F-Chart Software

No. 49, Fall 2025

EESy Solutions

Engineering Equation Solver Newsletter

Inside this issue:

Welcome

This is the 49th issue of EESy Solutions, a newsletter that provides news, tips, and other updates for users of the Engineering Equation Solver software. This issue describes a new feature that allows other programs to call the EES Professional app. The process uses the FindWindow and SendMessage Windows commands and can therefore be carried out from any application that supports these commands.

The $IGMixture directive is also presented. $IGMixture allows you to define a custom ideal gas mixture that can then be used like any other fluid in property routines or the Component library. The new EES Community Forum is presented and some new features in the EESyGrader software are discussed.

EES has been commercially available for more than two decades. Previous issues of EESy Solutions can be downloaded from https://fchartsoftware.com.

Sending Messages to EES

There are macro commands available that allow EES to start and interact with other applications such as Python, Excel, and MATLAB. However, these commands require that EES be the program controlling the interaction. There may be times when you would like another application to control EES. Dynamic Data Exchange (DDE) provided a method to accomplish this, but DDE is no longer supported by most programs. Starting with version 12.131, Professional EES Licenses will respond to messages from other applications that are sent with the SendMessage command using the Windows Operating System. These messages will instruct EES to either Solve or Run a Macro in an open EES program.

The approach for using this technique is to first open EES and load the file that you want to run. The calling application exports the information that needs to be passed to EES in a data file. When EES receives an instruction to either Solve or Run a Macro, it will read the file, carry out the calculations that are programmed, and finally export the outputs which can be read by the calling application.

This approach is relatively straightforward and easy to debug as both sides of the interaction can be programmed and tested in isolation. The method is also relatively fast since EES does not need to start each time a message is sent.

The FindWindow Command

In order to establish communication with EES you must first obtain a handle to an open instance of the EES program. This is done using the FindWindow command:

hWnd = FindWindow('TEES_D','')

where the first argument is the string 'TEES_D' which indicates that you are looking for the EES program and the second argument is a null string.

The SendMessage Command

Once a handle to the EES program is obtained, commands are sent using the SendMessage command. The protocol for the SendMessage command is:

Result = SendMessage(hWnd, wm_RunEES, em_Command, xtra)

where hWnd is the handle to the EES program, wm_RunEES = 32777 is a flag that is recognized by EES, em_Command is an integer that specifies the action to be taken by EES, and xtra is an extra parameter that should be set to 0 unless em_Command = 20 (corresponding to Run a Macro).

The various commands recognized by EES are summarized below. The first three commands manipulate a log file that records the commands executed by EES. The log file is a text file that resides in the the EES program folder named EESCallFromApp.log.

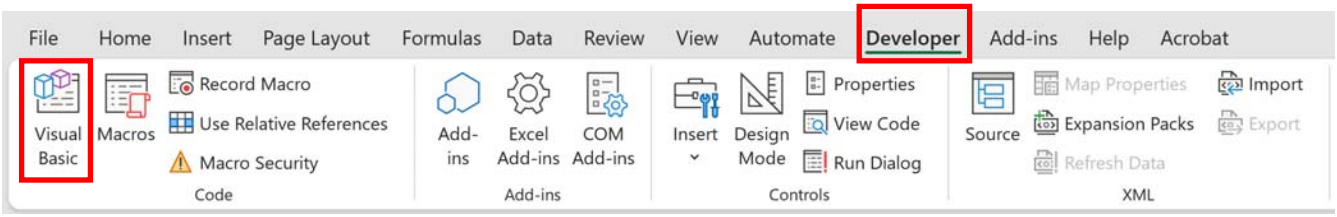
Integer	em_Command Parameter	Summary
0	em_DeleteLog	Delete the log file EESCallFromApp.log
1	em_LogOff	Turn off logging to the log file EESCallFromApp.log
2	em_LogOn	Turn on logging to the log file EESCallFromApp.log
10	em_Solve	Solve the open EES file
20	em_RunMacroTab	Run the macro in the open EES file that is contained in the tab specified by the parameter xtra

The SendMessage command returns 0 if there was no error running EES. Otherwise the command returns the error that occurred. The error is also written to the log file.

An Example of Calling EES from Excel

This example shows how to call EES from Excel using the SendMessage Windows command. There is a video as well showing the steps. The Excel and EES code can be found here.

Start Excel and select the Developer menu to access the Visual Basic option from the ribbon.



Once you have Microsoft Visual Basic open, you will want to import the ExceltoEES.bas file containing the EES-to-Excel utilities. The file includes declarations of the SendMessage and FindWindow commands as well as definitions for the constants used in these commands, as shown next. Two utility functions are also present.

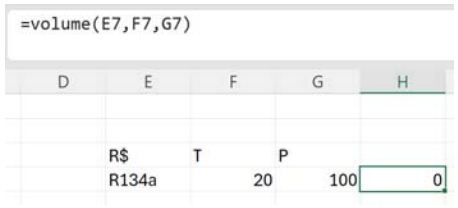
```
Public Declare PtrSafe Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hwnd As LongPtr, ByVal Msg As
Public Declare PtrSafe Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal lpClassName As String, ByVal lp
Public Const hwnd As LongPtr = 0
Public Const wm_RunEES As LongPtr = 32777
Public Const em_LogOff As LongPtr = 1
Public Const em_LogOn As LongPtr = 2
Public Const em_Solve As LongPtr = 10
Public Const em_RunMacroTab As LongPtr = 20
Public Const notUsed As LongPtr = 0
Public Result As LongPtr

Sub DeleteFile(FileName)
  On Error Resume Next
  Kill FileName
  On Error GoTo 0
End Sub

Function OutputFileExists(filePath As String) As Boolean
    OutputFileExists = (Len(Dir(filePath)) > 0)
End Function
```

We will do something simple here which is to use EES to calculate a thermodynamic property for a specific fluid at a given temperature and pressure. Let's define a function called Volume that takes in three inputs (a string for the fluid and two doubles for the temperature and pressure) and returns one output (a double for the specific volume). Within function Volume, we will first call the subroutine SetInputs that writes the inputs to a file in a folder used for communication designated by the global string Folder. The subroutine SetInputs writes the input string (with single quotes appended) and the input doubles to the file State.dat in the designated folder.

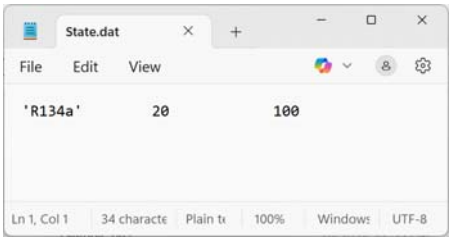
```
Public Const Folder As String = "C:\ExcelExample1"

Sub SetInputs(R As String, T As Double, P As Double) 'user needs to export inputs to EES as required
    Dim RR As String
    RR = "'" + R + "'"
    Open Folder + "\State.dat" For Output As #1
      Print #1, RR, T, P
    Close #1
End Sub

Function Volume(R As String, T As Double, P As Double) As Double
    Call SetInputs(R, T, P)  'write inputs to a file for EES
End Function
```

An Example of Calling EES from Excel (continued)

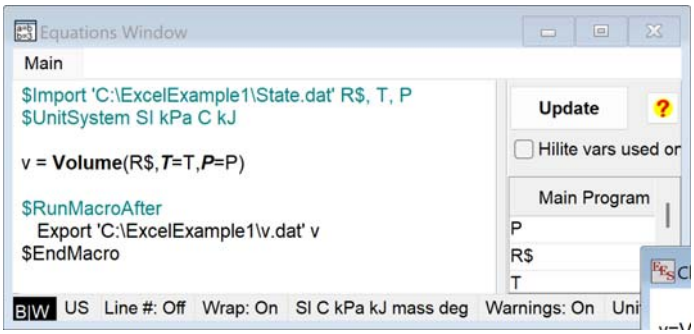
We can test this part of the process by returning to the spreadsheet and calling the function Volume just as you would any other function. While the function doesn't yet return the specific volume, it will write a file in the communications folder called State.dat. If you open the file you will see the fluid, temperature and pressure that you specified.

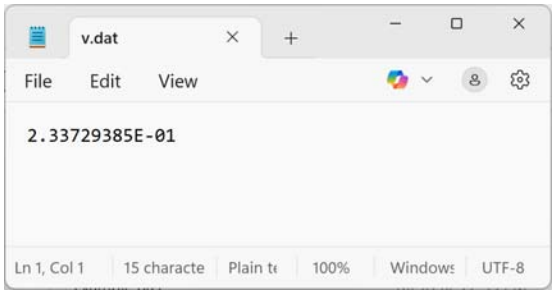


Now that the inputs have been written, we can turn our attention to building an EES file that reads the inputs from the file State.dat, calculates specific volume, and writes the result to an output file v.dat. We will do this using directives **$Import** (which runs before calculations are executed) and **$RunMacroAfter** (which runs after the calculations are completed) .



The resulting EES code is shown below. This code can be tested in isolation since the file State.dat already exists. Solve the code and you will see that a new file has been created, v.dat, that contains the calculated specific volume.





All that remains is to make our Excel function send the message to EES to Solve, read the contents of v.dat and finally return it. We'll add to our Volume function a FindWindow command to identify the open instance of EES. If the command fails (returning a 0) a message is displayed indicating that you need to start EES and then the function is terminated. Otherwise messages are sent to EES instructing it to delete the log file (em_DeleteLog) and start logging commands (em_LogOn). The Solve (em_Solve) is then sent. It will either return an error if the value of Result is not 0 or use the GetOutput function to retrieve the stored value of specific volume.

```
Function Volume(R As String, T As Double, P As Double) As Double
    Call SetInputs(R, T, P)  'write inputs to a file for EES

    hWndApp = FindWindow("TEES_D", vbNullString)
    If (hWndApp = 0) Then
        MsgBox "EES is not running"
        Volume = -999
        Exit Function
    End If
    Call SendMessage(hWndApp, wm_RunEES, em_DeleteLog, notUsed)
    Call SendMessage(hWndApp, wm_RunEES, em_LogOn, notUsed)

    Result = SendMessage(hWndApp, wm_RunEES, em_Solve, notUsed)
    If (Result <> 0) Then
        MsgBox "EES returned Error #" + Str(Result) + ""
    Else
        Volume = GetOutput()
    End If

End Function
```

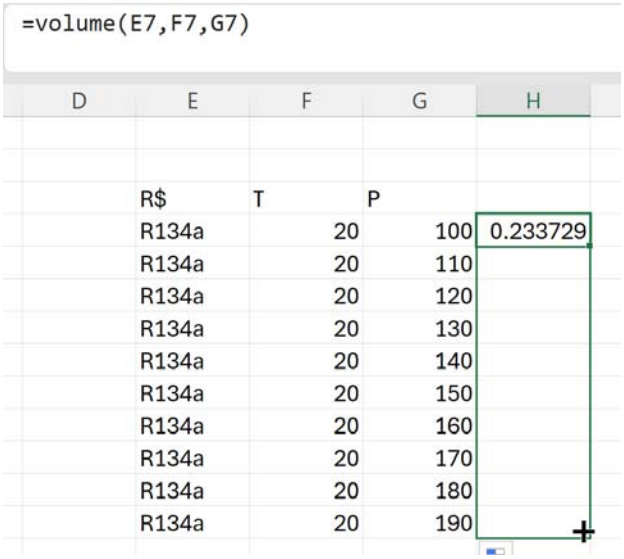
An Example of Calling EES from Excel (continued)

The GetOutput function opens the file v.dat and returns the value contained in it. The final, complete Visual Basic module is shown below.

```
Public Declare PtrSafe Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hwnd As LongPtr,
Public Declare PtrSafe Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal lpClassName As Stri
Public Const hwnd As LongPtr = 0
Public Const wm_RunEES As LongPtr = 32777
Public Const em_LogOff As LongPtr = 1
Public Const em_LogOn As LongPtr = 2
Public Const em_Solve As LongPtr = 10
Public Const em_RunMacroTab As LongPtr = 20
Public Const notUsed As LongPtr = 0
Public Result As LongPtr

Public Const Folder As String = "C:\ExcelExample1"

Sub DeleteFile(FileName)
  On Error Resume Next
  Kill FileName
  On Error GoTo 0
End Sub

Function OutputFileExists(filePath As String) As Boolean
    OutputFileExists = (Len(Dir(filePath)) > 0)
End Function

Sub SetInputs(R As String, T As Double, P As Double) 'user needs to export inputs to EES as required
    Dim RR As String
    RR = "'" + R + "'"
    Open Folder + "\State.dat" For Output As #1
      Print #1, RR, T, P
    Close #1
End Sub

Function GetOutput() As Double
    Dim OutputFile As String
    Dim Volume As Double

    OutputFile = Folder + "\v.dat"
    If (Not OutputFileExists(OutputFile)) Then
        MsgBox "output file " + OutputFile + " was not found"
        GetOutput = -999
    Else
        Open OutputFile For Input As #3
        Input #3, Volume
        Close #3
    End If
    GetOutput = Volume
End Function

Function Volume(R As String, T As Double, P As Double) As Double
    Call SetInputs(R, T, P)  'write inputs to a file for EES

    hWndApp = FindWindow("TEES_D", vbNullString)
    If (hWndApp = 0) Then
        MsgBox "EES is not running"
        Volume = -999
        Exit Function
    End If
    Call SendMessage(hWndApp, wm_RunEES, em_DeleteLog, notUsed)
    Call SendMessage(hWndApp, wm_RunEES, em_LogOn, notUsed)

    Result = SendMessage(hWndApp, wm_RunEES, em_Solve, notUsed)
    If (Result <> 0) Then
        MsgBox "EES returned Error #" + Str(Result) + ""
    Else
        Volume = GetOutput()
    End If

End Function
```

An Example of Calling EES from Excel (continued)

The Volume function can now be used from within Excel in the same way as any other Excel function. For example, you can drag the formula across a range of rows in Excel to compute the specific volume of R134a over a range of pressures at a given temperature, as shown below.



R$	T	P	
R134a	20	100	0.233729
R134a	20	110	0.212003
R134a	20	120	0.193896
R134a	20	130	0.178573
R134a	20	140	0.165437
R134a	20	150	0.154051
R134a	20	160	0.144087
R134a	20	170	0.135294
R134a	20	180	0.127477
R134a	20	190	0.120481

A more complicated example of calling EES from Excel that includes having EES build a thermodynamic property plot that is then used to update an image in the Excel spreadsheet is presented in this online help page.

Calling EES from other Applications

The same approach can be used to call EES from any application that supports the FindWindow and SendMessage commands. A simple example is presented in this page that allows Python to access the property routines that reside in EES. The EES program used for this purpose imports the name of the refrigerant and number of data points from the file Input.dat. The temperature and pressure at each point are imported from the files T.dat and P.dat, respectively. The array of specific volumes are then exported to the file v.dat. From within Python the files are written and the Solve command is sent to the open EES file which executes and writes the v.dat file. The results are then read into Python and displayed.

The ability to integrate EES with Python has many potential applications as it allows functionality unique to EES (e.g., equation solving, thermodynamic property determination, property plots, etc.) to be accessed from within Python. The approach can be extended to other applications, e.g., LabView or MATLAB that support the FindWindow and SendMessage commands.

The $IGMixture Directive

The $IGMixture directive allows up to 6 custom ideal gas mixtures to be defined that you can use in your EES program in the same way as any other built-in fluid. The form of the $IGMixture directive is:

$IGMixture N Component1 MassFraction1 Component2 MassFraction2 …

The value of the parameter N represents the number of the mixture (1 = 'IGMixture1', 2 = 'IGMixture2', etc.). The remaining parameter pairs specify the components (which must be ideal gas fluids in the EES data base) and either the mass or molar composition (depending on the Unit System setting). For example, the directive below specifies the fluid 'IGMixture1' to be a three component ideal gas mixture consisting of 50% N2, 23% Ar, and 27% Xenon (by mass).

```
$UnitSystem SI Mass K Pa J
$IGMixture 1 N2 0.5 Ar 0.23 Xe 0.27
```

The resulting fluid ('IGMixture1') can be used just like any other fluid in a property routine:

```
$VarInfo k Units='W/m-K'
k = Conductivity('IGMixture1',T=300 [K])
```

which leads to k = 0.02156 W/m-K. It can also be used by procedures in the Heat Transfer Library:

```
$VarInfo DELTAP units=Pa
$VarInfo h_H units=W/m^2-K
$VarInfo h_T units=W/m^2-K
T=363 [K]                    "average temperature of fluid in pipe"
P=500000 [Pa]                "pressure in pipe"
m_dot=0.05 [kg/s]            "flow rate"
D=0.025 [m]                  "pipe diameter"
L=50 [m]                     "pipe length"
RelRough=0                   "relative roughness"
CALL PipeFlow('IGMixture1', T, P, m_dot, D, L,RelRough: h_T, h_H, DELTAP, Nusselt_T,  f, Re)
```

which leads to h_T = h_H = 182 W/m^2-K and DELTAP = 28532 Pa. Finally, the fluid can even be used within the models found in the Component Library.

```
$Load Component Library
$VarInfo h_in units=J/kg
$VarInfo h_out units=J/kg
$VarInfo W_dot units=W  altunits=kW

F$='IGMixture1'
P_in=1e5 [Pa]
h_in=Enthalpy(F$,T=300 [K])
P_out=1e6 [Pa]
m_dot=0.1 [kg/s]
eta= 0.9 [-]
CALL Compressor2_CL(h_in, P_in, P_out, m_dot, F$, eta: h_out, W_dot, eta_s)
```

which leads to W_dot = 23.95 kW.

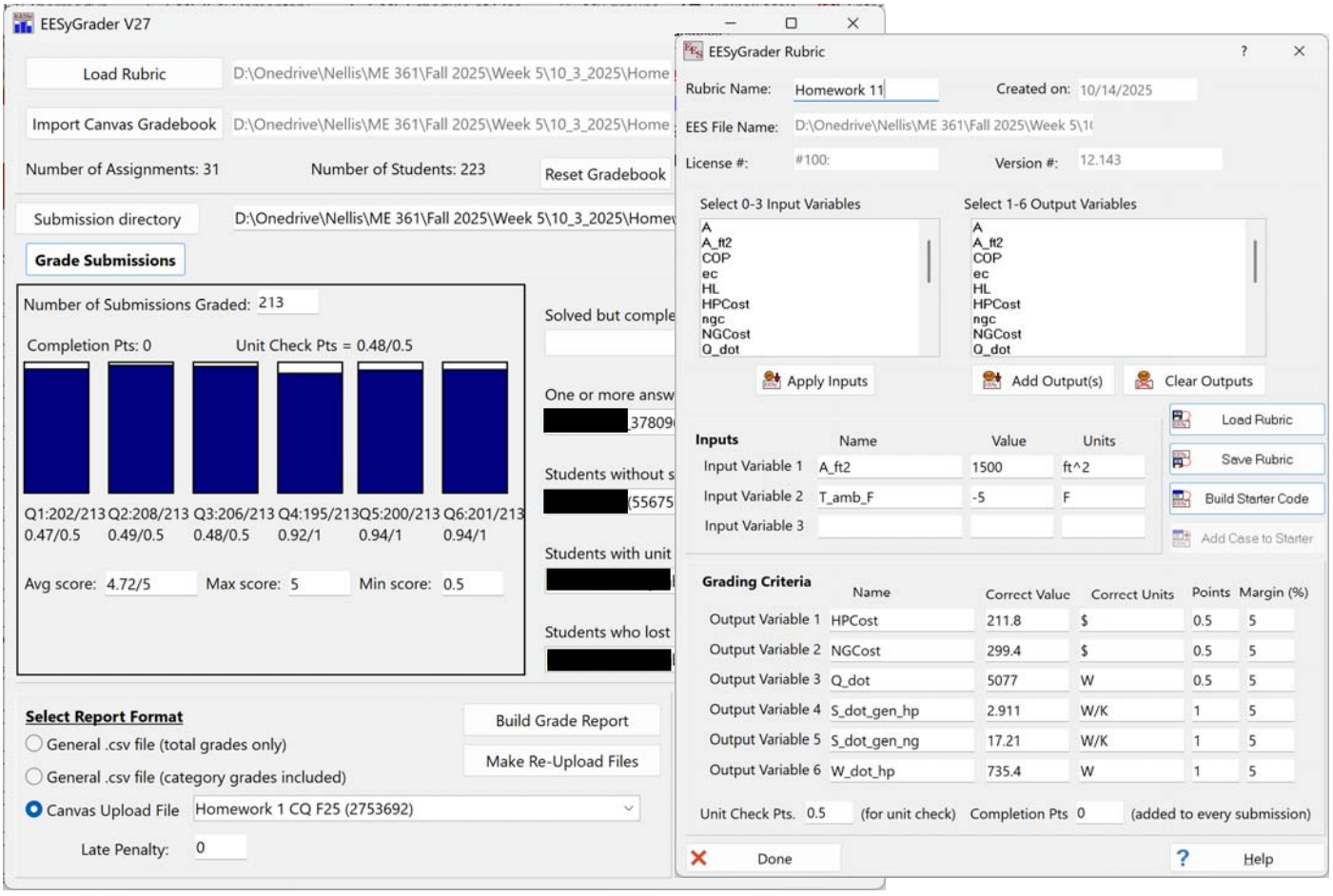
The EES Community Forum

The EES YouTube channel now includes an Community Forum where subscribers can post questions or comments to other users or the developers. Consider using this as a place to post suggestions, comments, and best practices to the EES community. Specific questions should be sent by email to info@fchart.com.

EESyGrader

The EESyGrader grading tool continues to be developed and improved. The current release is version 27 and can be downloaded here. EESyGrader allows you to quickly grade hundreds of submissions carried out in EES, saving hours of time for you or your instructional staff. The time saving can be that used instead to provide help to the students as they attempt to complete the homework or quizzes.

An overview of the EESyGrader tool can be accessed here. The tool allows you to create a rubric and starter code from within EES after you have completed your solution to the problem. The rubric is then used by EESyGrader to grade submitted assignments. EESyGrader creates (1) a grade report (that is compatible with learning management systems like Canvas), (2) a set of re-upload files that can be used to provide feedback to your students, and (3) a set of text files containing only the contents of the Equations Window which can be scanned by any comparator software to detect signs of plagiarism. Taken together the system allows you to quickly grade assignments in large enrollment classes while still allowing students to develop their solutions independently.



Recent Changes to EES

- An identifier can now be included with Diagram Text Inputs and Outputs. The user can now choose between the EES variable name, an identifier selected by the user, or no text at all.
- Selecting or double-clicking on any function name in the Equations Window will bring up a popup menu that includes an item to go to the help page for that function.
- Multiline If-Then-Else logic statements can now be used in Macros.
- The Professional license can be called from other applications using the SendMessage Windows command.
- The Plot Thumbnails has been revised substantially. The thumbnail size control causes the window size to change when the size of the thumbnails changes and the window can be moved out of the EES application frame in order to place the enlarged thumbnail view next to an EES plot for comparison.
- Viscosity and thermal conductivity are provided for both liquid and vapor phase ammonia-water solutions with the NH3H2O fluid. Functions to determine the constant pressure specific heat (SpecHeat) and the volumetric expansion coefficient (VolExpCoef) are also provided.
- Pressing the delete key in a table will clear the contents of the selected rows. If the Ctrl key is also pressed, the selected rows will be deleted.
- **Thermodynamic and transport property data for R465A and R480A are now available.**
- A function to determine the free convection heat transfer coefficient for a vertically arranged bank of horizontal tubes has been added to the Heat Transfer library.
- The equation of state for all R4xx and R5xx refrigerants have been refitted to improve the accuracy of the data in the neighborhood of the critical point and at high densities.
- The contents of the Diagram window in a Distributable program can be resized by clicking the right mouse button to bring up the Scale Diagram Window Dialog.
- The Sort, LogV, Font and TabStops macro commands have been added.
- **The $DefaultVarInfo directive allows the default settings for the guess value, limits, units, and display format to be redefined for new or existing variables.**
- Global Warming Potential (GWP) values are provided for many fluids in the Substance Information Dialog.
- The $IGMixture directive allows the user to specify up to 6 different ideal gas mixtures.
  The $Natural_Gas2 directive allows the composition of a natural gas mixture to be specified.

F-Chart Software

PO Box 44042
Madison, WI, 53744

Internet: http://fchartsoftware.com
E-mail: info@fchart.com

Instant Update & Technical Service

EES uses a different model for updating than most other programs. Each time that there is a change in the EES program, either to correct a problem or to add a new feature, the version number is incremented by 0.001 and the latest version of EES is placed on our website. EES has become very robust and stable. There have been many new versions of EES released with new capabilities since the last EESy Solutions was distributed

A user who has a current subscription to our Instant Update & Technical Service (IUTS) or Academic Update Service (AUS) can download the current version. All new Commercial and Professional licenses of EES are provided with a one year subscription of this service. The cost to continue IUTS or AUS after the first year is ~20% of the current cost of the program per year, provided that it is renewed within 12 months after expiration. Contact us if you wish to re-subscribe to IUTS or AUS.