

Calculations walkthrough of the Predictive Crystallizer Excel unitop

The predictive crystallizer unitop is able to calculate the solubility of a solid in a liquid phase based upon the heat of fusion, melting temperature, and liquid activity coefficient. The unit then “reacts” the liquid form of a component into the solid form, allowing us to calculate fractional crystallizations with a minimum of information.

The predictive crystallizer is an excel unit operation. As such all calculations are done in an excel workbook. In this particular case, all the calculations are handled by visual basic code. There are two main macros (visual basic subroutines) which handle the calculations:

- CalcSolubility: performs the calculations
- Reporting: outputs values to the worksheets

CalcSolubility:

There are six main steps in the calculation procedure:

- 1) add up all liquid and solid fraction of a component
- 2) calculate solubility of the component
- 3) Flash the outlet to determine new enthalpy

Step 2 is a loop, because as the liquid mole fractions change, so does the liquid activity coefficient, and thus solubility(which is dependent on activity coefficient).

Step 1) summing component flows

This step is performed using a function ComponentMoles. Component moles cycles through all of the inlets, summing up both solid and liquid portions, to return the total # of moles for that component coming in.

Next we use the LowestPressure function to return the lowest pressure entering the crystallizer, and we use TotalEnthalpy to sum up the total enthalpy entering the unit.

We reflash the combination of inlets at the specified temperature, and lowest pressure to get ready for the loop.

Step 2) Calculating Solubility

The Solubility calculations are done in an iterative loop, because as the component solubilities change so do the activity coefficients, which impact the solubility.

There is a second loop level where we loop through each possible solid component, calculating the solubility for each component one at a time. The activity coefficients are determined by using the function GetActivityCoeff. The actual solubility of a component is calculated in the function xsol. A mass balance is done to split any solid fractions to their respective solid components.

At this point the current solubility fractions are compared to the last iterations solubility, if the total sum of changes is less than .000001 mole fraction we consider the loop converged, if more than 200 iterations have run the loop did not converge .

Step 3) Reflashing, and sending outlet stream information back into CHEMCAD. The final steps are to transfer stream composition and unitop information back into the CHEMCAD simulation, this is simply a matter of calling the appropriate stream and unitop objects and putting the information into them

Reporting:

The reporting subroutine copies process information into the Excel spreadsheet as a summary sheet for the unitop. No real calculations are done in this subroutine, it is for reporting purposes only.

Sample Code:

What follows is a printout of all the code used to create this unitop. The actual unitop may be viewed by downloading the predictive crystallizer example from our website www.chemstations.net

CalcSolubility:

```
Sub CalcSolubility(ByVal ChemCADEntry As Object)

' this routine is to create an overall framework for the calculations
' 1) add up all liquid and solid fraction= TOTFRACTION (mole fraction)
' 2) calculate xsol
' 3) if xsol is greater than TOTFRACTION then it's all liquid
' 4) if xsol is less than TOTFRACTION then the liquid fraction is Xsol,
the balance is solid
' 5) TOTAL UP OTHER COMPONENT FLOWS
' 6) Using LowPressure and Enthalpy totals, do a PE flash

    ' get chemCAD objects
    Dim CurUnitOp As Object
    Dim strInfo As Object
    Dim strConv As Object
    Dim uopInfo As Object
    Dim uopConv As Object
    Dim Flash As Object

    Set CurUnitOp = ChemCADEntry.GetCurUnitOp
    Set strInfo = ChemCADEntry.GetStreamInfo
    Set strConv = ChemCADEntry.GetStreamUnitConversion
    Set uopInfo = ChemCADEntry.GetUnitOpInfo
    Set uopConv = ChemCADEntry.GetUnitOpSpecUnitConversion
    Set Flash = ChemCADEntry.GetFlash

    'General declarations
    Dim check As Integer
    Dim CompIndex As Integer
    Dim compCount As Integer
    Dim CmpflowratesLbmol_Hr(SIZE_COMP_ARRAY) As Single
    Dim n As Integer

'1) Combine flowrates
'Get the total flows of each component into CmpFlowratesLbmol_HR, sum
into totalmole

    Dim liquidmole As Single
    Dim solidmole As Single
    Dim totalmole As Single
```

```

compCount = strInfo.GetNoOfComponents

totalmole = 0 'Zero the total

For CompIndex = 1 To compCount Step 1

    CmpflowratesLbmol_Hr(CompIndex) = ComponentMoles(ChemCADEntry,
CompIndex)

    totalmole = totalmole + CmpflowratesLbmol_Hr(CompIndex)

Next CompIndex

'2)flash it at operating T and lowest P

Dim setTemperature
Dim inletIDs(SIZE_STREAM_ARRAY) As Integer
Dim tempR As Single
Dim PresPsia As Single
Dim vapFrac As Single
Dim enthBtu_Hr As Single
Dim compLbmol_Hr(SIZE_COMP_ARRAY) As Single

'Get Lowest Pressure and total Enthalpy
Dim LowestP As Single
Dim TEnthalpy As Single
Dim myID As Integer
Dim myPar(1 To SIZE_UNITOP_ARRAY) As Single
Dim StreamTotalEnthalpy As Single

myID = CurUnitOp.GetCurUnitOpID

check = uopInfo.GetUnitOpSpecByID(myID, myPar)

LowestP = LowPressure(ChemCADEntry, myID)
TEnthalpy = TotalEnthalpy(ChemCADEntry, myID)
StreamTotalEnthalpy = TEnthalpy

setTemperature = myPar(2)
'Flash it
check = Flash.DefineFeedStream(setTemperature, LowestP, TEnthalpy,
CmpflowratesLbmol_Hr)
check = Flash.CalculateTPFlash(setTemperature, LowestP)

'get the outlet liquid Stream data
Dim LiquidCmpFlowratesLbmol_hr(SIZE_COMP_ARRAY) As Single

check = Flash.GetLiquidStream(tempR, PresPsia, enthBtu_Hr,
totalmole, LiquidCmpFlowratesLbmol_hr)

'Get the liquid and solid components together
' get parameters and calculate fractions
Dim liquidComponent(5), SolidComponent(5) As Integer
Dim TOTFRACTION(5) As Single

```

```

' unitop parameters 2 and 3 define the liquid and solid components

Dim TmeltK(5) As Single
Dim Hfusion(5) As Single

For n = 1 To 5

    liquidComponent(n) = myPar(5 * n + 1)
    SolidComponent(n) = myPar(5 * n + 2)
    TmeltK(n) = (5# / 9#) * myPar(5 * n + 4)
    Hfusion(n) = myPar(5 * n + 3)

Next n

'calculate Xsol

'*****
Dim converged As Integer
Dim Error As Single
Dim Solid(5) As Single
Dim Iterations As Integer
Dim lastSolid(5) As Single
Dim ActCoeff(5) As Single
Dim XSolubility(5) As Single
Dim LiquidActivityFlows(1 To SIZE_COMP_ARRAY) As Single 'used to pass
flows to activity coeff fcn
Dim TotalLiquidmoles As Single

Dim TemperK As Single

'convert Temp data
TemperK = (5# / 9#) * myPar(2)

'initialize last iterations values
Iterations = 0

For n = 1 To 5
    lastSolid(n) = LiquidCmpFlowratesLbmol_hr(SolidComponent(n))
    ActCoeff(n) = 1
Next n

Do While converged = 0

    'reset error value for this iteration
    Error = 0

    check = Flash.DefineFeedStream(setTemperature, LowestP, TEnthalpy,
LiquidCmpFlowratesLbmol_hr)
    check = Flash.CalculateTPFlash(setTemperature, LowestP)

    'get the outlet liquid Stream data

```

```

    check = Flash.GetLiquidStream(tempR, PresPsia, enthBtu_Hr,
totalmole, LiquidCmpFlowratesLbmol_hr)

    For n = 1 To 5
    CompIndex = liquidComponent(n)

    If CompIndex = 0 Then
        XSolubility(n) = 0
        GoTo skip:
    End If

    'Special case for pure component
    If (LiquidCmpFlowratesLbmol_hr(liquidComponent(n)) +
LiquidCmpFlowratesLbmol_hr(SolidComponent(n))) / totalmole = 1 Then

        If TemperK < TmeltK(n) Then
            XSolubility(n) = 0#
        Else
            XSolubility(n) = 1#
        End If

    Else

        ActCoeff(n) = GetActivityCoeff(ChemCADEntry, CompIndex,
myPar(2), PresPsia, enthBtu_Hr, LiquidCmpFlowratesLbmol_hr)
        ThisWorkbook.ActiveSheet.Cells(3 + n, 6).value =
ActCoeff(n)

        XSolubility(n) = Xsol(TemperK, TmeltK(n), Hfusion(n),
ActCoeff(n))

        If XSolubility(n) < 0 Then
            XSolubility(n) = 0
        End If

    End If

'put Xsol in unit par #6,15,20,25,30
myPar(5 * n + 5) = XSolubility(n)

    TotalLiquidmoles = LiquidSum(ChemCADEntry, myPar(2), PresPsia,
LiquidCmpFlowratesLbmol_hr)

    If TotalLiquidmoles > 0 Then
        TOTFRACTION(n) = (LiquidCmpFlowratesLbmol_hr(SolidComponent(n))
+ LiquidCmpFlowratesLbmol_hr(liquidComponent(n))) / TotalLiquidmoles
    Else
        TOTFRACTION(n) = 0#
    End If

    'check if Xsolubility> TOTFRACTION then solid forms, otherwise it's
all liquid

```

```

        If XSolubility(n) > TOTFRACTION(n) Then

            LiquidCmpFlowratesLbmol_hr(liquidComponent(n)) =
            CmpflowratesLbmol_Hr(SolidComponent(n)) +
            CmpflowratesLbmol_Hr(liquidComponent(n))

            LiquidCmpFlowratesLbmol_hr(SolidComponent(n)) = 0 'No
solid

            Else 'solid forms

                LiquidCmpFlowratesLbmol_hr(liquidComponent(n)) =
XSolubility(n) * TotalLiquidmoles
                LiquidCmpFlowratesLbmol_hr(SolidComponent(n)) =
(CmpflowratesLbmol_Hr(SolidComponent(n)) +
CmpflowratesLbmol_Hr(liquidComponent(n))) - (XSolubility(n) *
TotalLiquidmoles)

                End If

        'check convergence

        Solid(n) = LiquidCmpFlowratesLbmol_hr(SolidComponent(n))
        If Solid(n) > 0 Then
            Error = Error + (Abs(lastSolid(n) - Solid(n)) / (Solid(n)))
        ElseIf lastSolid(n) > 0 Then
            Error = Error + 1
        End If

        lastSolid(n) = Solid(n)
skip:

        Next n

        If Error < 0.000001 Then
            converged = 1
        End If

        If Iterations > 200 Then
            converged = -1
        End If

        Iterations = Iterations + 1

    Loop

'Reflash it, to calculate heat duty

```

```

    check = Flash.DefineFeedStream(myPar(2), LowestP, TEnthalpy,
LiquidCmpFlowratesLbmol_hr)
    check = Flash.CalculateTPFlash(myPar(2), LowestP)

'Get Vapor and Liquid outs
    Dim Tout As Single
    Dim Pout As Single
    Dim vaporHout As Single
    Dim LiquidHout As Single
    Dim vaporTmoles As Single
    Dim liquidTmoles As Single
    Dim VaporOutMoles(SIZE_COMP_ARRAY) As Single
    Dim LiquidOutMoles(SIZE_COMP_ARRAY) As Single

    check = Flash.GetVaporStream(Tout, Pout, vaporHout, vaporTmoles,
VaporOutMoles)

    check = Flash.GetLiquidStream(Tout, Pout, LiquidHout, liquidTmoles,
LiquidOutMoles)

    Dim OutletEnthalpy As Single

    OutletEnthalpy = vaporHout + LiquidHout

'write out heat duty
    myPar(3) = OutletEnthalpy - StreamTotalEnthalpy

'Get outlet IDs
    Dim outletIDs(1 To SIZE_STREAM_ARRAY) As Integer
    check = CurUnitOp.GetOutletIDs(outletIDs)
'If there is one outlet, it all goes out it, if 2 outlets, vapor out
the first, s+1 out second.

    Dim componentcount As Integer
    Dim TotalCompOut(SIZE_COMP_ARRAY) As Single
    Dim BlankCompArray(SIZE_COMP_ARRAY) As Single

    If CurUnitOp.GetNoOfOutlets = 1 Then 'only one outlet

        For componentcount = 1 To SIZE_COMP_ARRAY
            TotalCompOut(componentcount) =
VaporOutMoles(componentcount) + LiquidOutMoles(componentcount)
        Next componentcount

    'Calculate vapor fraction safely
    Dim vaporfrac As Single
    vaporfrac = VaporFraction(vaporTmoles, liquidTmoles)

```

```

        check = strInfo.PutStreamByID(outletIDs(1), Tout, Pout,
vaporfrac, OutletEnthalpy, TotalCompOut)
    End If

    'If there are two outlets vapor goes out the first
    'Liquid goes out the bottom
    If CurUnitOp.GetNoOfOutlets > 1 Then

        check = strInfo.PutStreamByID(outletIDs(1), Tout, Pout, (1#),
vaporHout, VaporOutMoles)

        check = strInfo.PutStreamByID(outletIDs(2), Tout, Pout, (0#),
LiquidHout, LiquidOutMoles)

        For n = 3 To CurUnitOp.GetNoOfOutlets
            check = strInfo.PutStreamByID(outletIDs(n), Tout, Pout,
(0#), (0#), BlankCompArray)
        Next n

    End If

    'write out unitop parameters
    check = uopInfo.PutUnitOpSpecByID(myID, myPar)
End Sub

```

Component Moles function

```
Function ComponentMoles(ByVal ChemCADEntry As Object, CompIndex As Integer) As Double
'returns the sum of moles entering the unitop for component compindex
Dim ninlets, i As Integer
Dim inletIDs(1 To SIZE_STREAM_ARRAY) As Integer
Dim tempR As Single
Dim vapFrac As Single
Dim enthBtu_Hr As Single
Dim PresPsia As Single

Dim compLbmol_Hr(1 To SIZE_COMP_ARRAY) As Single
Dim check As Integer
Dim id As Integer

    ' get chemCAD objects
    Dim CurUnitOp As Object
    Dim strInfo As Object
    Dim strConv As Object
    Dim uopInfo As Object
    Dim uopConv As Object

    Set CurUnitOp = ChemCADEntry.GetCurUnitOp
    Set strInfo = ChemCADEntry.GetStreamInfo
    Set strConv = ChemCADEntry.GetStreamUnitConversion
    Set uopInfo = ChemCADEntry.GetUnitOpInfo
    Set uopConv = ChemCADEntry.GetUnitOpSpecUnitConversion

    'zero it
    ComponentMoles = 0

ninlets = CurUnitOp.GetNoOfInlets

check = CurUnitOp.GetInletIDs(inletIDs) ' sets the array of stream ids

For i = 1 To ninlets
    id = inletIDs(i)

    check = strInfo.GetStreamByID(id, tempR, PresPsia, vapFrac,
enthBtu_Hr, compLbmol_Hr)

    ComponentMoles = ComponentMoles + compLbmol_Hr(CompIndex)
Next i

End Function
```

GetActivityCoeff function

```
Function GetActivityCoeff(ByVal ChemCADEntry As Object, ComponentID As Integer, tempR As Single, PresPsia As Single, enthBtu_Hr As Single, LiquidCmpFlowratesLbmol_hr) As Single
```

```
' Returns a single activity coefficient
```

```
Dim actFlash As Object
```

```
Dim actkvalues As Object
```

```
Dim check As Integer
```

```
Dim actCurUnitOp As Object
```

```
Dim actuopInfo As Object
```

```
Dim n, i As Integer
```

```
Dim tempflowrates(1 To SIZE_COMP_ARRAY) As Single
```

```
Set actFlash = ChemCADEntry.GetFlash
```

```
Set actkvalues = ChemCADEntry.GetKValues
```

```
Set actCurUnitOp = ChemCADEntry.GetCurUnitOp
```

```
Set actuopInfo = ChemCADEntry.GetUnitOpInfo
```

```
Dim myID As Integer
```

```
Dim myPar(1 To SIZE_UNITOP_ARRAY) As Single
```

```
myID = actCurUnitOp.GetCurUnitOpID
```

```
check = actuopInfo.GetUnitOpSpecByID(myID, myPar)
```

```
For n = 1 To SIZE_COMP_ARRAY
```

```
    tempflowrates(n) = LiquidCmpFlowratesLbmol_hr(n)
```

```
    For i = 1 To 5
```

```
        If n = myPar(5 * i + 2) Then
```

```
            tempflowrates(n) = 0#
```

```
        End If
```

```
    Next i
```

```
Next n
```

```
check = actFlash.DefineFeedStream(tempR, PresPsia, enthBtu_Hr, tempflowrates)
```

```
If actFlash.CalculateTPFlash(tempR, PresPsia) = 1 Then
```

```
    ' try vtflash
```

```
    If actFlash.CalculateVTFlash(0#, tempR) = 1 Then
```

```
        ' try vpflash
```

```
        If actFlash.CalculateVPFlash(0#, PresPsia) = 1 Then
```

```
            'Flash didn 't converge
```

```
        End If
```

```
    End If
```

```
End If
```

```

Dim LiquidcompLbmol_Hr(1 To SIZE_COMP_ARRAY) As Single
Dim VaporcompLbmol_Hr(1 To SIZE_COMP_ARRAY) As Single
Dim TotalLFlow As Single
Dim TotalVFlow As Single
Dim LiquidHBtu_Hr As Single
Dim VaporHBtu_Hr As Single

'now separate the vapor and liquid
If Not (actFlash.GetLiquidStream(tempR, PresPsia, LiquidHBtu_Hr,
TotalLFlow, LiquidcompLbmol_Hr)) > 0 Then
    For n = 1 To SIZE_COMP_ARRAY
        LiquidcompLbmol_Hr(n) = tempflowrates(n)
    Next n

End If

check = actFlash.GetVaporStream(tempR, PresPsia, VaporHBtu_Hr,
TotalVFlow, VaporcompLbmol_Hr)

'send the vapor and liquid to the kvalue object
check = actkvalues.DefineLiquidStream(tempR, PresPsia,
LiquidcompLbmol_Hr)
check = actkvalues.DefineVaporStreamComponentRates(VaporcompLbmol_Hr)

'get the activity coefficients
Dim ActivityC(SIZE_COMP_ARRAY) As Single

check = actkvalues.GetActivityCoefficients(ActivityC)

If ActivityC(ComponentID) > 0 Then
    GetActivityCoeff = ActivityC(ComponentID)
Else
    GetActivityCoeff = 1
End If

End Function

```

LiquidSum function

```
Function LiquidSum(ChemCADEntry As Object, tempR As Single, PresPsia As
Single, componentflows) As Single
Dim Flash As Object
Dim Kvalues As Object
Dim check As Integer
Dim CurUnitOp As Object
Dim uopInfo As Object

Dim n, i As Integer

Set Flash = ChemCADEntry.GetFlash
Set Kvalues = ChemCADEntry.GetKValues
Set CurUnitOp = ChemCADEntry.GetCurUnitOp
Set uopInfo = ChemCADEntry.GetUnitOpInfo

Dim myID As Integer
Dim myPar(1 To SIZE_UNITOP_ARRAY) As Single
Dim tempcomponentflow(1 To SIZE_COMP_ARRAY) As Single

myID = CurUnitOp.GetCurUnitOpID

check = uopInfo.GetUnitOpSpecByID(myID, myPar)

For n = 1 To SIZE_COMP_ARRAY
tempcomponentflow(n) = componentflows(n)
For i = 1 To 5
If n = myPar(5 * i + 2) Then
tempcomponentflow(n) = 0
End If

Next i
Next n
check = Flash.DefineFeedStream(tempR, PresPsia, 0, tempcomponentflow)
check = Flash.CalculateTPFlash(tempR, PresPsia)

Dim LiquidcompLbmol_Hr(SIZE_COMP_ARRAY) As Single
Dim VaporcompLbmol_Hr(SIZE_COMP_ARRAY) As Single
Dim TotalLFlow As Single
Dim TotalVFlow As Single
Dim LiquidHBtu_Hr As Single
Dim VaporHBtu_Hr As Single

'now separate the vapor and liquid
check = Flash.GetLiquidStream(tempR, PresPsia, LiquidHBtu_Hr,
TotalLFlow, tempcomponentflow)
LiquidSum = TotalLFlow

End Function
```

LowPressure function

```
Function LowPressure(ChemCADEntry As Object, UnitID As Integer) As
Single

'Returns the lowest NONZERO pressure in psia from feed streams, meant
to find the low pressure
Dim CurUnitOp As Object
Dim strInfo As Object
Dim tempR As Single
Dim vapFrac As Single
Dim enthBtu_Hr As Single
Dim compLbmol_Hr(1 To SIZE_COMP_ARRAY) As Single
Dim check As Integer

Dim ninlets As Integer
Dim inletIDs(1 To SIZE_STREAM_ARRAY) As Integer

Set CurUnitOp = ChemCADEntry.GetCurUnitOp
Set strInfo = ChemCADEntry.GetStreamInfo

ninlets = CurUnitOp.GetNoOfInlets
check = CurUnitOp.GetInletIDs(inletIDs)

Dim Pressurearray(SIZE_STREAM_ARRAY) As Single
Dim i As Integer

    For i = 1 To ninlets
        check = strInfo.GetStreamByID(inletIDs(i), tempR,
Pressurearray(i), vapFrac, enthBtu_Hr, compLbmol_Hr)
    Next i

i = 1 'reset counter

'Now find the lowest non-zero pressure
LowPressure = Pressurearray(1)
For i = 2 To 13
If (Pressurearray(i) < LowPressure And Not (Pressurearray(i) = 0)) Then
LowPressure = Pressurearray(i)
End If
Next i

End Function
```

Reporting Subroutine

```
Sub Reporting(ByVal ChemCADEntry As Object)
'The purpose of this subroutine is to Copy reporting information into
the Excel Spreadsheet
    On Error Resume Next

    ' generate a table of inlet stream data in current user units

    ' get chemCAD objects
    Dim CurUnitOp As Object
    Dim strInfo As Object
    Dim strConv As Object
    Dim uopInfo As Object
    Dim uopConv As Object

    Set CurUnitOp = ChemCADEntry.GetCurUnitOp
    Set strInfo = ChemCADEntry.GetStreamInfo
    Set strConv = ChemCADEntry.GetStreamUnitConversion
    Set uopInfo = ChemCADEntry.GetUnitOpInfo
    Set uopConv = ChemCADEntry.GetUnitOpSpecUnitConversion

    'inlets
    Dim check As Integer
    Dim ninlets As Integer
    Dim noutlets As Integer

    ninlets = CurUnitOp.GetNoOfInlets
    noutlets = CurUnitOp.GetNoOfOutlets

    Dim inletIDs(1 To SIZE_STREAM_ARRAY) As Integer
    Dim outletIDs(1 To SIZE_STREAM_ARRAY) As Integer
    check = CurUnitOp.GetOutletIDs(outletIDs)

    ' setup table header
    Dim tempUnit As String
    Dim presUnit As String
    Dim enthUnit As String
    Dim moleRateUnit As String
    Dim massRateUnit As String
    Dim stdLRateUnit As String
    Dim stdVRateUnit As String
    Dim compUnit As String
    Dim compCategory As String

    check = -1
    'check = strConv.GetCurUserUnitBStr(tempUnit, presUnit, enthUnit,
moleRateUnit, massRateUnit, stdLRateUnit, stdVRateUnit, compUnit,
compCategory)
    strConv.GetCurUserUnitString tempUnit, presUnit, enthUnit,
moleRateUnit, massRateUnit, stdLRateUnit, stdVRateUnit, compUnit,
compCategory

    Dim inletSheet As Worksheet
    Set inletSheet = Worksheets("INSTREAMS")
```

```

    inletSheet.Cells(STREAM_ROW_SECTION_TITLE, STREAM_COL_NAMES).value
= "Inlet Streams"
    inletSheet.Cells(STREAM_ROW_ID, STREAM_COL_NAMES).value = "Stream
IDs"
    inletSheet.Cells(STREAM_ROW_LABEL, STREAM_COL_NAMES).value =
"Labels"

    inletSheet.Cells(STREAM_ROW_TEMPERATURE, STREAM_COL_NAMES).value =
"Temperature"
    inletSheet.Cells(STREAM_ROW_TEMPERATURE, STREAM_COL_UNITS).value =
tempUnit

    inletSheet.Cells(STREAM_ROW_PRESSURE, STREAM_COL_NAMES).value =
"Pressure"
    inletSheet.Cells(STREAM_ROW_PRESSURE, STREAM_COL_UNITS).value =
presUnit

    inletSheet.Cells(STREAM_ROW_ENTHALPY, STREAM_COL_NAMES).value =
"Enthalpy"
    inletSheet.Cells(STREAM_ROW_ENTHALPY, STREAM_COL_UNITS).value =
enthUnit

    inletSheet.Cells(STREAM_ROW_MOLE_FRACTION, STREAM_COL_NAMES).value
= "Vapor Mole Fraction"

    inletSheet.Cells(STREAM_ROW_MOLE_FLOWRATE, STREAM_COL_NAMES).value
= "Total Mole FlowRate"
    inletSheet.Cells(STREAM_ROW_MOLE_FLOWRATE, STREAM_COL_UNITS).value
= moleRateUnit

    inletSheet.Cells(STREAM_ROW_MASS_FLOWRATE, STREAM_COL_NAMES).value
= "Total Mass FlowRate"
    inletSheet.Cells(STREAM_ROW_MASS_FLOWRATE, STREAM_COL_UNITS).value
= massRateUnit

    inletSheet.Cells(STREAM_ROW_STD_LIQ_VOLRATE,
STREAM_COL_NAMES).value = "Total Std. Liq. Vol. FlowRate"
    inletSheet.Cells(STREAM_ROW_STD_LIQ_VOLRATE,
STREAM_COL_UNITS).value = stdLRateUnit

    inletSheet.Cells(STREAM_ROW_STD_VAP_VOLRATE,
STREAM_COL_NAMES).value = "Total Std. Vap. Vol. FlowRate"
    inletSheet.Cells(STREAM_ROW_STD_VAP_VOLRATE,
STREAM_COL_UNITS).value = stdVRateUnit

    Worksheets("INSTREAMS").Activate
    Cells(STREAM_ROW_COMPFLOW_VALUE, STREAM_COL_NAMES).value =
compCategory
    Cells(STREAM_ROW_COMPFLOW_VALUE, STREAM_COL_UNITS).value = compUnit

    'intletSheet.Cells(STREAM_ROW_STD_VAP_VOLRATE, STREAM_COL_VALUE +
1).Value = TotalStdVVolFlowRate()

    Dim CompIndex As Integer
    Dim compCount As Integer
    Dim curRow As Integer
    Dim compName As String

```

```

Dim compID As Integer
curRow = STREAM_ROW_COMPFLOW_VALUE
compCount = strInfo.GetNoOfComponents

For CompIndex = 1 To compCount Step 1
    compName = strInfo.GetComponentNameByPosBaseOne(CompIndex)
    compID = strInfo.GetComponentIDByPosBaseOne(CompIndex)

    inletSheet.Cells(curRow + CompIndex, STREAM_COL_NAMES).value =
compName
    inletSheet.Cells(curRow + CompIndex, STREAM_COL_UNITS).value =
compID
Next CompIndex

' prepare flash data
Dim tempR As Single
Dim PresPsia As Single
Dim vapFrac As Single
Dim enthBtu_Hr As Single
Dim compLbmol_Hr(1 To SIZE_COMP_ARRAY) As Single

Dim Pressure As Single
Dim enthalpy As Single
Dim temperature As Single
Dim mvf As Single
Dim moleRate As Single
Dim massRate As Single
Dim stdLRate As Single
Dim stdVRate As Single
Dim compRate(1 To SIZE_COMP_ARRAY) As Single

Dim streamIndex As Integer
Dim id As Integer
Dim idcopy As Integer
Dim col As Integer
Dim curCol As Integer
curCol = STREAM_COL_VALUE

Dim valKeeper As Integer

For streamIndex = 1 To ninlets Step 1
    valKeeper = streamIndex
    id = inletIDs(streamIndex)
    idcopy = id

    check = strInfo.GetStreamByID(id, tempR, PresPsia, vapFrac,
enthBtu_Hr, compLbmol_Hr)

    Rem These two statements are always used together
    check = strConv.DefineStreamInInternalUnit(tempR, PresPsia,
enthBtu_Hr, compLbmol_Hr)
    check = strConv.GetStreamInCurUserUnit(temperature, Pressure,
enthalpy, moleRate, massRate, stdLRate, stdVRate, compRate)

    col = curCol + valKeeper
    inletSheet.Cells(STREAM_ROW_ID, col).value =
inletIDs(valKeeper)

```

```

        inletSheet.Cells(STREAM_ROW_LABEL, col).value =
strInfo.GetStreamLabelByID(idcopy)

        inletSheet.Cells(STREAM_ROW_TEMPERATURE, col).value =
temperature
        inletSheet.Cells(STREAM_ROW_PRESSURE, col).value = Pressure
        inletSheet.Cells(STREAM_ROW_ENTHALPY, col).value = enthalpy
        inletSheet.Cells(STREAM_ROW_MOLE_FRACTION, col).value = vapFrac

        inletSheet.Cells(STREAM_ROW_MOLE_FLOWRATE, col).value =
moleRate
        inletSheet.Cells(STREAM_ROW_MASS_FLOWRATE, col).value =
massRate
        inletSheet.Cells(STREAM_ROW_STD_LIQ_VOLRATE, col).value =
stdLRate
        inletSheet.Cells(STREAM_ROW_STD_VAP_VOLRATE, col).value =
stdVRate

        For CompIndex = 1 To compCount Step 1
            inletSheet.Cells(curRow + CompIndex, col).value =
compRate(CompIndex)
        Next CompIndex

        streamIndex = valKeeper
    Next streamIndex

    ' generate a table of equipment parameters in current user units

    ' unitop ID
    Dim myID As Integer
    myID = CurUnitOp.GetCurUnitOpID

    ' get parameters
    Dim myPar(1 To SIZE_UNITOP_ARRAY) As Single
    Dim myParInUserUnits(1 To SIZE_UNITOP_ARRAY) As Single

    check = uopInfo.GetUnitOpSpecByID(myID, myPar)
    check = uopConv.FromInternalUnitsToCurUserUnits(myPar,
myParInUserUnits)

    Dim parCount As Integer
    Dim parIndex As Integer
    Dim leadRow As Integer
    leadRow = UNITOP_COL_VALUE
    parCount = uopConv.GetNoOfParameters(myID)

    Dim simDataSheet As Worksheet
    Set simDataSheet = Worksheets("SIMDATA")
    simDataSheet.Activate

    Dim parName As String
    Dim parUnitString As String
    For parIndex = 2 To parCount Step 1
        valKeeper = parIndex

        check = uopConv.GetCurUserUnitString(myID, parIndex,
parUnitString, parName)

```

```

        simDataSheet.Cells(leadRow + parIndex, UNITOP_COL_NAMES).value
= parName
        simDataSheet.Cells(leadRow + parIndex, UNITOP_COL_UNITS).value
= parUnitString
        simDataSheet.Cells(leadRow + parIndex, UNITOP_COL_VALUE).value
= myParInUserUnits(parIndex)

        parUnitString = ""
        parName = ""
    Next parIndex

    'add the header
    simDataSheet.Cells(UNITOP_ROW_SEC_TITLE, UNITOP_COL_NAMES).value =
"Current UnitOp Parameters"
    simDataSheet.Cells(UNITOP_ROW_ID, UNITOP_COL_NAMES).value = "UnitOp
ID"
    simDataSheet.Cells(UNITOP_ROW_ID, UNITOP_COL_VALUE).value = myID

    simDataSheet.Cells(UNITOP_ROW_LABEL, UNITOP_COL_NAMES).value =
"Label"
    simDataSheet.Cells(UNITOP_ROW_LABEL, UNITOP_COL_VALUE).value =
uopInfo.GetUnitOpLabelByID(myID)

'Get Outlet Stream Data
    Dim outletsheet As Worksheet
    Set outletsheet = Worksheets("OUTSTREAMS")

    outletsheet.Cells(STREAM_ROW_SECTION_TITLE, STREAM_COL_NAMES).value
= "Inlet Streams"
    outletsheet.Cells(STREAM_ROW_ID, STREAM_COL_NAMES).value = "Stream
IDs"
    outletsheet.Cells(STREAM_ROW_LABEL, STREAM_COL_NAMES).value =
"Labels"

    outletsheet.Cells(STREAM_ROW_TEMPERATURE, STREAM_COL_NAMES).value =
"Temperature"
    outletsheet.Cells(STREAM_ROW_TEMPERATURE, STREAM_COL_UNITS).value =
tempUnit

    outletsheet.Cells(STREAM_ROW_PRESSURE, STREAM_COL_NAMES).value =
"Pressure"
    outletsheet.Cells(STREAM_ROW_PRESSURE, STREAM_COL_UNITS).value =
presUnit

    outletsheet.Cells(STREAM_ROW_ENTHALPY, STREAM_COL_NAMES).value =
"Enthalpy"
    outletsheet.Cells(STREAM_ROW_ENTHALPY, STREAM_COL_UNITS).value =
enthUnit

    outletsheet.Cells(STREAM_ROW_MOLE_FRACTION, STREAM_COL_NAMES).value
= "Vapor Mole Fraction"

    outletsheet.Cells(STREAM_ROW_MOLE_FLOWRATE, STREAM_COL_NAMES).value
= "Total Mole FlowRate"

```

```

    outletsheet.Cells(STREAM_ROW_MOLE_FLOWRATE, STREAM_COL_UNITS).value
= moleRateUnit

    outletsheet.Cells(STREAM_ROW_MASS_FLOWRATE, STREAM_COL_NAMES).value
= "Total Mass FlowRate"
    outletsheet.Cells(STREAM_ROW_MASS_FLOWRATE, STREAM_COL_UNITS).value
= massRateUnit

    outletsheet.Cells(STREAM_ROW_STD_LIQ_VOLRATE,
STREAM_COL_NAMES).value = "Total Std. Liq. Vol. FlowRate"
    outletsheet.Cells(STREAM_ROW_STD_LIQ_VOLRATE,
STREAM_COL_UNITS).value = stdLRateUnit

    outletsheet.Cells(STREAM_ROW_STD_VAP_VOLRATE,
STREAM_COL_NAMES).value = "Total Std. Vap. Vol. FlowRate"
    outletsheet.Cells(STREAM_ROW_STD_VAP_VOLRATE,
STREAM_COL_UNITS).value = stdVRateUnit

    Worksheets("OUTSTREAMS").Activate
    Cells(STREAM_ROW_COMPFLOW_VALUE, STREAM_COL_NAMES).value =
compCategory
    Cells(STREAM_ROW_COMPFLOW_VALUE, STREAM_COL_UNITS).value = compUnit

    curRow = STREAM_ROW_COMPFLOW_VALUE
    compCount = strInfo.GetNoOfComponents

    For CompIndex = 1 To compCount Step 1
        compName = strInfo.GetComponentNameByPosBaseOne(CompIndex)
        compID = strInfo.GetComponentIDByPosBaseOne(CompIndex)

        outletsheet.Cells(curRow + CompIndex, STREAM_COL_NAMES).value =
compName
        outletsheet.Cells(curRow + CompIndex, STREAM_COL_UNITS).value =
compID
    Next CompIndex

    ' prepare flash data

    curCol = STREAM_COL_VALUE

    For streamIndex = 1 To noutlets
        valKeeper = streamIndex
        id = outletIDs(streamIndex)
        idcopy = id

        check = strInfo.GetStreamByID(id, tempR, PresPsia, vapFrac,
enthBtu_Hr, compLbmol_Hr)

        Rem These two statements are always used together
        check = strConv.DefineStreamInInternalUnit(tempR, PresPsia,
enthBtu_Hr, compLbmol_Hr)
        check = strConv.GetStreamInCurUserUnit(temperature, Pressure,
enthalpy, moleRate, massRate, stdLRate, stdVRate, compRate)

```

```

        col = curCol + valKeeper
        outletsheet.Cells(STREAM_ROW_ID, col).value =
outletIDs(valKeeper)
        outletsheet.Cells(STREAM_ROW_LABEL, col).value =
strInfo.GetStreamLabelByID(idcopy)

        outletsheet.Cells(STREAM_ROW_TEMPERATURE, col).value =
temperature
        outletsheet.Cells(STREAM_ROW_PRESSURE, col).value = Pressure
        outletsheet.Cells(STREAM_ROW_ENTHALPY, col).value = enthalpy
        outletsheet.Cells(STREAM_ROW_MOLE_FRACTION, col).value =
vapFrac

        outletsheet.Cells(STREAM_ROW_MOLE_FLOWRATE, col).value =
moleRate
        outletsheet.Cells(STREAM_ROW_MASS_FLOWRATE, col).value =
massRate
        outletsheet.Cells(STREAM_ROW_STD_LIQ_VOLRATE, col).value =
stdLRate
        outletsheet.Cells(STREAM_ROW_STD_VAP_VOLRATE, col).value =
stdVRate

        For CompIndex = 1 To compCount Step 1
            outletsheet.Cells(curRow + CompIndex, col).value =
compRate(CompIndex)
        Next CompIndex

        streamIndex = valKeeper
    Next streamIndex

End Sub

```

TotalEnthalpy function

Function TotalEnthalpy(ChemCADEntry As Object, UnitID As Integer) As Single

'Returns the totalenthalpy in internal units

```
Dim CurUnitOp As Object
Dim strInfo As Object
Dim tempR As Single
Dim vapFrac As Single
Dim enthBtu_Hr As Single
Dim Pressure As Single
Dim compLbmol_Hr(1 To SIZE_COMP_ARRAY) As Single
Dim check As Integer
```

```
Dim ninlets As Integer
Dim inletIDs(1 To SIZE_STREAM_ARRAY) As Integer
```

```
Set CurUnitOp = ChemCADEntry.GetCurUnitOp
Set strInfo = ChemCADEntry.GetStreamInfo
```

```
ninlets = CurUnitOp.GetNoOfInlets
check = CurUnitOp.GetInletIDs(inletIDs)
'initialize
TotalEnthalpy = 0
enthBtu_Hr = 0
```

```
Dim i As Integer
```

```
For i = 1 To ninlets
    check = strInfo.GetStreamByID(inletIDs(i), tempR, Pressure,
vapFrac, enthBtu_Hr, compLbmol_Hr)
```

```
    TotalEnthalpy = TotalEnthalpy + enthBtu_Hr
```

```
Next i
```

```
End Function
```

VaporFraction function

```
Function VaporFraction(Vapormoles As Single, LiquidMoles As Single) As  
Single  
'Calculate vapor fraction safely  
  
If LiquidMoles = 0 Then  
    If Vapormoles > 0 Then  
        VaporFraction = 1  
    Else  
        VaporFraction = 0  
    End If  
Else  
    VaporFraction = (Vapormoles / (Vapormoles + LiquidMoles))  
  
End If  
End Function
```

Xsol function

```
Function Xsol(Temp As Single, Tmelt As Single, Hfusion As Single,
ActCoeff As Single) As Single
Rem returns Xm(mol fraction) as a function of
Rem melting Temp(Kelvin)
Rem Heat of fusion (cal/mol)
Rem Temp(kelvin)
Rem Activity Coefficient

Rem Returns -1 if there is an error

If Temp <= 0 Or Tmelt <= 0 Or Hfusion <= 0 Then
Rem input error
GoTo Error
End If

Xsol = (Exp((Hfusion / (1.987 * Temp)) * ((Temp / Tmelt) - 1))) /
ActCoeff
If Xsol > 1 Then Xsol = 1

GoTo endofF
Error:
Xsol = -1
Rem input error

endofF:

End Function
```