Mise à jour et calcul de la transfo

On se réfère ici à l'exemple de la classe SolarCollector, dont le modèle est présenté dans la note "SolarCollector.doc".

L'enchaînement des opérations est le suivant :

1) mise à jour du composant avant calcul par chargement des valeurs de la transfo et du point amont
2) lecture des paramètres sur l'écran du composant externe
3) calcul de la puissance mise en jeu et de l'état du point aval
4) calcul des charges thermiques des thermocoupleurs
5) mise à jour de l'écran du composant externe

Examinons maintenant les problèmes rencontrés en pratique à chacune de ces étapes. Quelques extraits du code sont donnés ci-dessous, mais il est recommandé de lire la suite de cette note en ayant sous les yeux l'intégralité de la classe SolarCollector.java.

1) mise à jour du composant par chargement des valeurs de la transfo et du point amont

La petite difficulté est ici qu'un composant externe n'a pas un accès direct aux variables du simulateur : ces grandeurs sont obtenues par des méthodes particulières génériques, qui construisent des Vector de structure différente selon l'objet désiré.

La marche à suivre n'est pas compliquée mais demande à être respectée :

```
String[] args=new String[2]; //tableau des arguments
args[0]="process";//type d'élément désiré (ici une transfo)
args[1]=tfe.getCompName();//nom de la transfo (obtenu par la référence tfe)
Vector vProp=proj.getProperties(args);//méthode de Projet donnée en annexe 2 du tome 3
Double f=(Double)vProp.elementAt(3);
double flow=f.doubleValue();//valeur du débit, propagé automatiquement depuis la transfo amont
String amont=(String)vProp.elementAt(1);//nom du point amont
getPointProperties(amont);//décodage automatique du Vector (méthode de ExtProcess)
Tamont=Tpoint;//ici T1
```

La méthode getPointProperties() d'ExtProcess charge automatiquement l'état d'un point dans des valeurs aisément manipulables, appelées Tpoint, Ppoint, lecorps… Elle est donnée ci-dessous

```
public void getPointProperties(String nom){
        String[] args=new String[2];
        args[0]="point";//type of the element (see method getProperties(String[] args))
        args[1]=nom;//name of the process (see method getProperties(String[] args))
        Vector vProp=proj.getProperties(args);
  lecorps=(Corps)vProp.elementAt(0);
  nomCorps=(String)vProp.elementAt(1);
  Double y=(Double)vProp.elementAt(2);
  Tpoint=y.doubleValue();
  y=(Double)vProp.elementAt(3);
        Ppoint=y.doubleValue();
  y=(Double)vProp.elementAt(4);
        Xpoint=y.doubleValue();
  y=(Double)vProp.elementAt(5);
        Vpoint=y.doubleValue();
  y=(Double)vProp.elementAt(6);
        Upoint=y.doubleValue();
  y=(Double)vProp.elementAt(7);
        Hpoint=y.doubleValue();
  y=(Double)vProp.elementAt(9);
        DTsatpoint=y.doubleValue();
```

```
        String dum=(String)vProp.elementAt(8);
        isTsatSet=Util.lit_b(Util.extr_value(dum));
        dum=(String)vProp.elementAt(10);
        isPsatSet=Util.lit_b(Util.extr_value(dum));
}
```

2) lecture des paramètres sur l'écran du composant externe

Le package extThopt fournit un certain nombre de méthodes simples mais robustes pour convertir en double les String affichés dans les champs JTextField utilisés dans l'interface graphique, et réciproquement pour afficher les double dans ces champs. Ils sont implémentés sous forme de méthodes statiques de la classe extThopt.Util (cf. annexe 3) :

```
P=Util.lit_d(P_value.getText());
A=Util.lit_d(A_value.getText());
tau=Util.lit_d(tau_value.getText());
K=Util.lit_d(K_value.getText());
Tex=Util.lit_d(Tex_value.getText())+273.15;
```

3) calcul de la puissance mise en jeu et de l'état du point aval

On commence par estimer le Cp du fluide caloporteur en faisant un développement limité de la fonction enthalpie, ce qui amène à utiliser la méthode CalcPropCorps() du package Corps, et la méthode getSubstProperties() d'ExtProcess, qui charge automatiquement l'état d'un point dans des valeurs aisément manipulables, appelées Tsubst, Psubst, etc. :

```
public void getSubstProperties(String nom){
        String[] args=new String[2];
        args[0]="subst";//type of the element (see method getProperties(String[] args))
        args[1]=nom;//name of the process (see method getProperties(String[] args))
        Vector vProp=proj.getProperties(args);
   Double y=(Double)vProp.elementAt(0);
   Tsubst=y.doubleValue();//température
   y=(Double)vProp.elementAt(1);
        Psubst=y.doubleValue();//pression
   y=(Double)vProp.elementAt(2);
        Xsubst=y.doubleValue();//titre
   y=(Double)vProp.elementAt(3);
        Vsubst=y.doubleValue();//volume massique
   y=(Double)vProp.elementAt(4);
        Usubst=y.doubleValue();//énergie interne massique
   y=(Double)vProp.elementAt(5);
        Hsubst=y.doubleValue();//enthalpie massique
   y=(Double)vProp.elementAt(6);
        Ssubst=y.doubleValue();entropie massique
y=(Double)vProp.elementAt(7);
     Msubst=y.doubleValue();//masse molaire
     Integer i=(Integer)vProp.elementAt(8);
     typeSubst=i.intValue();//type de corps (1 pour l'eau, 2 pour une vapeur, 3 pour un gaz pur, 4 pour un gaz composé, 5 pour un corps externe)
 }
```

Ceci permet de calculer le Cp comme suit :

```
double H=Hpoint;//enthalpie du point amont
lecorps.CalcPropCorps(Tpoint+1, Ppoint, Xpoint);// recalcul du corps amont par une fonction de Thermoptim
getSubstProperties(nomCorps);//récupération des valeurs du recalcul (méthode de ExtSubstance)
double Cp=(Hsubst-H);//valeur estimée de Cp
```

On calcule alors une valeur estimée de Taval pour pouvoir déterminer la puissance thermique absorbée Qex :

```
double DT0=tau*P/K-Tamont+Tex;
double T=Tamont+(DT0)*(1-Math.exp(-K*A/flow/Cp));
double DT=T-Tpoint;
double Qex=Cp*DT*flow;
```

On détermine la valeur de l'enthalpie massique du point aval, puis on inverse cette fonction pour déterminer la valeur exacte de Taval (méthode de Corps)

```
double hAval=Qex/flow+Hpoint;
Tpoint=lecorps.getT_from_hP(hAval,Ppoint);

getSubstProperties(nomCorps);//récupération des valeurs du recalcul (méthode de ExtProcess)
Xpoint=Xsubst;//mise à jour de la valeur du titre pour le cas où l'état est diphasique
```

4) calcul des charges thermiques des thermocoupleurs

Dans cet exemple simple, le problème ne se pose pas, le composant ne mettant pas en jeu de thermocoupleur. Quelques indications sont données plus loin, ainsi que dans la section relative aux nœuds externes.

5) mise à jour de l'écran du composant externe

La méthode de Thermoptim setupPointAval() permet de mettre à jour le point aval à partir des valeurs chargées dans un Vector construit ici par la méthode getProperties() de ExtProcess :

```
tfe.setupPointAval(getProperties());
```

La valeur du rendement du capteur est ensuite déterminée et affichée.

```
eff_value.setText(Util.aff_d(Qex/P/A, 4));
```

Pour cet exemple, les mises à jour avant et après recalcul sont très simples. Dans d'autres cas, il peut être nécessaire d'accéder à d'autres données du simulateur.