

USEFUL APPLICATIONS OF PROCESS CALCULATORS

Roberto Paron, Prode www.prode.com

paper presented at Chemputers Europe,
Conference & Exhibition of Computer Technology for Chemical Engineers
November 15-16 1994 Milano Italia

Introduction

Almost all chemical engineers do use of custom procedures, it might be a compact set of equations in an electronic sheet or a large and complex application. There are several reasons to do so, they might have a specific know-how in the matter, or prefer to see exactly what the program does, or the calculations reduce to two equations, or finally, they haven't the budget for that particular software product. Whatever the reason the development of custom procedures is a common practice and it appears interesting to examine the tools for that purpose. Perhaps the most utilized are the electronic sheets which offer the industry-standard graphical interface for numerical calculations and a macro-language. Those macro-languages are powerful for defining procedures partly because of unique cell-interface approach which permits to enter directly equations in a flow structure fashion. In addition some product supports also Basic constructs and has functions to link external libraries. However, also with these capabilities, the definition of complex procedures involving thermodynamic calculations remains a difficult task. In those cases usually one prefers development systems supporting languages as Fortran, Basic, C, Pascal etc. which consent a large flexibility for accessing libraries, defining complex algorithms etc. For small to medium complexity problems an interesting alternative is constituted by process calculators. These are programmable tools particularly suited for chemical engineering calculations. The term 'process calculators' probably originates from process simulators where frequently a implementation resides as Fortran window. However process calculators are also as separate tools and, in that form, can constitute a small and versatile development systems for creating custom applications. The purpose of this work is to analyze the applications of process calculators in several areas of chemical engineering and show how these tools can result useful. Details of proposed methods are included as sample listings, the main characteristics of the program utilized in this work are resumed in following paragraph.

Characteristics of a process calculator

A process calculator has been developed (using C++ language) and utilized in this work.

The application has three main graphical objects ,

- an editor which permits to write applications
- a calculator from where to solve problems
- a database editor for accessing the chemical's data-bank

Actual versions includes a data bank with about 1000 chemicals. Programming language permits to define both procedures and elements of graphical interface.

Graphical interface can include cells for numeric or alphanumeric values and buttons for activating processes at user's request.

For each cell a default value can be assigned and range limits defined.

Cells can be defined as read-only.

A conversion calculator object is associated to each numeric cell, in procedure a code identifies the type of quantity and calculator makes several different units of measurement for that quantity. Internal representation for all quantities follows SI code.

Programming language follows C style with some extension required for scientific calculations.

Program supports four data types (booleans, integers, floatings and strings).

There are the four fundamental operators plus power operator.

Arrays are defined as runtime resources and direct operations are permitted.

A library of methods for mathematical and thermodynamic calculations is included.

Case 1: flow device sizing

a common task in plant's design is the sizing of flow devices as valves, orifice plates, nozzles etc. This is usually accomplished with the use of several tools as electronic sheets, custom programs, process simulators etc. A process calculator permits to create quickly a series of short procedures for that purpose, here we examine a method for sizing a primary flow device (orifice plate), the inclusion of two additional equations (for calculating new discharge coefficients) permits to extend the application also to nozzles and venturi meters. Although not sophisticated (there are examples of similar calculations solved with pocket calculators) the program should serve the purpose to show the capabilities of a process calculator in equipment's design.

The basic flow equation for a meter with circular orifice is

$$q_m = 1.11072 \cdot \alpha \cdot \varepsilon \cdot d^2 \cdot \sqrt{\Delta p \cdot \rho_1}$$

where :

q_m	= mass rate of flow	(Kg/s)
α	= flow coefficient	
ε	= expansion factor	
d	= orifice diameter	(m)
Δp	= differential pressure	(Pa)
ρ	= mass density of the fluid	(Kg/m ³)

in both AGA Report 3 [1] and ISO 5167 [2] standards, empirical equations are given to calculate the flow coefficient in orifice plates, in ISO 5167 the parameter C (see below) is given by the Stolz equation :

$$\alpha = \frac{C}{\sqrt{1 - \beta^4}}$$

$$Re_D = \frac{4 \cdot q_m}{3600 \cdot \pi \cdot D \cdot \mu}$$

$$C = 0.5959 + 0.0312 \cdot \beta^{2.1} - 0.184 \cdot \beta^8 + 0.0029 \cdot \beta^{2.5} \cdot \left(\frac{10^6}{Re_d} \right)^{0.75} + \\ + \frac{0.009 \cdot K_1 \cdot \beta^4}{1 - \beta^4} - 0.0337 \cdot K_2 \cdot \beta^3$$

where :

C	= coefficient of discharge	
β	= diameter ratio	(d/D)
K_1, K_2	= quotients of distance	(l/d)
Re_D	= Reynolds nr. referred to pipe diameter	
μ	= dynamic viscosity	(Pa.s)
D	= pipe internal diameter	(m)

The expansion factor (ϵ) can be calculated as :

where :

κ	= isentropic exponent	(cp/cv)
p	= inlet pressure	(Pa)

Additional equations are in AGA 3 [1] report and in API standard 2530 [3].

In design one generally knows the range of measurement and needs to calculate the orifice diameter. Since the discharge coefficient (C) has an approximately constant value, it appears useful to treat it as a constant in preliminary calculations. In orifice plates the discharge coefficient has a value of about 0.6, in ISO5167 nozzles about 1.0 etc. , with this assumption it becomes possible to calculate a preliminary value for the orifice diameter with an error usually less than 10% :

$$d = \sqrt{\frac{q_m}{1.11072 \cdot C \cdot \epsilon \cdot \sqrt{\Delta p \cdot \rho}}}$$

Starting from this value a iterative method such as Newton-Raphson can be utilized to refine the result, generally two iterations are sufficient to obtain five significant digits.

Utilizing the Newton-Raphson algorithm it appears convenient to use a simplified derivative considering (again) the coefficient of discharge as a constant, this because the complete derivative is expensive to calculate and doesn't provide concrete advantages in terms of speed of convergence, with that assumption the flow correlation and derivative are :

$$q_m = 1.11072 \cdot \frac{C}{\sqrt{1 - \beta^4}} \cdot \epsilon \cdot d^2 \cdot \sqrt{\Delta p \cdot \rho}$$

$$f_1 = 2.22144 \cdot C \cdot \epsilon \cdot D^6 \cdot d \cdot \frac{\sqrt{\Delta p \cdot \rho}}{(D^4 - d^4)^{1.5}}$$

at each step the new trial for orifice diameter is calculated as:

$$d_n = \frac{q_{m(n-1)} - w}{f_{1(n-1)}}$$

where:

w = required mass rate of flow (Kg/s)

The described procedure might be coded in an electronic sheet but, since the procedure requires data as densities, viscosities etc. the user might be required to enter the values from tables or other sources.

A process calculator can represent a good arrangement, the creation of a short procedure (as that described above) requires a relatively little effort. See below the procedure coded in Pc language, the interface definition requires about 20 lines of code and the complete program about 80 lines.

REFERENCES

- [1] American Gas Association, report nr. 3 (1955)
- [2] International Organization for Standardization, ISO 5167, 1980
- [3] American Petroleum Institute, API standard 2530 (second edition , 1985)

```

/*****
orifice plate sizing, standard ISO5167
*****/
input {
CELL = tin, LABEL= "inlet temperature", TYPE= 120,IN=1,
CELL = pin, LABEL= "inlet pressure", TYPE= 111,VALUE=0,IN=1,
CELL = w, LABEL= "full range flow", TYPE= 130,VALUE= 0,IN=1,
CELL = dp, LABEL= "differential press.", TYPE= 110, VALUE=0,IN=1,
CELL = D, LABEL= "pipe int. diam.", TYPE= 101,VALUE=0.05,IN=1,
CELL = visc, LABEL= "fluid dyn. visc.", TYPE= 150,VALUE=0,IN=0,
CELL = dens, LABEL= "fluid density", TYPE= 140,VALUE=0,IN=0,
CELL = Re, LABEL= "Reynolds nr (pipe)", TYPE= 100,VALUE=0,IN=0,
CELL = d, LABEL= "orifice diameter", TYPE= 101,VALUE=0,IN=0,
STREAM = "inlet",
BUTTON = save, LABEL = "save",
BUTTON= load, LABEL = "load",
BUTTON = flange_taps, LABEL = "flgT",
BUTTON= corner_taps, LABEL = "cornT",
BUTTON = line_taps, LABEL = "lineT",
}
int taps_type;
taps_type = 0; /* default to flange taps */
main() {
int stream = 1; /* 1 means the inlet stream, the only defined */
float qm, beta,C, m1, f1, k1, k2;
dens =StrmLiqDens(stream,tin,pin);/* obtain values for density and viscosity from property
generator */
visc = StrmLiqVisc(stream,tin,pin);
exp_fact = 1; /* include here exp.fact. formula to extend calc. to gas state */
C=0.6; /* calculate a preliminary value for orifice diameter (assume discharge coefficient = 0.6)
*/
d = (w/(1.11072*C*exp_fact*(dp*dens)**0.5)**0.5;
if (taps_type==0) { /* flange taps */
k1=0.0254 / D;
k2=k1;
}
else if (taps_type==1) { /* corner taps */
k1=0;
k2=0;
}
else if (taps_type==2) { /* line taps */
k1=1;
k2=0.47;
}

qm = w;
do
{
beta = d / D;
m1 = beta**4 / (1-beta**4);
if ( k1>0.433) { m1=0.039; }
C = 0.5959 + 0.0312*beta**2.1-0.184*beta**8+76.509*beta**2.5*(D*visc/w)**0.75+0.09*k1*m1-
0.337*k2*beta**3;
qm = 1.11072*C/((1-beta**4)**0.5) * exp_fact * d*d* (dp*dens)**0.5;
f1 = 2.22144*C*exp_fact*D**6*d*(dp*dens)**0.5/(D**4 - d**4)**1.5;
d = (qm-w)/f1;
}
while (abs(qm-w)/w > 1e-4);
}
save() { /* activated by Save button, save data in ASCII text file */
write(tin,pin,w,dp,D,visc,dens,Re,d);
}
load() { /* activated by load button, load data from ASCII text file */
read(tin,pin,w,dp,D,visc,dens,Re,d);
}
flange_taps() { /* activated by flgT button, defines flange taps orifice */
taps_type=0;
}
corner_taps() { /* activated by cornT button, defines corner taps orifice */
taps_type=1;
}
line_taps() { /* activated by lineT button, defines line taps orifice */

```

```
taps_type=2;  
}
```

Case 2 : modeling of data

Generally, with the term modeling of data one refers to the process of defining the parameters of a model (or the model itself) in order to reproduce as well as possible some predetermined condition. This definition suggests the presence of a method to calculate deviations (or a likelihood estimator) and one function to minimize that value by modifying some parameter in the model. A well-known application of data modeling is the curve fitting. In that procedure the likelihood estimator employs usually a derivation of least square method while the minimization routine follows one of several algorithms proposed in literature [1] [2].

Several tools, as electronic sheets or statistical software include minimization methods and are widely utilized in chemical engineering. However, if one has a set of experimental values for fitting with an appropriate model and the model requires calculations as vapor -liquid equilibrium, enthalpies etc. a process calculator can give several benefits.

An interesting possibility, with process calculators, is the use of library methods instead to code complex algorithms. The advantage of that solution is the simplicity which permits to define the procedure and see the results in a very short time.

Suppose we would design a sour water stripper and the problem is to find an adequate model to represent vapor -liquid equilibrium for water/ammonia binary.

A solution is to use a equation of state as the Soave Redlich Kwong or Peng Robinson models by finding an adequate interaction parameter. That value might be obtained by regressing experimental data and, since the column operates about atmospheric pressure, following experimental points are :

Ammonia (1)		Water (2)	
Temp. (K)	Press.(Bar.a)	X1	Y1
333,15	0,853	0,1199	0,792
353,15	1,044	0,0604	0,531
363,15	0,866	0,0147	0,192
373,15	1,078	0,005	0,0614

As discussed above the regression routine requires a likelihood estimator and a minimization algorithm. There are several methods in literature for likelihood estimators especially when applied to experimental data which might be of poor quality, see for example [3]. However, since we want to keep low the complexity of procedure a solution is to use only those data which we consider affordable,

in our example temperature, pressure and liquid composition (being usually vapor composition affected by higher errors in measurement).

In addition we assume that measurement errors are negligible . At this point the problem reduces to calculate for each experimental point the bubble temperature (or pressure) with the given liquid composition (x_n for ammonia and $1-x_n$ for water) and converge K_{ji} for finding the minimum difference from calculated and experimental values.

Likelihood estimator is defined as :

$$err = \frac{tcalc_1 - texp_1}{texp_1} + \frac{tcalc_2 - texp_2}{texp_2} + \dots + \frac{tcalc_n - texp_n}{texp_n}$$

where :

$tcalc_1$ = calculated dew point temperature (or pressure)

$texp_1$ = experimental data

Since usually the function is relatively smooth (in K_{ij}) a simple minimization algorithm as the secant method can be used.

We start assuming as initial values 0 and 0.01 for k_{ij} , and we calculate new trials according :

$$K_{ij_{new}} = k_{ij_1} - err_1 \cdot \frac{k_{ij_1} - k_{ij_0}}{err_1 - err_0}$$

Following the procedure coded in Pc language, the function streamLFT() is utilized, it returns the temperature for a fixed liquid fraction and composition.

```
main () {
.....
.....
kij=1;
kijold=kij;
err = errcalc(kij);
kij=0.01;
while(err>0.0001) {
    errold=err;
    err=errcalc(kij);
    kijnew=kij-err*(kij-kijold)/(err-errold);
    kijold=kij;
    kij=kijnew;
}

float errcalc(float kij)
/* calculates total deviation as
   (calc.value - meas.value) / meas.value
*/
{
int count;
float err;
err=0;
defMP(1,0,1,kij);
for (count=0;count<nrddata;count=count+1)
{
    err = err + streamLFT(count,press[count],1)-texp[count]/
        texp[count];
}
return err;
}
```


the final value calculated for k_{ij} is 0.27 and the convergence rate is satisfactory.

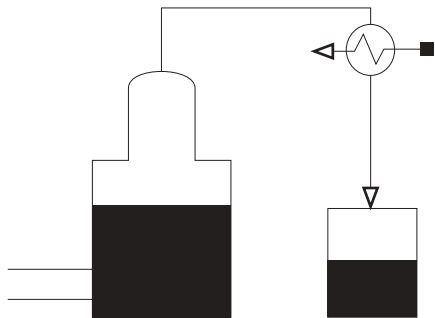
More complex procedures for finding the constants for a model of excess Gibbs energy as UNIQUAC, Wilson etc. are given in [4][5]

REFERENCES

- [1] Numerical methods that work , Acton F.S, Mathematical Association of America, 1990
- [2] Dennis J.E. , Schnabel R.B, Numerical methods for unconstrained optimization and nonlinear equations, Prentice Hall, 1983
- [3] Bard Y, Nonlinear Parameter estimation, Academic Press 1974
- [4] Reid R.C., Prausnitz J.M., Poling B.E., The properties of gases & liquids, McGraw-Hill 1988
- [5] Prausnitz J.M., Anderson T.F., Grens E.A., Eckert C.A., Hsieh R., O'Connell J.P., Computer calculations for multicomponent vapor-liquid and liquid-liquid equilibria, Prentice Hall 1980

Case 3, use direct integration for pseudo-dynamic simulations

Frequently one needs to evaluate transient or unsteady conditions. The two terms mean essentially the same thing, the time dimension is introduced into the description. The simulation of the behavior of large plants in transient conditions is a difficult task having usually to deal with thousands of differential equations and related problems, we examine here a simple technique which can be applied in many situations giving results of accuracy equivalent to more sophisticated methods. Supposing to have a small plant where a mixture is charged to a reboiler, a heat is supplied and mixture vaporizes, the vapors are taken to a condenser and to a liquid receiver.



The vapor leaving the reboiler is in equilibrium with the liquid but, since the vapor is richer of the more volatile components, the compositions of liquid and vapor are not constant. In order to evaluate what happens after a certain period of time one must know how the liquid mixture changes with respect to the time. The use of a process calculator permits to solve that problem by using a direct integration technique. As the term suggest the method consists in to subdivide the total time in n steps and solve the model at each step tacking in account for variations.

A sum of the type

$$\sum_{i=0}^n a_i f(t_i)$$

is utilized for approximating the integral

$$Q = \int_a^b f(t) dt$$

There are advantages and drawbacks in this solution, the main advantage is undoubtedly the intrinsic simplicity.(see below the example for batch separation in Pc code).

Among the drawbacks there is the difficult to define properly the

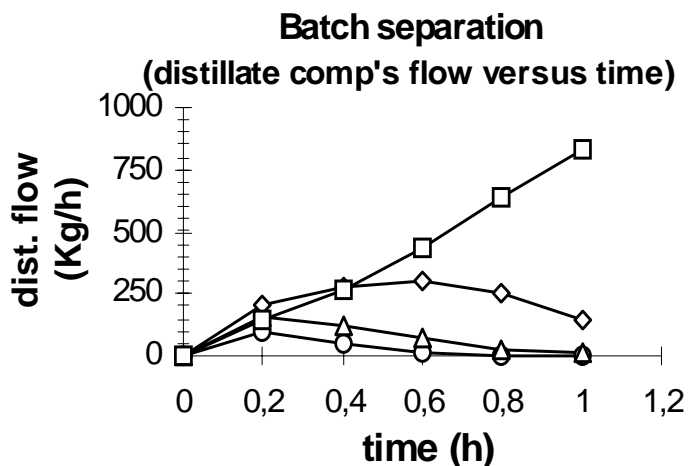
step-size, where the function to integrate is sharply concentrate in one or more peaks one is forced to utilize a very high number of steps and the method becomes impractical. Of course one can use an adaptive system to achieve some predetermined accuracy in the solution, but this lacks the intrinsic simplicity of the method. That above is not, however, a common case in thermodynamic calculations and usually the direct integration technique works quite well.

In simulation we use the method HPFlash() included in library , the function performs a VLE flash considering the addition of a quantity of heat (Dh).

That function has a widely area of application, a useful utilize is in the design of exchangers (a process calculator can generate the VLE and transport properties curves).

```
for (count=0;count<nsteps (steps);count = count+1)
{
    tcalc=HPFlash(stream, tcalc,Pcalc,Pcalc,Dh);
    UpdateLiqComp();
}
```

The results of simulation have been shown in graph, notice the heavy component (n-Hexane) which constitutes almost all the distillate flow at the end of warm-up period.



Case 4, depressuring a vessel

When the external surface of a pressurized vessel is exposed to fire a quantity of heat is absorbed. Depending from thermal resistances, equivalent specific heats and mass ratios the heat from walls may be transferred almost all to the fluids contained in vessel or produce a large temperature increase in plates. In that situation the metal might reach a temperature higher than the design for that vessel and a stress rupture might occur also with an internal pressure lower than the vessel's design gage pressure. The use of an emergency vapor depressuring system is one method of avoiding such an occurrence, the system provides an adequate venting capacity to permit reduction of the vessel pressure to a level considered safe within a reasonable period of time. Pressure versus time diagram defines the requirements for venting system.

For petrochemical plants, API[1] recommended design practices define as sizing criteria the reduction of pressure to a level equivalent to 50 percent of the vessel's design gage pressure within approximately 15 minutes. The criterion is based on vessel's wall temperature versus stress to rupture and applies to vessels with wall thickness of approximately 1 inch or more. Although there are circumstances which require different approaches that practice is widely applied. The design of a depressuring flow system is one example where a process calculator and the direct integration technique can be useful. Here a short program is proposed to simulate the depressuring of a vessel containing a liquid mixture. The procedure is iterative and the operating is equivalent to that discussed for batch separation. The purpose of the method isn't to provide a sophisticated solution but instead to result in a rough but robust skeleton for subsequent modifications. The algorithm subdivides the total pressure drop in n steps and, at each step, finds the quantity of heat introduced by fire with a iterative procedure.

Since this quantity is related to time which is calculated as :

$$step_time_{(hours)} = \frac{Q_g}{W_{disch}}$$

where :

$step_time$ = time of n step (hours)

Q_g = quantity of gas to discharge(kg)

W_{disch} = discharging flow (kg/h)

and quantity of gas vapor ized is related also to heat added the program utilizes the method of direct substitution to solve the system of nonlinear equations. This might appear expensive in terms of computational resources, an alternative is to use in each step, for example, a simplified model for VLE calculations and assume a

constant heat of vaporization.

The quantity of heat due to fire is calculated with the correlation proposed by API [2]:

$$\Delta H_f = step_time_{(hours)} \cdot 43190 \cdot Wt_{area}^{0.82} \quad (3)$$

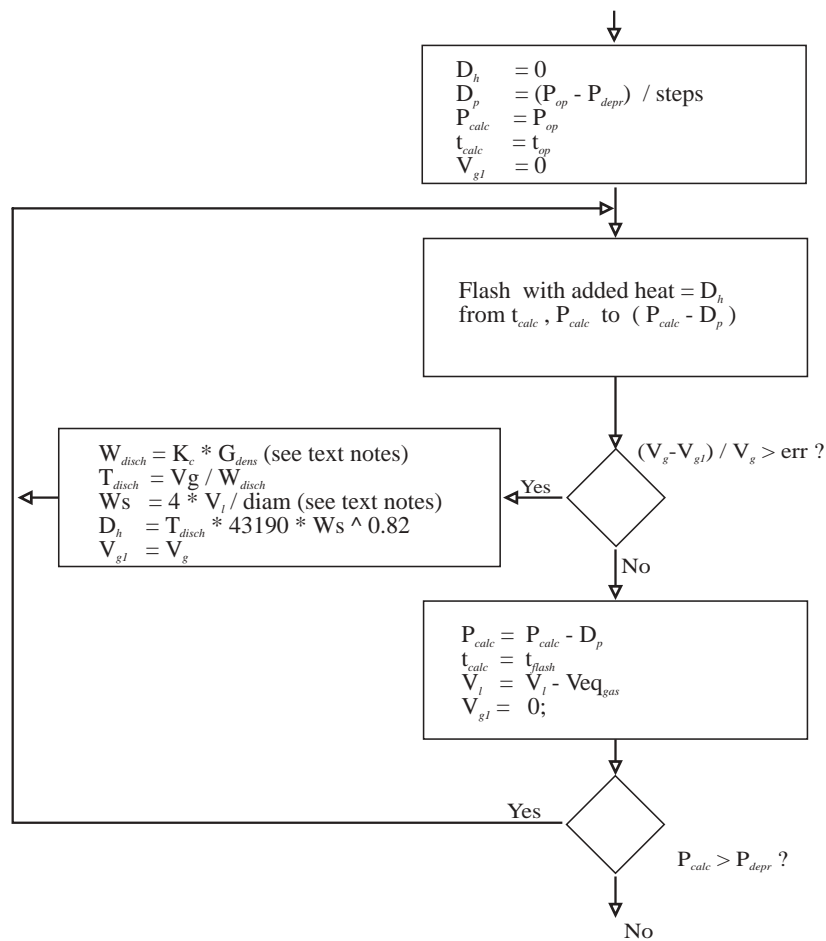
The procedure doesn't take in account for the distribution of heat through the plates exposed to fire (which is a important contribute with high values of wall's thickness). A better solution should be to include a finite difference method to calculate the temperatures distribution in plates, this should permit to estimate the maximum wall's temperature, also. Descriptions of such methods are given in literature [3]. In the procedure the value of wetted surface is updated at each iteration according to effective liquid level, a correlation for a vertical cylindrical vessel is included, wetted area is approximated (neglecting base contribute) as :

$$Wt_{area} = \frac{4 \cdot vol}{diam} \quad (4)$$

where :

$$Wt_{area} = \text{wetted area} \quad (m^2)$$

$$vol = \text{liquid volume} \quad (m^3)$$



```

Dh      = 0;
Dp      = (Pop-Pdepr) / steps (steps);
Pcalc   = Pop - Dp;
tcalc   = top;
Vgl     = 0;
while (Pcalc > Pdepr) {
  tflash = HPFlash (stream,tcalc,Pcalc,Pcalc,Dh);
  gasdens= StrmGasDens (stream,tcalc,Pcalc);
  Vg     = Lqty * (1 - StrmLiqVal (stream, tcalc, Pcalc) / gasdens;
  while ((Vg - Vgl) / Vg > err) {
    wdisch = calcWdisch (stream, tcalc, Pcalc); /* user defined method */
    Tdisch = Vg / wdisch;
    ws     = 4 * Lqty / (diam * StrmLiqDens(stream, tcalc, Pcalc));
    Dh     = Tdisch * 43190 * ws ** 0.82;
    tflash = HPFlash (stream,tcalc,Pcalc,Pcalc,Dh);
    Vg     = Lqty * (1 - StrmLiqVal (stream, tcalc, Pcalc) / gasdens;
  }

  Pcalc   = Pcalc - Dp;
  tcalc   = tflash;
  Dh      = 0;
  Vgl     = 0;
  UpdateLiqComp();
}

```

In order to evaluate the results a design case is proposed. A vertical cylindrical vessel with a base diameter of 2 meters contains about 10,000 kg of hydrocarbons in liquid phase (100 C @

20 Bar, Soave-Redlich-Kwong VLE model), simulation of depressuring cycle starts at 20 Bar @ 100 C and ends at 10 Bar . Three different depressuring flows (8000 10,000 and 15,000 Kg/h) have been evaluated producing three separate simulations, the results are shown in graphs. In simulations the value of depressuring flow has been fixed for entire cycle, in real operating however this value varies depending from gas densities, operating conditions etc. , the program includes a calculation routine (calcWdisch) which permits to enter appropriate equations as operating parameters for discharging lines, flow orifices etc. Also in that case the direct integration permits to find a numerical solution for a problem which might be hard to treat analytically.

The first graph shows the simulation with a depressuring flow of 8000 Kg/h, due to relatively long discharging time (48 min.) the final quantity of liquid in vessel is 3500 Kg (about 70 % of liquid vapor izes).

In the third graph the estimated discharging time is about the requirements (15 minutes) and the final liquid temperature is calculated in 82.6 C , from figure appears that heat absorbed from liquid due to vapor ization always exceeds the heat absorbed by liquid from fire.

The liquid temperature parameter should not be considered representative of vessel temperature since, as previously discussed, higher temperatures are usual in walls not wetted and exposed to fire. Unless it is possible a complex simulation taking in consideration effective heat absorbed and exchanged by walls, the values (time, pressure) suggested by API represent an experienced solution.

REFERENCES

- [1] American Petroleum Institute "Guide for pressure-relieving and depressuring systems", API Recommended Practice 521
- [2] American Petroleum Institute "Recommended practice for the design and installation of pressure-relieving systems in refineries" , API Recommended Practice 520
- [3]M.Necati Ozisik, Heat Transfer, a basic approach, McGraw-Hill 1985

