Thermoptim model of a reverse osmosis unit

We have developed an external class in which the salt solution is modeled by a **boiling point elevation** quadratic function of the concentration, the other thermodynamic properties being those of water (class EauSalee).

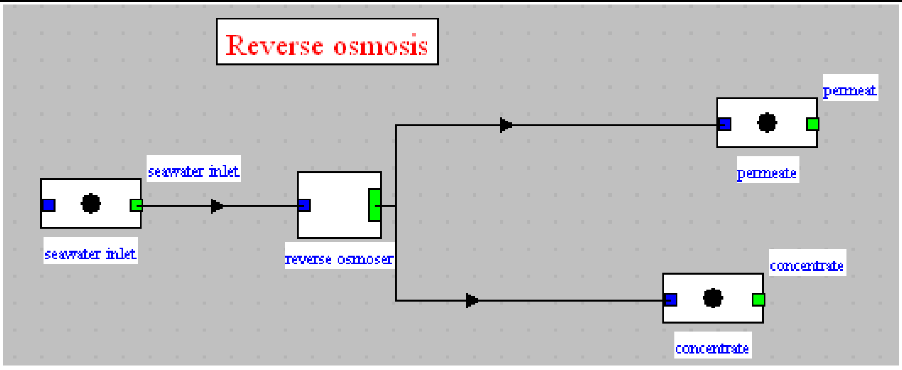

Figure 1: Diagram of the component

In an operation of desalination, a process is to perform a reverse osmosis.

A reverse osmosis unit behaves like a divider receiving at its input the salt water under pressure and from which exit two fluids, the permeate corresponding to the purified water and the concentrated solution.

The model structure of the unit is given in Figure 1.

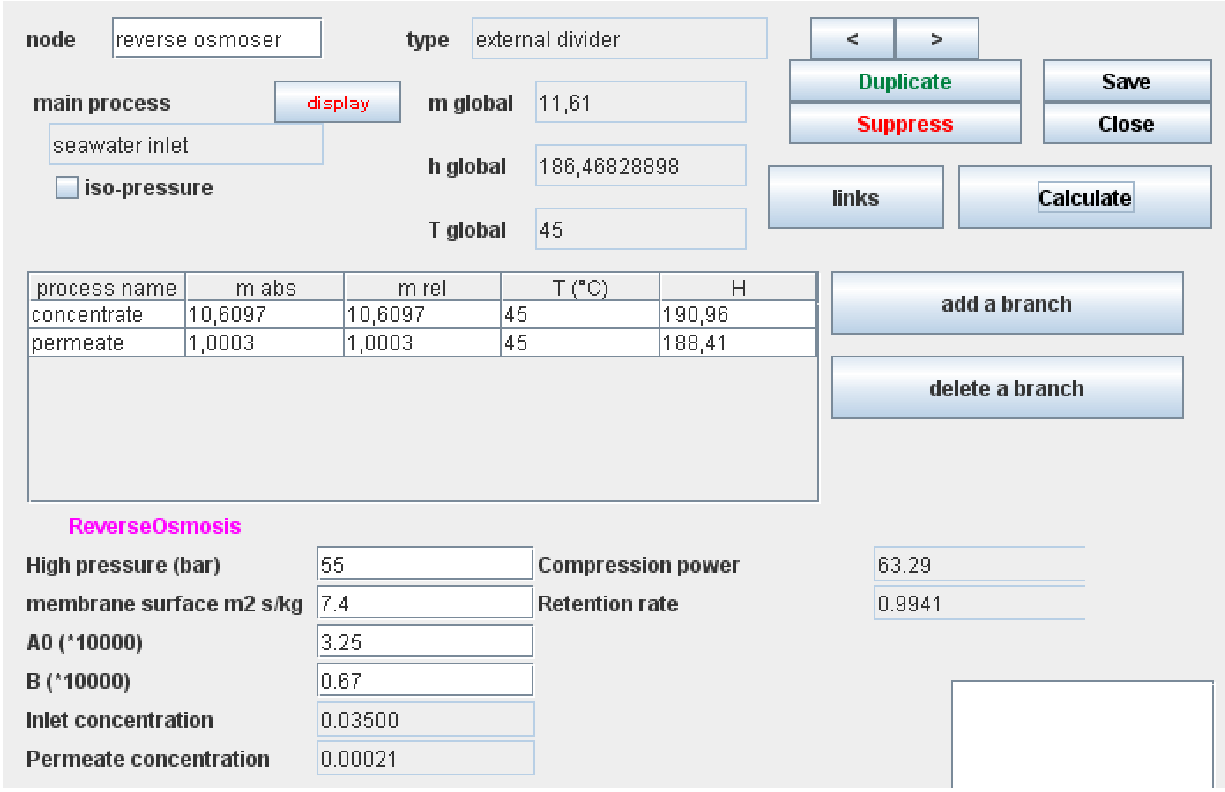

Figure 2: Component screen

Physical model

The energy provided corresponds only to the work of compression of the initial solution, and is therefore much lower than that put into play (as heat) in most other desalination devices.

The law of van't Hoff (1) states that the osmotic pressure exerted by the solute is equal to the pressure it would have exercised in the perfect gas state in the same volume and at the same temperature. If the solute is dissociated as ions, the osmotic pressure is multiplied by the number or ions present.

$$\pi = n_i \, X \, R \, T \qquad (1)$$

For sea water, the order of magnitude of π is 25 to 30 bar.

ΔP is the pressure difference across the membrane, it can be shown that the flow of solvent J_e through the membrane is given by (2).

$$J_e = A \, (\Delta P - \Delta\pi) \qquad (2)$$

A is called water permeability of the membrane. This is a characteristic parameter of the membrane, which depends on temperature according to an Arrhenius type law (3).

$$A = A_0 \exp\left[\frac{E}{R} \left(\frac{1}{298} - \frac{1}{T} \right) \right] \qquad (3)$$

Although there is a preferential transfer of the solvent, one cannot prevent a small fraction of the solute from crossing the membrane. The flow of solute Js is given by (4). It is proportional to the difference in concentration.

$$J_s = B \, \Delta X \qquad (4)$$

B is the salt permeability of the membrane. It depends on the membrane but not the temperature.

The conversion rate is the ratio of flow rate through the membrane to the feed rate, and the retention rate is the ratio of the concentration difference between the initial solution to the permeate concentration of the initial solution.

Note that the system of equations above corresponds to a set of quadratic equations in Xperm Xconc that we chose to solve by iteration.

Figures 2 and 3 show the screen of the component and the Thermoptim synoptic view of the model of a membrane FilmTech SW30-4040 from Dow Chemical used to desalinate seawater

The values of A and B were estimated from the manufacturer's documentation.

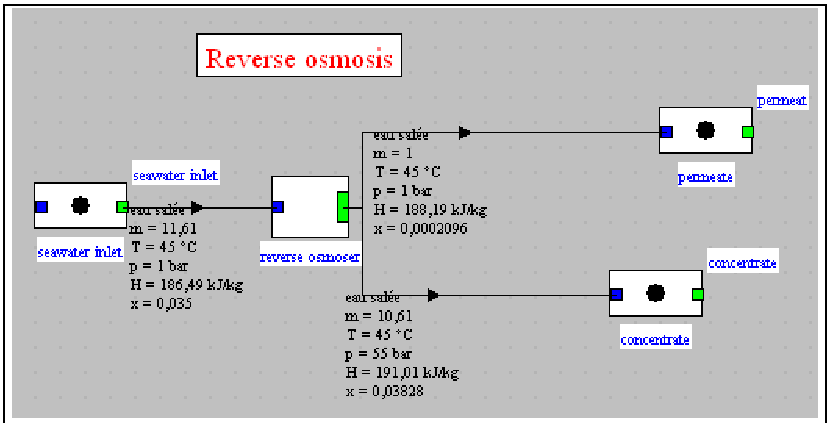

Figure 3: Synoptic view of the component

Compression power to produce 1 kg/s of fresh water is equal to 63.3 kW, which is almost double that required by the mechanical vapor compression. However, it is possible to recover a part of this power by expanding the concentrate in a turbine coupled to the compressor.

Study of the external class ReverseOsmosis

Consistency tests on the construction of the external divider are made by the method checkConsistency() to check that the fluids are well connected: in this case, a product as inlet and the product and water as outlet. The tests implemented here are very basic and could be improved. Please refer to Volume 3 of the reference manual for explanations on this point, valid for all external nodes.

The study of the external class ReverseOsmosis shows how the model was implemented. Three steps allow one to perform the calculations:

1) we first perform the initialization from the values entered on the screen of the component and those of the inlet process, and a method searchs the solution by dichotomy on the salt concentration of permeate Xperm

```
Pconc=Util.lit_d(HP_value.getText());
double tauCompr=Vinlet*(Pconc-Pinlet)*100*minlet;
tauCompr_value.setText(Util.aff_d(tauCompr,2));
A0=Util.lit_d(A0_value.getText())*1e-4;
B_=Util.lit_d(B_value.getText())*1e-4;
At=getmembraneWaterPermeability(Tinlet);

deltaPtm=(Pconc-Pperm);//  (bar)
Amembr=Util.lit_d(Amembr_value.getText())*minlet;

Xperm=Util.dicho_T(this, 0, 0, "calcXperm", 0, Xinlet/10, 0.0001);
```

2) the method calculates the residue resXperm of the value of Xperm, calculated in an iterative manner between 0 and Xinlet/10

```
public double f_dicho(double X, double T,String fonc){

    if (fonc.equals("calcXperm"))return resXperm(X);

    return 0;
}

double resXperm(double X){

        deltaPitm=((Xconc+Xinlet)/2-X)*2/M_NaCl*R_/1.01325*Tinlet;//MPa
        deltaPitm=deltaPitm*10;//bars

        mperm=Amembr*At*(deltaPtm-deltaPitm);//débit de perméat
        mconc=minlet-mperm;//débit de la solution concentrée

        Js=B_*Amembr*((Xconc+Xinlet)/2-X);//débit de sel dans le perméat

        Xperm=Js/mperm;
        Xconc=(minlet*Xinlet-Js)/mconc;//concentration en sortie

        double z=1000*(Xperm-X);
        return z;
}
```

3) the retention rate is then determined, and the node is updated using the generic methods described in the reference manual

```
    double R=(Xinlet-Xperm)/Xinlet;//retention rate
    R_value.setText(Util.aff_d(R,4));

    vTransfo= new Vector[nBranches+1];
    vPoints= new Vector[nBranches+1];
    setupPoorSolution(minlet,Tinlet,Pinlet,Xinlet);
    setupConcSolution(mconc,Tinlet,Pconc,Xconc);
    setupBuees(mperm,Tinlet,Pperm,Xperm);
    updateDivider(vTransfo,vPoints,Tinlet,Hinlet);
    de.updateProcess(setEnergyTypes(permProcess,tauCompr,0,0));


    Xinlet_value.setText(Util.aff_d(Xinlet,5));
    Xperm_value.setText(Util.aff_d(Xperm,5));
    double z=At/A0;
    System.out.println("deltaPtm: " + deltaPtm + "\tdeltaPitm "+ deltaPitm);
    System.out.println("At/A0 "+ z+ "\tJs " + Js+ "\tR " + R);
}
```