvPath=()))
    setup1.Edit()
    setup1.SendCommand(Command="/define/models/viscous/ke-standard? y")
    setup1.SendCommand(Command="/define/boundary-conditions/velocity-inlet inlet no no yes yes no 5 no 0. n n y 2 5")
    setup1.SendCommand(Command="/define/boundary-conditions/pressure-outlet outlet no 0 no yes n n y 2 5 no no no")
    setup1.SendCommand(Command="/solve/initialize/initialize-flow")
    setup1.SendCommand(Command="(rsetvar 'number-of-iterations 200')")
    setup1.SendCommand(Command="it 200")
    setup1.SendCommand(Command="/close-fluent")
```
WB Journal did not capture operations performed in Mechanical

- Playing this journal file will not reproduce the complete project

Using `SendCommand` operations in Mechanical can be automated

- This will reproduce the project
Automation Scenario: CFD User

Scenario-C: Interested in automating some steps within FLUENT/CFX

- Examples:
  - Automate simulation setup
  - Automate post-processing in CFD Post
  - ...

- Use: CCL, Scheme etc.

Scenario-D: Interested in automating system level process involving FLUENT/CFX

- Examples:
  - Perform Design Points (DPs), Optimization with input parameters from in-house codes
  - ...

- Use: Python Script
Automation Scenario: Mechanical User

**Scenario-A**: Interested in automating some steps within Mechanical

- Examples:
  - Automate “Analysis Settings” from an Excel file
  - Generate contacts based on Named Selections
  - Apply transient Force/Moment loads from Excel
  - ...

- **Use**: JScript

**Scenario-B**: Interested in automating system level process involving Mechanical

- Examples:
  - Perform Design Points (DPs), Optimization with input parameters from MATLAB
  - Extract reports for all the DPs
  - ...

- **Use**: Python (+ JScript)
Automation Scenario: **MAPDL**

**Scenario-E**: Interested in automating some steps **within MAPDL**
- Examples:
  - Automate simulation setup
  - ...
- **Use: APDL**

**Scenario-F**: Interested in automating **system level** process involving MAPDL
- Examples:
  - Perform Design Points (DPs), Optimization with input APDL parameters
  - ...
- **Use: Python + APDL Script**
Different Customization Methods

Workbench:

• External Connection Add-in
  – Making external applications (not integrated with WB) to participate in workflow through parameters
  – Python scripting can be integrated to add functionality to WB (buttons, menu etc.)

• C# Add-in using Software Development Kit (SDK)
  – Integrate external applications in workflow through custom systems
  – WB GUI customization (addition of buttons, menu etc.)
  – Listening to various events in Workbench and performing actions
  – SDK offers much more access compared to External Connection

Data Integrated Applications (DIA):

• FLUENT: Scheme GUI, UDFs
• MAPDL: Tck/Tk GUI, UPFs
• DM, Meshing & Mechanical: JScript add-ins, Wizards
Customization Scenarios

**Scenario-G:** Integrate an in-house/third-party code with WB through parameters

- Examples:
  - Output parameter from a Mechanical simulation is the input parameter for the in-house code; Perform DPs involving the in-house code.
  - ...

- **Use:** External Connection

**Scenario-H:** Integrate an in-house/third-party code with WB (not just parameters are exchanged)

- Examples:
  - Use the Mechanical simulation results to perform a life analysis using in-house code
  - ...

- **Use:** C# Add-in (SDK)
Customization Scenarios (2)

- **Scenario-I**: Add a custom toolbar/menu on WB
  - Examples:
    - A WB menu to access internal best practice documents
    - A custom GUI to provide parameters for the simulation
  - Use: External Connection

- **Scenario-J**: Have an interactive workflow in Mechanical
  - Use: Wizard

- **Scenario-K**: Add custom toolbar/menu items within Design Modeler, Mechanical
  - Examples:
    - A toolbar button to select all bodies with same material
    - A toolbar button to prompt user for some inputs for
Functionality vs. Complexity

WB

Complexity

External Connection

Python Journal

Functionality

- Running WB from other software
  - MATLAB, Excel etc.
- Read/Write external data in WB

Integration of in-house/third-party code through parameters
- Hosting custom toolbar/menu, vertical applications

High-level integration of in-house/third-party code in WB workflow

DM, Meshing & Mechanical

C# add-in

Complexity

Wizard

Jscript add-in

Task automation in DM, Meshing, Mechanical etc.
- Create a workflow in DM, Meshing, Mechanical etc.
- Interface to take interactive inputs and task automation

Interface to take interactive inputs and task automation
- Add high-level functionality/GUI in DM, Meshing, Mechanical etc.
Different Automation Examples

Objectives

- This workshop demonstrates how to:
  - Create a JS add-in
  - Add a menu-bar item
  - Perform some task using this item
  - Enable/Disable the item programmatically
    - Enabled only when a body is selected
  - Register the add-in

- This workshop demonstrates how to:
  - Insert a predefined Command (the input)
  - Apply to a specified Selection
  - Specify the Selection

Part 1
- Inserting a predefined Command (the input)
  - Apply to a specified Selection
  - Specify the Selection

Part 2
- Insert a predefined Command (the input)

© 2010 ANSYS, Inc. All rights reserved.
Examples (1)

Example:
- Using External Connection, HFSS parameters and properties are available within Workbench
  - System acts as proxy for Ansoft HFSS application

Typical Usage:
- Integration of in-house/third-party code based on parametric input/output
Examples (2)

Example:
• Using External Connection, this simple vertical applications is built on R12

Typical Usage:
• Hosting a customized GUI to take inputs and drive the simulation
  - Impose restrictions/constraints in input parameters
  - Allow quick validity check for inputs and results
  - Allow user in decision making/perform branching in simulation process
• Integrate Microsoft Excel with Workbench process
  - Serve as a host for user inputs and simulation data management
Examples (3)

Example:
• Using Jscript add-in, Wizard functionality is added in DM

Typical Usage:
• Add high-level functionality in DM, Meshing, Mechanical etc.
• Add button/menu for extra features
  – Enable/disable dynamically
Examples (4)

Example:
• nCode is integrated in Workbench Project Schematic using C# add-in

Typical Usage:
• Integrate in-house/third-party codes deep in WB workflow
HFSS Workbench Integration

HFSS as a data-integrated application in R13

Satellite dish antenna – thermal-stress from resistive losses

HFSS in R12
Conclusion

ANSYS provides multiple methods to extend and customize Workbench

• Python scripting
• Application-level scripting
• External Connection add-in
• Workbench SDK

Different data-integrated applications also allow different types of customization

Power and complexity range from basic scripting to full programming

Choose method that best fits your needs
Future Direction

How do these components evolve?

• Python scripting
  – End-to-end “native” simulation applications are under active development
  – Enables end-to-end automation using only Python scripting

• Application-level scripting
  – Data-integrated applications will co-exist with native apps for many releases, investments made here will be useful for a long time

• External Connection add-in
  – Being extended to include data transfer in future releases

• Workbench SDK
  – As native simulation tools emerge, the SDK will enable development of full-featured custom applications
Thank you