Notes for the lectures on FORTRAN Programming

Prof. Walter AMBROSINI

Lesson No. 1

A simple program for the analysis of containment pressurization

The physical problem

EVALUATION OF THE FIRST PRESSURE PEAK IN A DRY CONTAINMENT FOLLOWING A LOCA

- **Problem data**
  - **Free containment volume:** V_c **(e.g., 70,000 m^3)**
  - **Initial containment pressure:** $p_{c,0}$ **(e.g., 0.1 MPa)**
  - **Initial containment temperature:** $T_{c,0}$ **(e.g., 30 °C)**
  - **Relative humidity:** $RH_0 = \frac{p_{v,0}}{p_{v,sat}(T_{c,0})} \times 100$ **% (e.g., 100%)**
  - **Volume of the PWR primary system:** V_p **(e.g., 270 m^3)**
  - **Initial pressure of in the primary system:** $p_{p,0}$ **(e.g., 15.5 MPa)**
  - **Average initial temperature of the primary liquid:** $T_{p,0}$ **(e.g., 290 °C)**
- **Simplifying assumptions adopted in manual calculations:**
  - **all the primary fluid is instantaneously released into the containment;**
  - **the heat transfer with the containment walls is negligible;**
  - **pressure is evaluated for the asymptotic condition that would be obtained in the containment after reaching thermal equilibrium between the atmosphere and the liquid pool: note that in actual conditions, superheated steam is also possible;**
  - **the effect due to the presence of compartments is neglected;**
  - **the heat and mass transfers due to the exchange with steam generators, core decay heat and delayed closure of MSIV are neglected**

- **Calculation procedure**

a) Evaluation of initial mass and energy in the containment

a.1) Mass and energy of the initial vapour

Initial partial pressure of the vapour in the containment

$$p_{v,0} = \frac{RH_0}{100} \times p_{v,sat}(T_{c,0})$$

Initial vapour specific volume in the containment:

$$v_{v,0} = v_v(p_{v,0}, T_{c,0})$$

Initial mass of vapour in the containment:

$$M_{v,c0} = \frac{V_c}{v_v(p_{v,0}, T_{c,0})}$$

Initial energy of vapour in the containment:

$$U_{v,c0} = M_{v,c0} u_v(p_{v,0}, T_{c,0})$$

a.2) Initial mass and energy of air in the containment

Initial partial pressure of air

$$p_{a,0} = p_{c,0} - p_{v,0}$$

Initial mass of air

$$M_{a,0} = M_a = \frac{V_c}{v_{a,0}} = V_c \frac{p_{a,0}}{R_a T_{c,0}}$$

Initial energy of air

$$U_{a,0} = M_a c_{v,a} T_{c,0}$$

b) Evaluation of initial mass and energy in the primary system

For the sake of simplicity, it is assumed that all the fluid in the primary system is subcooled liquid (actually, in the pressurizer there is a small bubble of vapour that we will neglect)

Initial specific volume of the liquid in the primary system

$$v_{l,p0} = v_l\left(p_{p,0}, T_{p,0}\right)$$

Initial liquid mass in the primary system

$$M_{l,p0} = \frac{V_p}{v_{l,p0}}$$

Initial liquid energy in the primary system

$$U_{l,p0} = M_{l,p0} u_l(p_{p,0}, T_{p,0})$$

c) Mass and energy balance

On the basis of the adopted assumptions, it is:

- **the mass of air in the containment does not change**

$$M_{a,1} = M_{a,0} = M_a$$

- **the water mass in the primary+containment system does not change before and after the blowdown**

$$M_{w,1} = M_w = M_{l,p0} + M_{v,c0}$$

- **the totel energy in the primary+containment system does not change before and after the blowdown**

$$U_1 = U_0 = U_{l,p0} + U_{v,c0} + U_{a,c0}$$

The final energy of the system is expressed in terms of the final conditions of the fluid in the containment:

$$\boxed{U_1 = M_w \left\{ (1-x)\, u_l\left(p_{c,1}, T_{c1}\right) + x\, u_{v,sat}\left(T_{c1}\right) \right\} + M_a c_{v,a} T_{c1}}$$

where the static quality is given by

$$V_c + V_p = M_w \left\{ (1-x) v_l\left(p_{c,1}, T_{c1}\right) + x v_{v,sat}\left(T_{c1}\right) \right\}$$

from which it is

$$x = \frac{(V_c + V_p)/M_w - v_l\left(p_{c,1}, T_{c1}\right)}{v_{v,sat}\left(T_{c1}\right) - v_l\left(p_{c,1}, T_{c1}\right)}$$

Final pressure is evaluated for a Dalton gas mixture:

$$p_{c1} = p_{sat}(T_{c1}) + \frac{M_a R_a T_{c1}}{V_c + V_p - M_w (1-x) v_l(p_{c1}, T_{c1})}$$

In manual calculations, to simplify the procedure it is assumed that:

$$v_l\left(p_{c,1}, T_{c1}\right) \approx v_{l,sat}\left(T_{c1}\right) \qquad u_l\left(p_{c,1}, T_{c1}\right) \approx u_{l,sat}\left(T_{c1}\right)$$

that is acceptable, since both the specific volume and the internal energy of the liquid depend weakly on pressure.

It is also acceptable to neglect the primary system volume with respect to the containment volume:

$$V_c + V_p \approx V_c$$

We have therefore reached two equations that must be iteratively solved to determine the final temperature:

$$\boxed{x = \frac{V_c / M_w - v_{l,sat}(T_{c1})}{v_{v,sat}(T_{c1}) - v_{l,sat}(T_{c1})}} \qquad \textbf{(1)}$$

$$\boxed{U_1 = M_w \left\{ (1-x)\, u_{l,sat}(T_{c1}) + x\, u_{v,sat}(T_{c1}) \right\} + M_a c_{v,a} T_{c1}} \qquad \textbf{(2)}$$

We proceed as follows:

1. **a trial value of the final temperature is chosen, T_{c1};**

2. **the static quality is evaluated by (1);**

3. **the RHS of (2) is computed; if it is found that:**

   3a. it is close to the LHS within a prescribed tolerance,
   $\Rightarrow$ *the calculation is stopped*

   3b. it is greater of the LHS beyond the tolerance
   $\Rightarrow$ *a lower T_{c1} is chosen restarting from 2*

   3c. it is smaller of the LHS beyond the tolerance
   $\Rightarrow$ *a higher T_{c1} is chosen restarting from 2*

When an automatic computation procedure is used, an appropriate iterative method can be selected (bisection, regula falsi, Newton) evaluating water properties with an appropriate routine

Once the final temperature is determined, the final pressure in the containment is evaluated as for a daltonian mixture of gases

$$p_{c1} = p_{sat}(T_{c1}) + \frac{M_a R_a T_{c1}}{V_c - M_w (1-x) v_{l,sat}(T_{c1})} \approx p_{sat}(T_{c1}) + \frac{M_a R_a T_{c1}}{V_c}$$

where the liquid pool volume has been neglected.

A FORTRAN ROUTINE

```
c-----------------------------------------------------------------------c
c                                                                       c
c     Program for evaluating the first pressure peak in a nuclear       c
c     reactor dry containment with simplified assumptions.              c
c                                                                       c
c     The program should be used for teaching purposes only.            c
c     NO ASSURANCE ABOUT ITS PHYSICAL RELIABILITY IS GIVEN              c
c-----------------------------------------------------------------------c
      program presscon
      implicit double precision (a-h,o-z)
c
      common / final / ama1,amw1,ener1,voltot,cvair,xstat
      common / reftg / trefg
c
      parameter ( rgas = 8.315d00 )
      parameter ( pmair = 28.966d-03 )
      parameter ( cpair = 1.014d03 )
c
      open (unit=5,file='presscon.dat')
      open (unit=6,file='presscon.out')
c
      read(5,*)
      read(5,*) volcon,pcon0,tcon0,trefg
c
      read(5,*)
      read(5,*) volpri,vlpri,ulpri
c
      voltot = volcon + volpri
c
c   Calculation of Containment Initial Conditions with the assumption
c   of Saturated Vapour
c
c   Vapour Properties:
c
      pv0 = psat (tcon0)
      vv0 = vvsat (tcon0)
      uv0 = uvsat (tcon0)
c
      write (6,105) pv0,1.d00/vv0,uv0
  105 format (' Initial Vapour Pressure      = ',1pe14.7,' Pa',/,
     &        ' Initial Vapour Density       = ',1pe14.7,' kg/m3',/,
     &        ' Initial Vapour Sp. Int. En.. = ',1pe14.7,' J/kg',/)
c
c   Vapour Mass and Energy:
c
      amv0 = volcon / vv0
      enerv0 = amv0 * uvsat (tcon0)
c
      write (6,107) amv0,enerv0
  107 format (/,' Initial Vapour Mass      = ',1pe14.7,' kg ',/,
     &        ' Initial Vapour Energy    = ',1pe14.7,' J ',/)
c
c   Air Properties:
c
      rair = rgas / pmair
      cvair = cpair - rair
c
      write(6,108) rair,cvair
```

```
  108 format(/,' Air gas constant          = ',f10.3,' J/(kg K) ',/,
     &         ' Air Cv                    = ',f10.3,' J/(kg K) ',/)
c
      pair0 = pcon0 - pv0
c
      dena0 = pair0 / ( rair * ( tcon0 + 273.15d00 ) )
c
      write (6,110) pair0,dena0
  110 format (/,' Initial Air Pressure         = ',1pe14.7,' Pa',/,
     &        ' Initial Air Density          = ',1pe14.7,' kg/m3',/)

      ama0 = volcon * dena0
      enera0 = ama0 * cvair * ( tcon0 + 273.15d00 - trefg )
c
      write (6,115) ama0,enera0
  115 format (/,' Initial Air Mass      = ',1pe14.7,' kg ',/,
     &        ' Initial Air Energy    = ',1pe14.7,' J ',/)
c
c   Calculation of Primary System Initial Conditions
c
      amlpri = volpri / vlpri
      enepri = amlpri * ulpri
c
      write (6,120) amlpri,enepri
  120 format (/,' Initial Liquid Mass in P.S.  = ',1pe14.7,' kg ',/,
     &          ' Initial Energy in P.S.       = ',1pe14.7,' J ',/)
c
c   Final Containment Mass and Energy
c
      ama1 = ama0
      amw1 = amv0 + amlpri
      ener1 = enerv0 + enera0 + enepri
c
      write (6,125) ama1,amw1,ener1
  125 format (/,' Final Air Mass in Containment    = ',1pe14.7,' kg ',/,
     &          ' Final Water Mass in Containment  = ',1pe14.7,' kg ',/,
     &          ' Final Energy in Containment      = ',1pe14.7,' J ',/)
c
c   Initial Approximations
c
      ta = 5.d00
      fta = fres (ta)
c
      tb = 200.d00
      ftb = fres (tb)
c
      prod = fta * ftb
c
          if(prod.gt.0.d00) then
          write(6,200)
          write(*,200)
  200     format(/,/,' The temperature range does not separate a root ',
     &    /,/)
          stop
          endif
c
c   Iteration Loop
c
      do 10 iter = 1,1000
c
         if(iter.le.100) then
         ratio = ( tb - ta ) / ( ftb - fta )
         tx = ta - fta * ratio
```

```
c
         else
         tx = 0.5d00 * ( ta + tb )
         endif
c
      ftx = fres (tx)
c
      write(6,150) iter,tx,ftx
      write(*,150) iter,tx,ftx
  150 format(1x,' Iter = ',i5,' T = ',f10.2,' F(T) = ',1pe14.7)
c
      if(dabs(ftx).lt.1.d00) goto 20
c
      prod = ftx * fta
c
          if(prod.lt.0.d00) then
          tb = tx
          ftb = ftx
          else
          ta = tx
          fta = ftx
          endif
c
   10 continue
c
   20 tcfin = tx
c
c  Calculation of Vapour and Air Partial Pressures
c
      pvfin = psat (tcfin)
c
      volair = voltot - amw1 * ( 1.d00 - xstat ) * vlsat (tcfin)
      pafin = ama1 * rair * ( tcfin + 273.15d00 ) / volair
c
      pfin = pvfin + pafin
c
      write(6,100) tcfin,pfin,pvfin,pafin
c
      write(*,100) tcfin,pfin,pvfin,pafin
c
      stop
  100 format (/,' Final Temperature = ',f14.2,' °C',/,
     &        ' Final Pressure =    ',1pe14.7,' Pa',/,
     &        ' Final Vapour P. =   ',1pe14.7,' Pa',/,
     &        ' Final Air P.    =   ',1pe14.7,' Pa',/)
c
      end
c-----------------------------------------------------------------------c
c                                                                       c
c    Function of the residual as a function of temperature              c
c                                                                       c
c-----------------------------------------------------------------------c
      double precision function fres (temp)
      implicit double precision (a-h,o-z)
c
      common / final / ama1,amw1,ener1,voltot,cvair,xstat
      common / reftg / trefg
c
      vcovmw = voltot / amw1
      vl = vlsat (temp)
      vv = vvsat (temp)
c
      anumx = vcovmw - vl
```

```
      adenx = vv - vl
c
      xstat = anumx / adenx
      cmpxst = 1.d00 - xstat
c
      ul = ulsat (temp)
      uv = uvsat (temp)
c
      enera1 = ama1 * cvair * ( temp + 273.15d00 - trefg )
c
      uw1 = cmpxst * ul + xstat * uv
      enerw1 = amw1 * uw1
c
      fres = ener1 - enera1 - enerw1
c
      return
      end
c-----------------------------------------------------------------------c
c                                                                       c
c   WARNING: The water property relationships in this program give      c
c            reasonable approximations of the actual values in the      c
c            temperature range 5 °C < T < 200 °C. However, they have    c
c            not been extensively tested                                c
c                                                                       c
c-----------------------------------------------------------------------c
c                                                                       c
c    Function for evaluating saturation pressure                        c
c                                                                       c
c-----------------------------------------------------------------------c
      double precision function psat (temp)
      implicit double precision (a-h,o-z)
c
      dimension coeff(7)
      data coeff /-6.12335d-09,4.70372d-06,4.41367d-05,3.96479d-02,
     &1.09506d+00,4.78970d+01,6.00109d+02/
c
c  Control on temperature values out of the domain boundary
c
          if ( (temp.gt.200.d00).or.(temp.lt.5.d00) ) then
          write(6,100)
          write(*,100)
  100     format(/,/,' Out of property validity range ',/,/)
          endif
c
c   Polynomial law
c
      psat = coeff(1)
c
      do 10 ic = 2,7
      psat = psat * temp + coeff(ic)
   10 continue
c
      return
      end
c-----------------------------------------------------------------------c
c                                                                       c
c    Function for evaluating saturated liquid specific volume           c
c                                                                       c
c-----------------------------------------------------------------------c
      double precision function vlsat (temp)
      implicit double precision (a-h,o-z)
c
      dimension coeff(7)
```

```
      data coeff /1.22545d-18,-8.43907d-16,2.65284d-13,-4.54617d-11,
     &7.41429d-09,-4.63022d-08,1.00016d-03 /
c
c  Control on temperature values out of the domain boundary
c
          if ( (temp.gt.200.d00).or.(temp.lt.5.d00) ) then
          write(6,100)
          write(*,100)
  100     format(/,/,' Out of property validity range ',/,/)
          endif
c
c   Polynomial law
c
      vlsat = coeff(1)
c
      do 10 ic = 2,7
      vlsat = vlsat * temp + coeff(ic)
   10 continue
c
      return
      end
c-----------------------------------------------------------------------c
c                                                                       c
c    Function for evaluating saturated vapour specific volume           c
c                                                                       c
c-----------------------------------------------------------------------c
      double precision function vvsat (temp)
      implicit double precision (a-h,o-z)
c
      dimension coeff1(7),coeff2(7)
      data coeff1/4.44428d-10,-2.24966d-07,4.67443d-05,-5.15763d-03,
     &3.25689d-01,-1.16109d+01,1.94440d+02/
      data coeff2/3.30034d-12,-3.16326d-09,1.26899d-06,-2.74159d-04,
     &3.39127d-02,-2.30607d+00,6.87450d+01 /
c
c  Control on temperature values out of the domain boundary
c
          if ( (temp.gt.200.d00).or.(temp.lt.5.d00) ) then
          write(6,100)
          write(*,100)
  100     format(/,/,' Out of property validity range ',/,/)
          endif
c
c   Polynomial law
c
          if(temp.lt.100.d00) then
c
          vvsat = coeff1(1)
c
          do 10 ic = 2,7
          vvsat = vvsat * temp + coeff1(ic)
   10     continue
c
          else
c
          vvsat = coeff2(1)
c
          do 20 ic = 2,7
          vvsat = vvsat * temp + coeff2(ic)
   20     continue

          endif
c
```

```
      return
      end
c-----------------------------------------------------------------------c
c                                                                       c
c    Function for evaluating saturated liquid specific internal energy c
c                                                                       c
c-----------------------------------------------------------------------c
      double precision function ulsat (temp)
      implicit double precision (a-h,o-z)
c
      dimension coeff(7)
      data coeff /-1.23164E-09,7.72062E-07,-1.74864E-04,2.08691E-02,
     &-1.15431E+00,4.20685E+03,4.05456E+01/
c
c  Control on temperature values out of the domain boundary
c
          if ( (temp.gt.200.d00).or.(temp.lt.5.d00) ) then
          write(6,100)
          write(*,100)
  100     format(/,/,' Out of property validity range ',/,/)
          endif
c
c   Polynomial law
c
      ulsat = coeff(1)
c
      do 10 ic = 2,7
      ulsat = ulsat * temp + coeff(ic)
   10 continue
c
      return
      end
c-----------------------------------------------------------------------c
c                                                                       c
c    Function for evaluating saturated vapour specific internal energy c
c                                                                       c
c-----------------------------------------------------------------------c
      double precision function uvsat (temp)
      implicit double precision (a-h,o-z)
c
      dimension coeff(7)
      data coeff /-1.39126d-10,9.35497d-08,-3.14164d-05,-2.88782d-03,
     &-1.15210d-01,1.38053d+03,2.37521d+06 /
c
c  Control on temperature values out of the domain boundary
c
          if ( (temp.gt.200.d00).or.(temp.lt.5.d00) ) then
          write(6,100)
          write(*,100)
  100     format(/,/,' Out of property validity range ',/,/)
          endif
c
c   Polynomial law
c
      uvsat = coeff(1)
c
      do 10 ic = 2,7
      uvsat = uvsat * temp + coeff(ic)
   10 continue
c
      return
      end
```