Guidance page for practical work : trigeneration by micro turbine and absorption cycle

1) Objectives of the practical work

The objective of this practical work is to show how, with a marginally reduced work, it is possible to add multi-functional custom components, relatively complicated in terms of thermodynamics, to the simpler ones predefined in the core of Thermoptim, that can be described as mono-functional since they involve only one form of energy (mechanical or thermal). In doing so, it is possible to easily model complex installations involving many components, as for a trigeneration plant.

2) References

[1] GICQUEL R., Energy Systems: A New Approach to Engineering Thermodynamics, CRC Press, 2011.

[2] ASHRAE, Fundamentals Handbook (SI), Thermodynamics and Refrigeration Cycles, 2001.

3) Main practical work

3.1 Statement

To extend the core of Thermoptim, we adds plug-ins written in the Java language, which define both the equations and the graphical interface. These additional elements are dynamically loaded when launching the software. The implementation of Thermoptim extension mechanism by adding external classes led to introduce two semantic distinctions that are important: first the one just defined between the mono- and multi-functional components, and the other one related to a new concept in the software package, that of thermocoupler. The thermocoupler are intended to complement conventional heat exchangers allowing components other than "exchange" processes to connect to one or more "exchange" processes to represent a thermal coupling.

This mechanism has a number of benefits, because it can be used to represent many thermal couplings that do not constitute a heat exchange in the traditional sense, like for example cooling the walls of the combustion chamber of a reciprocating engine, cooled compression, and above all supply or removal of heat from multi-functional external components.

We can continue to benefit of the whole Thermoptim environment, i.e. all available components and the diagram editor that allows you to very easily describe the internal structure of the system studied. Not only in such a way do we significantly simplify the modeling process and facilitate subsequent use and maintenance of the model, but mostly we secures its construction by automating the establishment of linkages between its components and ensuring consistency . This is especially important when the system under study includes a large number of components.

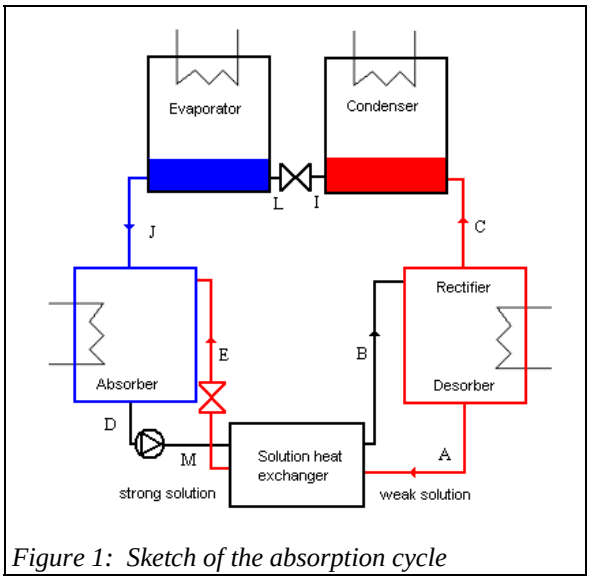

Figure 1: Sketch of the absorption cycle

A micro-gas turbine is a gas turbine of small capacity (several tens of kW), typically operating with a low compression ratio and with a regenerator to improve performance.

We are interested here in a trigeneration plant where gases coming out of a 125 kWe micro-turbine are used both to provide the necessary heat to the desorber of a LiBr-H_2O absorption machine (Figure 8.6.1) and secondly to

produce 0.5 kg/s of hot water at about 80 °C. The turbine sucks 0.78 kg/s of air that is compressed at 5 bar, then passes through a regenerator before being heated to 950 °C in the combustion chamber burning natural gas. Gases are expanded at 650 °C and then pass successively through the regenerator, the desorber and the cogeneration heat exchanger. A gas compressor is required to raise the pressure of natural gas from the distribution network. This is a variant of CHP example presented section 8.5.1 of [1].

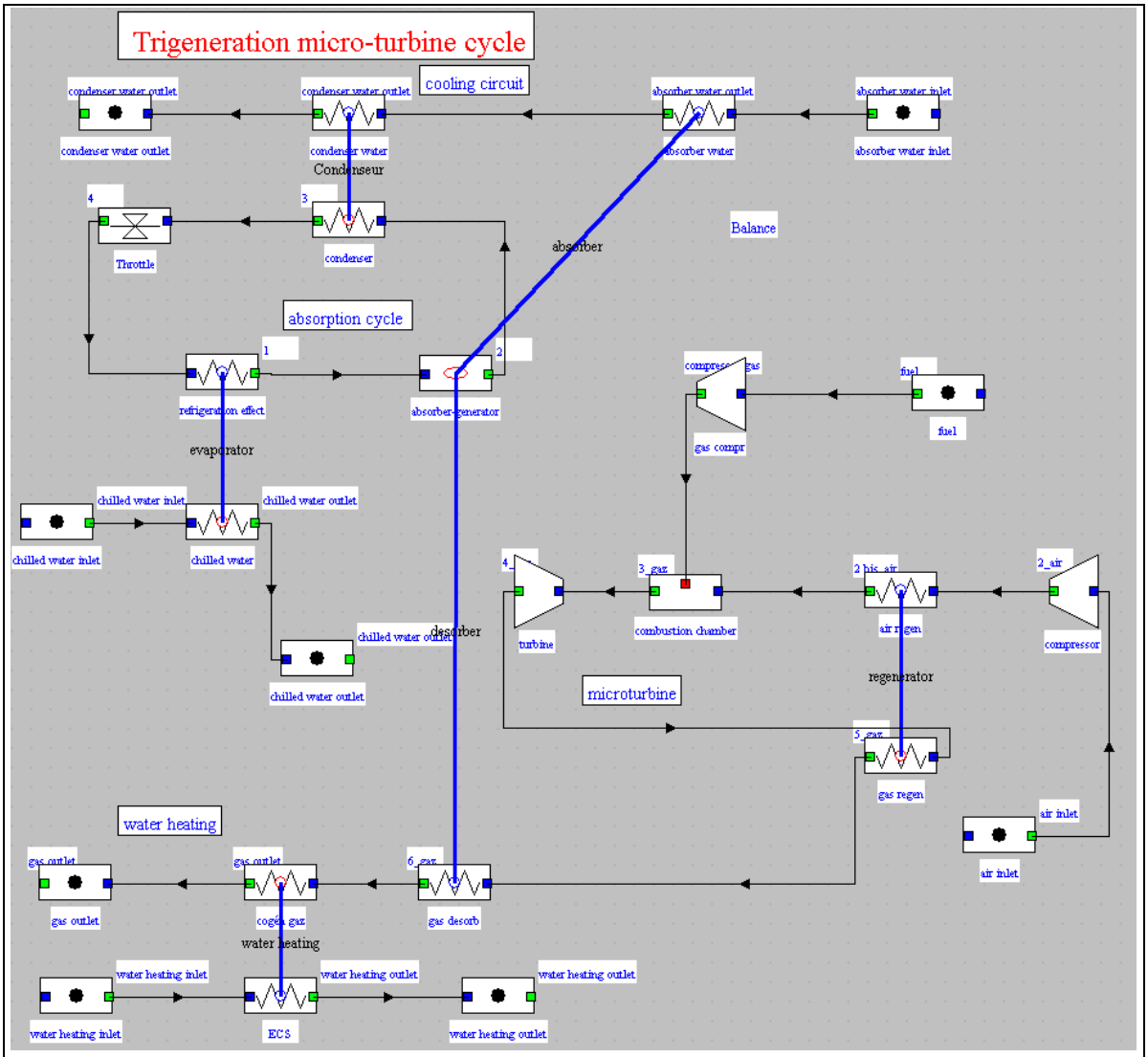

Figure 2: Diagram of the overall trigeneration facility

The modeling of this facility uses thirty components, representing several hundred coupled equations. All these components are available in Thermoptim core, with the exception of sub-system (absorber-solution exchanger-desorber) of the absorption refrigeration cycle (the three lower elements of the sketch in Figure 8.6.1), which replaces the compressor of a vapor compression cycle. We thus understand the interest of the external class mechanism in a case like this: by adding a specialized component to represent the missing module, the work involved is much smaller than if we had to write a program to model the whole cycle.

So we just have to create an external component to represent the module which does not exist, which firstly involves the LiBr-H_2O pair, whose properties can be modeled either directly in the external component or as a particular external substance. Furthermore this module requires both a heat supply at high temperature in the desorber, and heat extraction at medium temperature in the absorber. Representation of thermal coupling is possible using two thermocouplers, called "absorber" and "desorber" on the diagram in Figure 2: the external component "generator-absorber" calculates the thermal energy to be exchanged, and each thermocoupler recalculates the "exchange" process to which it is connected.

In this document, we explain how in practice to create this component, after giving results of the entire system simulation. To be easily comprehensible, the model presented here is relatively simple thermodynamically, easily calculable with the assumptions made. It thus presents several limitations: in particular, as it is implemented, it allows a direct calculation of the component knowing the temperature of the absorber and the desorber and the solution exchanger effectiveness, but not an inverse calculation where we seek to determine, for given operating conditions, one of these quantities. However, it has the great advantage of being already sufficiently complicated to show clearly problems that arise during the creation of an external component, and solutions that help solve them.

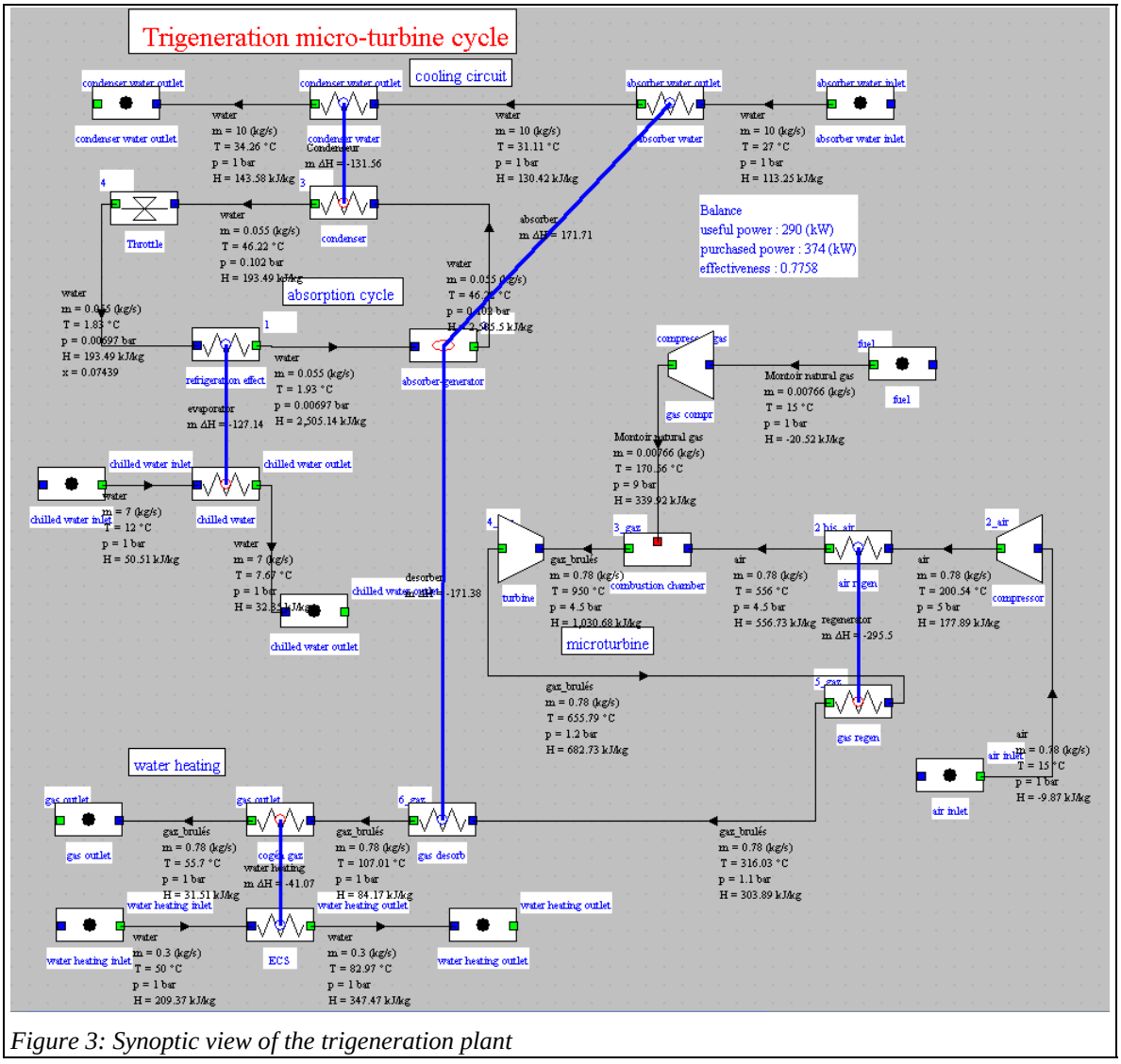

Figure 3: Synoptic view of the trigeneration plant

3.2 Model Results

Once the missing component is created, modeling the trigeneration plant poses no particular problem, and leads to the diagram in Figure 2, in which we recognize in the top left the refrigeration cycle with its cooling circuit, in the middle-right the micro-turbine, and in the lower left the hot water production exchanger.

The setting of the micro-turbine is that of an existing industrial machine, but the trigeneration plant itself is fictitious: we have simply sought to recover the available enthalpy in the gas turbine exhaust to produce cooling and heat hot water.

In the synoptic view of Figure 3, useful energy includes not only the compression and expansion work, but also the cooling energy and thermal energy supplied to the water system, which leads, for these conditions, to an overall efficiency of about 78% which corresponds to approximately 125 kW of electrical power, 176 kW of

heat input to the desorber, providing 127 kilowatts of cooling power, and 40 kW of hot water heat, for a fuel consumption of 375 kW. Mechanical efficiency is 32.7%, and heat-power ratio 1.76. The COP of the single-effect absorption cycle is equal to 0.72.

3.3 Design of the external component

General

The missing component is the external subsystem (absorber- solution exchanger –desorber) of the absorption cycle using the pair LiBr-H_2O (Figure 1). The model used is that of a simple effect machine with the following assumptions (note that in what follows, for reasons of consistency of notations, strength or weakness of the solution is expressed relative to the refrigerant (water) and not to the solvent (lithium bromide), unlike the usual convention in the U.S., which is that adopted by ASHRAE):

- the desorber and condenser are at the same pressure;
- the absorber and evaporator are at the same pressure;
- the refrigerant vapor leaving the evaporator is saturated;
- the liquid refrigerant leaving the condenser is saturated;
- the refrigerant vapor leaving the desorber is saturated at the equilibrium temperature of the solution weak in refrigerant at desorber pressure;
- the weak solution is boiling at the exit of the desorber, assumed isothermal;
- the strong solution leaves saturated the absorber, assumed isothermal.

Model parameters are:

- the absorber and desorber temperatures;
- the heat exchanger solution effectiveness.

The input data of the model are as follows (provided by other system components):

- the absorber and desorber pressures;
- the refrigerant flow-rate.

The outputs are:

- the absorber and desorber thermal loads;
- the weak and strong solution concentrations;
- the weak and strong solution flow-rates.

GUI

A component graphical interface can be deduced (Figure 4). The bottom third of the screen has to be created, the rest being defined as Thermoptim standard.

Parameters correspond to the first three lines added, the results of calculations to others. Below are listed the two types of thermocouplers that the component defines and sets.

The input data are provided by other components of the global model: the refrigerant flow set by the upstream "cooling effect" process is here 0.055 kg/s, the absorber pressure (point 1) is equal to 0.00697 bar, corresponding to an evaporation temperature of 1.93 °C and desorber pressure (point 2) is equal to 0.102 bar, corresponding to a condensing temperature of 46.22 °C.

A peculiarity of this component is that it does not change its downstream point 2, whose state is considered given. It would of course be possible to approach the problem from another angle, but as we said earlier, we do not want to overcomplicate things in this example.

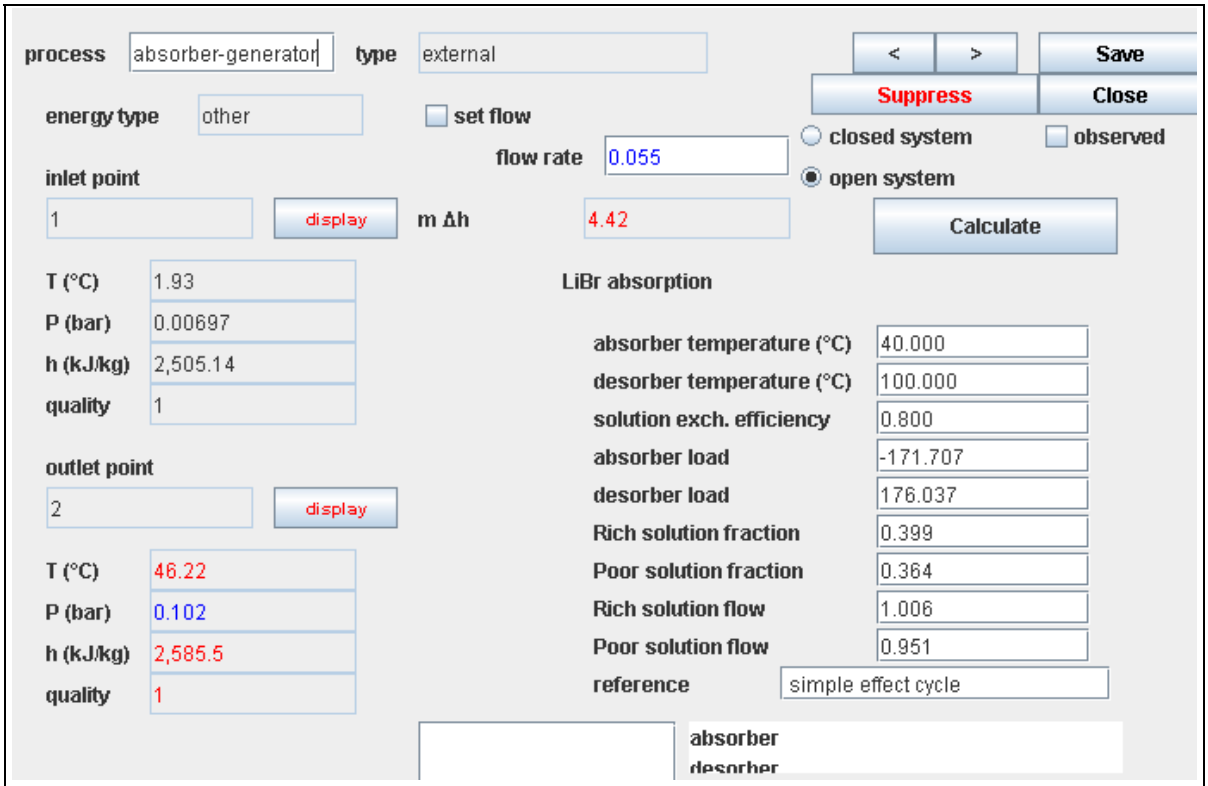

Figure 4: GUI of the external component

Thermodynamic model.

LiBr-H2O mixture

The LiBr-H2O mixture is modeled using the equations proposed by ASHRAE.

Absorber

The weak refrigerant solution is sprayed into rain and washes the refrigerant vapor at low pressure, which is absorbed, releasing its heat of condensation and heat of dilution.

This heat Q_{abs} is extracted by cooling water which then cools the condenser.

With the assumption that the absorber is at a constant temperature T_{abs} and the strong solution is saturated, the absorber equations are the following:

The equation of solution saturated vapor pressure $P_{abs} = P(x_{sr},T_{abs})$ provides the concentration of the strong saturated solution and therefore its enthalpy $h_{srD} = h(x_{sr},T_{abs})$.

Known data: m_r, h_{r1}, T_{abs}, P_{abs}

Unknown: m_{sr}, m_{sp}, x_{sp}, h_{spE}

Conservation of mass: $m_r + m_{sp} = m_{sr}$

Conservation of mass of solution: $(1 - x_{sp})\, m_{sp} = (1 - x_{sr})\, m_{sr}$

These two equations provide m_{sp} and m_{sr} if x_{sr}, x_{sp} and m_r are known:

$$m_{sr} = m_r \frac{1 - x_p}{x_r - x_p}$$

$$m_{sp} = m_r \frac{1 - x_r}{x_r - x_p}$$

Conservation of enthalpy: $m_r h_{r1} + m_{sp} h_{spE} = m_{sr} h_{srD} + Q_{abs}$

There are 8 variables, 4 known and 3 equations.

Generator/desorber

The solution strong in refrigerant is introduced at high pressure in the high-temperature generator where it boils by contact with tubes heated either directly by a fuel or by steam. Steam produced is almost pure refrigerant, due to the saturation pressure difference between the two fluids. It is then directed to the condenser. The depleted solution is extracted to be recycled.

With the assumption that the generator is at a constant temperature T_{gen} and the weak solution is saturated, the equations are:

The inversion of the equation of the solution saturated vapor pressure $P_{gen} = P(x_{sp},T_{gen})$ provides concentration x_{sp}, and enthalpy h_{spA}

New data: h_{r2}, T_{gen}, P_{gen}

New unknowns: h_{srB}, h_{spA}

Conservation of enthalpy: $m_r h_{r2} + m_{sp} h_{spA} = m_{sr} h_{srB} + Q_{gen}$

There are 5 variables, 3 data and 1 equation.

Solution exchanger

At this stage, two equations are missing to solve the model. They correspond to the solution heat exchanger:

$\varepsilon = f(T_A,T_B,T_D,T_E)$
$m_{sp} (h_{spA} - h_{spE}) = m_{sr} (h_{srB} - h_{srD})$

Sequence of calculations

Specifically, the sequence of calculations is as follows:

1) update the component before calculation by loading the values of m_r, h_{r1} and P_{abs} from the upstream point;
2) reading T_{abs}, T_{gen} and ε on the external component screen;
3) inversion of $P_{abs} = P(x_{sr},T_{abs})$ for x_{sr} then h_{srD};
4) loading values h_{r2} and P_{gen} from the downstream point;
5) reverse $P_{gen} = P(x_{sp},T_{gen})$ for x_{sp} and then h_{spA};
6) computing of m_{sp} and m_{sr};
7) calculation of T_B and T_E thanks to solution exchanger equations;
8) calculation of h_{srB} and h_{spE} assuming that these points are in equilibrium at their respective temperature and concentration;
9) calculation of thermal loads Q_{abs} and Q_{gen};
10) update of the external component display;

Now consider the problems encountered in practice at each of these steps. In the following we try to explain as simply as possible to show the principle of model building. For further details please refer to the documentation of Thermoptim on external classes. Some excerpts of the code are given below, but it is recommended to read the rest of this paper having in front the entire class LiBrAbsorption.

1) component update by loading values of m_r, h_{r1}, and P_{abs} from the upstream point

The small difficulty here is that a external component does not have direct access to the simulator variables: these quantities are obtained by special generic methods which build Vectors of different structure depending on the desired object.

The procedure is not complicated but needs to be met:

```
String[] args=new String[2];
args[0]="process";//type of the element (see method getProperties(String[] args))
args[1]=tfe.getCompName();//name of the process (see method getProperties(String[] args))
Vector vProp=proj.getProperties(args);
Double f=(Double)vProp.elementAt(3);// flow value, automatically propagated from the upstream process
double flow=f.doubleValue();
String amont=(String)vProp.elementAt(1);//gets the upstream point name
getPointProperties(amont);//direct parsing of point property vector
Tamont=Tpoint;// here T1
Pamont=Ppoint;// here P1 and therefore Pabs
double Hamont=Hpoint;// here h1
```

To access the downstream point, the approach is quite similar:

2) reading of T_{abs}, T_{gen} and ε in the external component screen

The extThopt package provides a number of simple but robust methods to convert to double the String displayed in the JTextField used in the GUI, and vice versa to show double in these fields. They are implemented as static methods of the class extThopt.Util:

```
epsiSol=Util.lit_d(epsiSol_value.getText());
Tabs=Util.lit_d(Tabs_value.getText())+273.15;
Tgen=Util.lit_d(Tgen_value.getText())+273.15;
```

3) inversion of $P_{abs} = P(x_{sr},T_{abs})$ to get x_{sr} then h_{srD}

The first step is to define the functions for calculating the properties of the pair LiBr-H2O. As noted above, we chose the equations proposed by ASHRAE [2], expressed with respect to X LiBr mass fraction between 0 and 100. Two functions are provided to calculate the pressure and the enthalpy of the mixture at equilibrium. They are implemented in methods getP(double x,double T) and getH(double x,double T).

To reverse the functions, the class extThopt.Util provides a dichotomy method called dicho_T, which receives as arguments:

- The calling class, which must implement the interface Inversable
- The value to reach
- A possible parameter
- a String characterizing the function in method f_dicho() defining the functions to be inversed
- The minimum and maximum values of the search interval
- The required accuracy

In f_dicho(), we define the function to reverse by a string, here "P_LiBr"

```
if (fonc.equals("P_LiBr"))return getP(x,T);
```

With this signature inversion is obtained by:

```
xsr=Util.dicho_T (this,Ppoint,Tabs, "P_LiBr",0.3, 0.55,0.00001);
```

which is interpreted as: reverse the function defined in f_dicho() by "P_LiBr", the target value being Ppoint, the parameter being Tabs, the search interval being [0.3 to 0.55] and the precision required 10^{-5}.

The interval was chosen to match the validity of the equations of ASHRAE, given for X = 100 (1 - x) between 45% and 70%.

Steps 4-10

These steps do not pose any particular problem. Note that the code provides for the printing of intermediate results, useful mainly in the phase of development.

11) update and calculation of associated thermocouplers

Since thermocouplers are types of heat exchangers, it is interesting to characterize them by values such as effectiveness ε, the UA, the NTU or LMTD, which can be calculated from similar equations.

For this, we need the external component to transmit to each of its thermocoupler values of equivalent flow rates, inlet and outlet temperatures and thermal energy transferred they must take into account in their calculations. Specific methods were placed in the interface for this purpose.

We should however be aware that the analogy with heat exchangers has some limitations: for example, temperature crossings unacceptable in a heat exchanger may occur in a thermocoupler, leading to absurd calculations if we are not careful.

We will therefore often prefer passing on values that are not likely to lead to this situation: a solution can be to consider for the calculation that the thermocoupler is isotherm, such as in the model used here for the absorber and the desorber.

The method updateThermoCoupler () of extProcess allows us to perform this update:

```
updateThermoCoupler("absorber", Tabs, Tabs, Qabs, msr);
updateThermoCoupler("desorber", Tgen, Tgen, Qgen, msp);
```

For example, here we have assumed that the absorber is at temperature Tabs, receives the heat load Qabs and is crossed by the flow msr. If we had not taken the temperature of the absorber as a reference for the exchange calculation, keeping those of water vapor entering and leaving the external process, this would have resulted in an unacceptable temperature crossing .

Saving and loading the model parameters

You can save the settings of external components in the regular Thermoptim project files (and then re-read them) using two specific methods:

```
public String saveCompParameters ()
public void readCompParameters (String ligne_data)
```

The only constraint is that the entire setting of external components must fit on one line, with a format compatible with that used in the core of the software: the different backup fields are separated by tabs.

ExtThopt.Util provides a generic method to attribute to the value of a parameter a code allowing one to identify it:

```
public static String extr_value (ligne_data String, String search)
```

The backup is done in the form "parameter = value" and the search is in the form:

```
value = Util.lit_d (Util.extr_value (ligne_data, "parameter"));
```

If the setting of the component is too complex to be saved in this way, nothing prevents a user to to save the name of a specific file setup, and then re-read as he wishes this file.

4) Variants

It might be interesting to extend the model of the single-effect absorption cycle to represent a double- effect cycle, more efficient.

5) Work files

Project and diagram files of Thermoptim models are attached in the archive Trigeneration_en.zip, which also contains the files of the external class LiBrAbsorption.