RUTGERS
UNIVERSITY

CCD Report 2011-4
Cross Platform Design Simulation and Automation

April 20, 2011

Blake Koop
Center for Computational Design
Dept Mechanical and Aerospace Engineering
Rutgers University
98 Brett Road
Piscataway, NJ 08854
bkoop@eden.rutgers.edu, doyleknight@gmail.com

CROSS PLATFORM DESIGN SIMULATION AND AUTOMATION REPORT CCD 2011-04

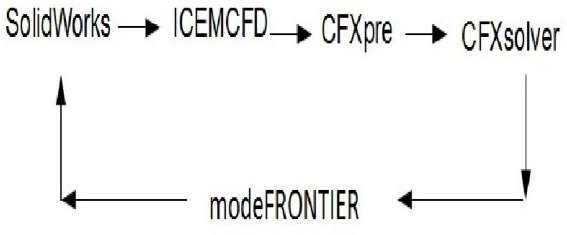
Blake Koop
Center for Computational Design
Dept Mechanical and Aerospace Engineering
Rutgers University, New Brunswick, NJ USA
bjkoop@gmail.com

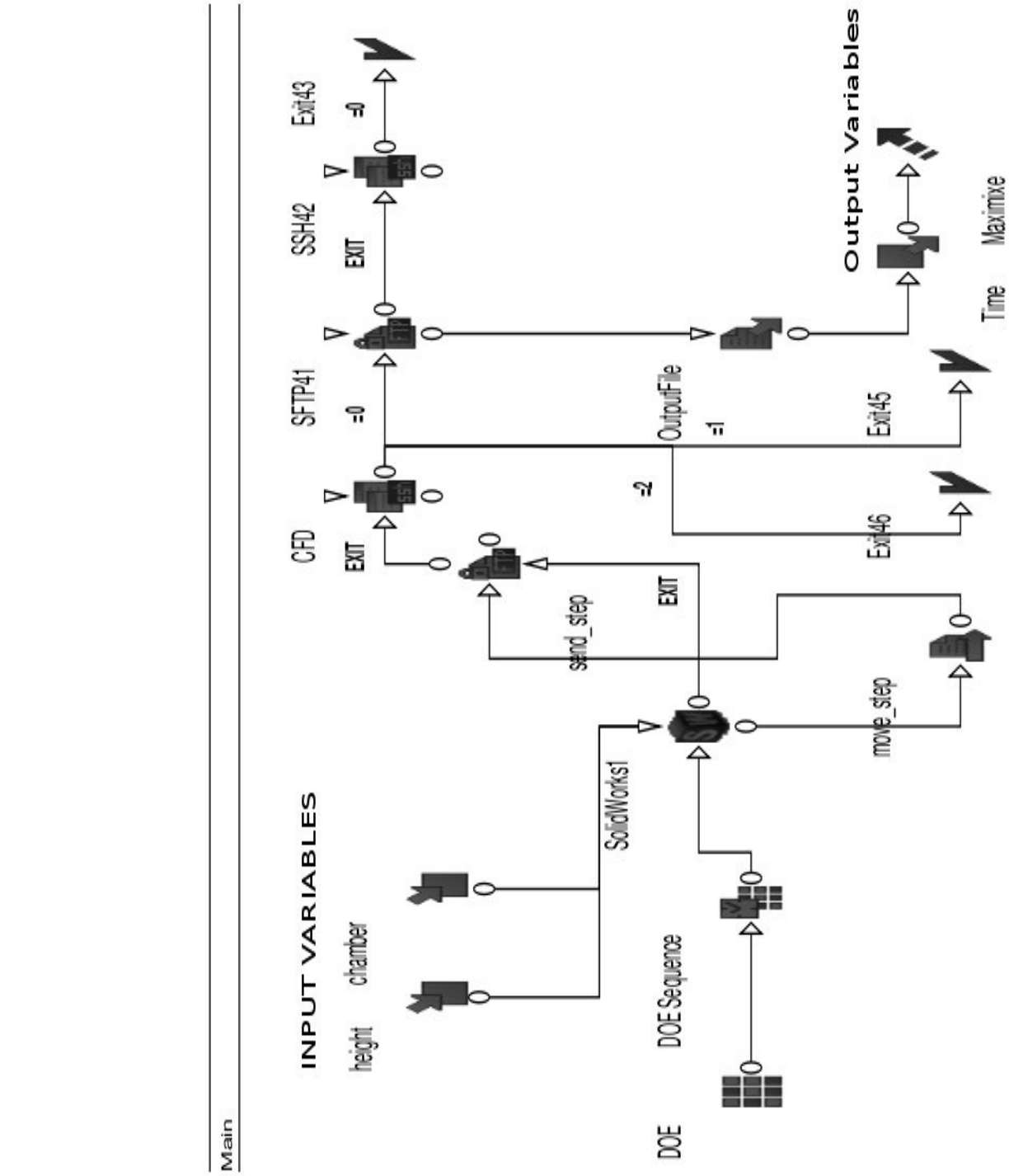
1. Introduction

A simplified gasifier design will be created and processed through a typical computational fluid dynamics sequence. This process will then be automated by creating a software loop in the multi-variate optimization design environment modeFRONTIER. This specific process described only varies the physical geometry of the design and focuses on the *process* by which this loop can be completed. The loop will be created on and executed by a Windows machine, but make use of the computational capability of a Linux machine.

2. Reference Figures

The first figure shows graphically the process outline. The second shows the complete workflow diagram. These will be helpful when moving through each step.




Main
INPUT VARIABLES
height
chamber
DOE
DOE Sequence
SolidWorks1
CFD
EXIT
SFTP41
=0
SSH42
EXIT
Exit43
=0
send_step
EXIT
OutputFile
=1
=2
Exit45
Exit46
move_step
Output Variables
Time
Maximize

3. Process Motivation and Outline

This report intends to inform the user on the *process* by which a cross platform CFD loop can be completed and automated. The specific design and variable optimization will be left for the user to apply to each unique project. Detailed here is a generally applicable process allowing users to function in Windows and then transfer files and commands to more powerful systems to expedite calculations.

The specific software packages used in the CFD analysis are:

Solidworks for geometry creation
ICEMCFD for mesh generation
CFXpre for boundary condition set up
CFXsolver for flow calculation
modeFRONTIER for software integration and loop execution

A basic model of a gasifier is built in Solidworks on a Windows machine, and then transferred as a *.step file to the Linux machine. The *.step file is then meshed in ICEMCFD using commands sent from Windows to Linux. This mesh is then imported into CFXpre where the inlet and outlet conditions are defined and the solver input file is written. The solver, CFXsolver, simulates the flow through the gasifier and creates a results file. This results file can be post processed in any way the user sees fit. All of the CFD steps excluding the geometry creation occur on the Linux side of the project, but are executed by commands initiated on the Windows side. ModeFrontier then restarts the loop, alters the geometry of the gasifier, in this case the height, and again simulates the flow conditions.

4. DOE and Sequence Nodes

The first node in the process flow shown in the workflow diagram is the DOE. This Design of Experiments node is responsible for creating all the variable values to be used in the design. There are several different ways to generate these values. This demonstration uses a predetermined set of values rather than the other variable generation models which would be more useful in a specific optimization. The sequence node then chooses what order these values are used in the optimization. Any specific project would have individual needs for sequence selection in order to run the most efficient optimization. This project followed the order of the predetermined DOE: 5 different height values for the gasifier ranging from 1 to 6 meters. It is important to understand here that these two nodes begin the *process flow*, or logic flow, of the experiment. The next section describes to beginning of the *data flow* of the experiment, and it is crucial to understand the difference between the two flows and how they intersect. This subtly gains the user insight into how the program functions and therefore is helpful in overcoming the steep learning curve. In this case, after the DOE chooses variable values and sends them to the scheduler to be ordered, the values are then sent to the SolidWorks node. The SolidWorks node will be described in a subsequent section.

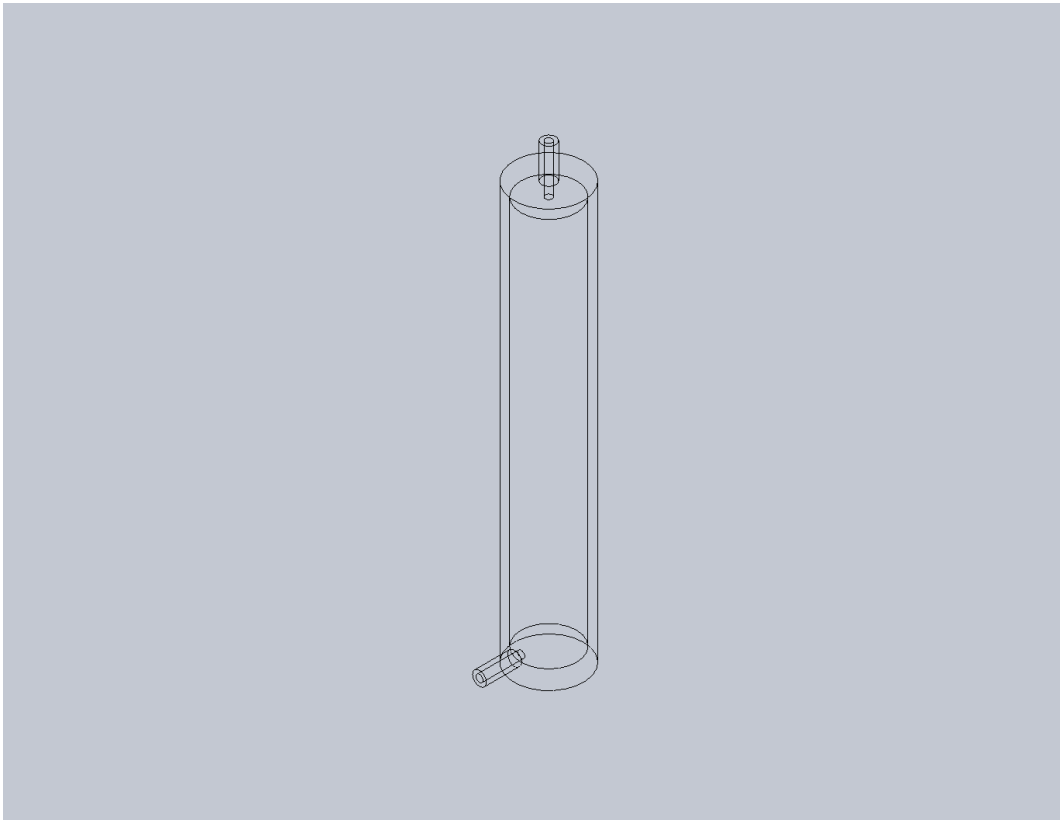
5. Input Variable Nodes

This portion of the workflow commences the data flow through the design process. It is here where the user specifies the name and upper/lower bounds of the variables. This project required two input variables due to the way the geometry was created: the height of the gasifier as well as the inner chamber height. At the next step in the process, these geoemtric values will be specified as variable entities. A requirement for this, due to the fact modeFRONTIER is only able to vary certain types of entities, is that the height values cannot be defined simply by a dimension. The dimension must be defined as an equation within SolidWorks. At this point in the project, there is no SolidWorks node, so the input variable node describes just name and outer limits of the variable values. They won't refer to a specific entity until the SolidWorks node is completed.

This project only varies the geometry of the simulation, but it should be known that essentially any value can be used as a variable. A previous project varied the input velocity of the air flowing throught the gasifier. Varying a flow condition such as this requires a bit more work, as there is no node integrated into modeFRONTIER for the software used to set the inflow conditions, CFXpre. What is done in that case, is to record a session in CFXpre and use this recording as an input file. The variable node is then connected to the input file node, and from within this input file the specific value to be varied can be highlighted. In the previous project, the input velocity was selected as a variable and allowed to range from 10 to 50 m/s. This makes modeFRONTIER much more powerful; it allows variable selection at almost any point during the CFD process.

6. SolidWorks Node

The SolidWorks node is the first intersection of the data and logic flows. As modeFRONTIER has fully integrated the SolidWorks package, a batch session recording is not necessary as it is in the CFX packages. All that is necessary is that the desired variables be defined by equations when the geometry is created. If SolidWorks is being used for more than just geometry creation, that is if it is being used for physical analysis, more variable options become available, but this project makes use of more powerful analyzers so this option is not explored. Here, all that is needed is for the dimensions of both the outer height and inner chamber are defined by an equation. Specify the *.SLDPRT file within the node, and modeFRONTIER does the rest; the selectable variables are found from within the geometry and presented to the user. The process flow then moves to the SFTP node where the files can be transferred to the other machine, while the data flow moves through a file transfer node. Because the next step in the CFD requires an input file in *.step form, the SolidWorks exort setting is set to 'export as STEP'. It is this *.step file that in transferred between machines.



7. Transfer Nodes

This is the point in the project where files and commands are moved from the Windows machine with SolidWorks to the Linux machine running the CFD package. The data flow simply requires a transfer file node. This takes the *.step file created by SolidWorks and moves that data to the SFTP node, which actually sends that data between machines. The file transfer node requires that the name of the file to be moved agree with the name of the initial geometry file, but with a *.step extension rather than a *.sldprt due to the export setting selection. The base directory option should be kept as the local directory, '.'.

The SFTP node establishes a connection between the machines and sends the *.step file. Within the SFTP dialog box, identify a host name, username, and password and then test the connection. Once a connection is established, the "PUT" action needs to be specified, along with the local file to be transferred and the remote directory location. The SFTP node can be thought of as a data flow process for the Linux machine. At this point, the *.step file is on the Linux machine and the CFD process can be initiated.

8. SSH Node

The function of the SSH node is to send commands from the machine running modeFRONTIER to the other machine to be executed. This node can be thought of as the logic flow for the Linux machine. The transfer nodes have sent the necessary *.step file, and now a script must be written to execute the CFD procedure in such a way that the process can be repeated without any more input from the user. The first command is just to set the directory. Moving between systems and softwares can lead to working directory discrepancies; it is more convienient to use a **cd** command at the begining of scripts to ensure that files are being read/written to or from the necessary directory. The entire script for this project goes as follows:

```
cd /home/bkoop

rm /home/bkoop/project2002.out
rm /home/bkoop/project2002.res

/home/ansys12.1/v121/icemcfd/linux64amb/bin/icemcfd -batch /home/bkoop/icemscript.rpl
/home/ansys12.1/v121/CFX/bin/cfx5pre -batch /home/bkoop/project1pre2.pre
/home/ansys12.1/v121/CFX/bin/cfx5solve -def /home/bkoop/project2.def
```

8.0.1. *Directory Cleaning.* The next step is to remove the created results files. This step becomes necessary due to the naming defaults in modeFRONTIER. The program is designed to finish a project with all of the results files intact and readable, and to that end each individual design gets a unique number which is attached to the end of the file name. For example, project.res becomes project001.res on the first run and project002.res on the second and so on. This naming method is not useful for our purposes because we need to call on each results file to extract an output variable at a later point in the loop. This is easiest to do if the name of the file remains the same for every design. modeFRONTIER searches the working directory for a previous results file when naming each design, and chooses the number suffix accordingly. To ensure that the name becomes what we need it to be, a **rm** funtction is used in the shell script to clean out the directory just before the CFD portion of the process is initiated. There may be built in ways to achieve this, but this method is not terribly inconvienient and is effective for our purposes.

8.0.2. *Mesh Generation.* The next step is to generate a grid using ICEMCFD. This is done by recording a session in ICEMCFD of the user creating the desired grid from the *.step file. The recording file format is *.rpl, and will serve as the step by step guide for mesh generation in the subsequent design loops. The inlet and outlet surfaces need to be defined at this point so that they can be called on in the CFXpre file to be defined as boundaries. CFXpre easily recognizes parts defined in the mesh, so both the inlet and outlet were defined as surfaces, and those surfaces were defined as parts. Mesh generation can be a difficult part of this process, and

there will most likely be project specific difficulties that arise. With the grid geneartation session file recorded, the CFXpre phase can be undertaken.

8.0.3. *Boundary Condition Set-up.* Again a session file recording is necessary. The difficulty here is that when recording a session in CFXpre, the default boundary conditions established by the program need to be specified by the user. This is not necessary when running the program regularly. The easiest way to generate the default settings in a recording is to simply record a session without worrying about this problem, and then copy those default setting lines from a previous *.pre file which was successfully used. After this, depending on geometry, the CFXpre session can be fairly simple. This project only required defining the inlet/outlet conditions, and that the fluid in question was air at STP. The last step in the session recording is to define the solver input file, which leads to the next software phase, CFXsolver.

8.0.4. *Solving the Flow.* Writing the solver input generates a *.def file. All that needs to done from this point is to open that *.def file with the solver from within the SSH script. The solver will run for number of iterations specified or to the residual convergence specified when the *.def was written. At this point 2 session files have been recorded, one for ICEMCFD, and one for CFXpre. The *.step file has already been deposited in the working directory of the remote machine, so when the ICEMCFD session file is told to execute in the SSH script, the result is a mesh of the geometry in the form of a *.cfx5 file. Then the SSH script executes the CFXpre script, which imports the *.cfx5, establishes boundary conditions, and generates a *.def file. Finally, CFXsolver is called on to open the *.def and solve the flow. This produces the desired *.out file. It is this *.out file which will be used to find an output variable.

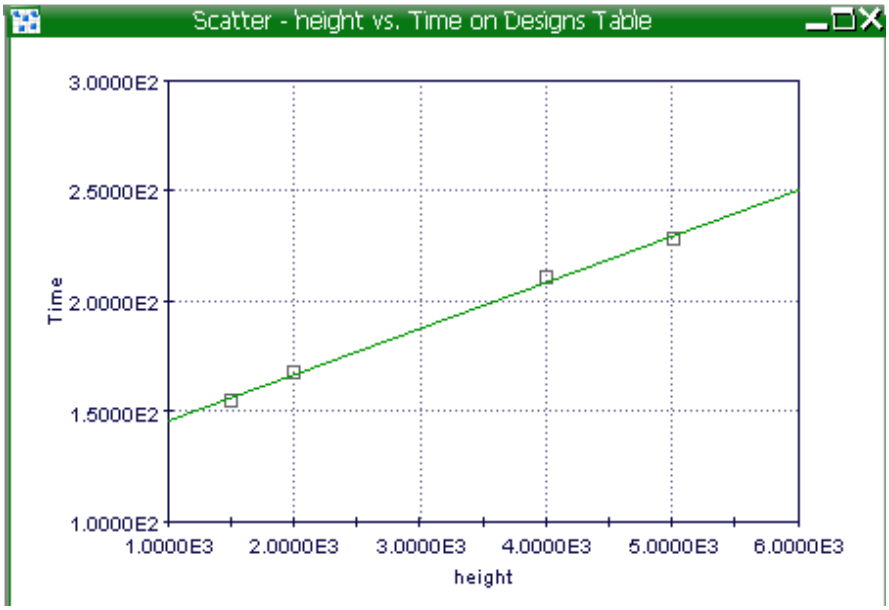
Due to the sensitivity of the mesh geneartion, at this point a security measure was put in place. Mesh generation is geometry dependant, and should be tailored to each specific geometry when looking for the most accurate results. Because this process uses a generic mesh technique and varying geometries, it is subject to failure at the meshing step. To guard against loop termination, three exit options are put in place here rather than just 1. The error free exit status moves forward with the process as intended, but error returns 1 and 2 are led to exit nodes. The exit nodes direct the loop to simply restart with the next design value. In this design experiment, four of five designs completed regularly and one failed at mesh generation.

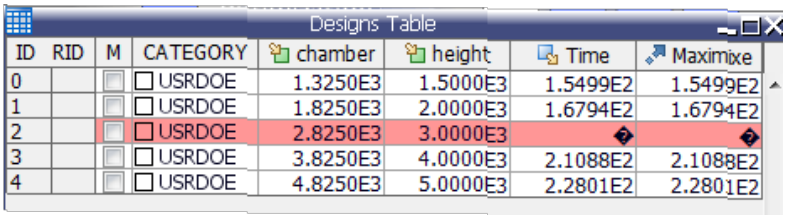
9. File Return and Variable Selction

Now the *.out file is on the remote machine, and it needs to be transferred to the machine running modeFRONTIER. Another SFTP node is used, but this time the action is set to "GET" rather than "PUT". The remote file is the *.out file and the local file should be set to "./" which indicates that the *.out should be placed in the current design directory. Now both the process and data flow have returned

to the local machine. The process flow returned as the SSH script finished, and the data returned with the SFTP "GET" action.

Now that the output file is on the local machine, it is used as a template for output variable selection. The output file node should be put in place and set up to use the *.out file as a template. This requires either running the loop once to generate the *.out file locally or moving it between the machines by some other method. Within the *.out file, the desired variable can be found; in this project that variable is total advection time. Because the solver was set to run 25 iterations, each *.out file has the same number of rows and columns, and this allows to define the location of the output vairiable by what ModeFROTNIER refers to as the 'absolute' location. Simply highlight the desired quantity from within the output file and set the mining rule to absolute. Each time the loop completes, the output variable is stored and can be optimized however needs may dictate. The results from this project are shown below. The time, meaning total advection time in the chamber, is measured in seconds and height in milimeters.





Designs Table

ID	RID	M	CATEGORY	chamber	height	Time	Maximixe
0			USRDOE	1.3250E3	1.5000E3	1.5499E2	1.5499E2
1			USRDOE	1.8250E3	2.0000E3	1.6794E2	1.6794E2
2			USRDOE	2.8250E3	3.0000E3	[illegible]	[illegible]
3			USRDOE	3.8250E3	4.0000E3	2.1088E2	2.1088E2
4			USRDOE	4.8250E3	5.0000E3	2.2801E2	2.2801E2