XV COBEQ
A Engenharia Química e o Crescimento Sustentável
II Congresso Brasileiro de Termodinâmica Aplicada - CBTERMO
26 a 29 de setembro de 2004

AVANÇOS NO DESENVOLVIMENTO DO SIMULADOR DE PROCESSOS EMSO

R. de P. Soares, A. R. Secchi

Grupo de Integração, Modelagem, Simulação, Controle e Otimização de Processos (GIMSCOP)
Departamento de Engenharia Química - Universidade Federal do Rio Grande do Sul
Rua Prof. Luiz Englert, s/n – CEP: 90040-040 – Porto Alegre – RS – Brasil
Telefone: (51)3316-4165 – Fax: (51)3316-3528 – Email: {rafael, arge}@enq.ufrgs.br

RESUMO – Neste trabalho alguns avanços no desenvolvimento da ferramenta chamada EMSO (*Environment for Modeling, Simulation and Optimization*) são apresentados. Esta é uma ferramenta baseada em equações para modelagem, simulação e otimização de processos estáticos e dinâmicos, que provê uma linguagem própria orientada a objetos, análises automáticas entre outras características. Entretanto, já existem diversas soluções desenvolvidas em outros simuladores ou códigos independentes. Vislumbrando estes casos, interfaces CAPE-OPEN foram implementadas no EMSO juntamente com um novo sistema de interface que possibilita a utilização direta de software externo ao simulador dentro dos modelos. Utilizando estas interfaces, diagramas de processos complexos como plantas termoelétricas e processos de separação requerendo cálculos termodinâmicos e de propriedades físicas foram simulados com sucesso.

PALAVRAS-CHAVE: Simulação de Processos, Simulação Dinâmica, Sistemas de Índice Elevado.

ABSTRACT – In this work, some recent advances of the general purpose equation-based simulation tool named EMSO are presented. This tool provides an object-oriented modeling language and can perform static and dynamic simulations solvability analysis, and many other interesting tasks. However, there are several solutions developed within another simulators or independent codes. In order to reuse these solutions CAPE-OPEN interfaces were developed besides a new interface which makes possible to load at run-time external software within models. Using this method, complex flowsheet diagrams as thermoelectrical power plant and complex separation processes were successfully solved.

1.INTRODUÇÃO

Simulador é uma ferramenta valiosa, pois pode ser utilizada em uma gama de aplicações: controle, operação, aumento de produção, redução de custos, entre outros. Estes são alguns dos motivos do aumento do interesse industrial em ferramentas de simulação, mas estas ainda são consideradas inadequadas pelos usuários (Che-Comp, 2002). Esta insatisfação está intimamente ligada à falta de flexibilidade, dificuldade de utilização e/ou no aprendizado e alto custo. Outro fato relevante é a precariedade na compatibilidade de soluções desenvolvidas entre as diversas ferramentas de simulação. Além disto, os usuários têm apontado algumas características desejadas, tais como: comportamentos padrões e interfaces inteligentes (Hlupic, 1999).

Neste contexto, a ferramenta chamada EMSO (*Environment for Modeling, Simulation, and Optimization*) é introduzida (Soares e Secchi, 2003). Esta ferramenta visa fornecer uma maior flexibilidade ao usuário em um sistema atualizado que provê, entre outras, uma linguagem orientada a objetos, diferenciação automática e simbólica. A Figura 1 apresenta uma visão geral da arquitetura desta ferramenta.

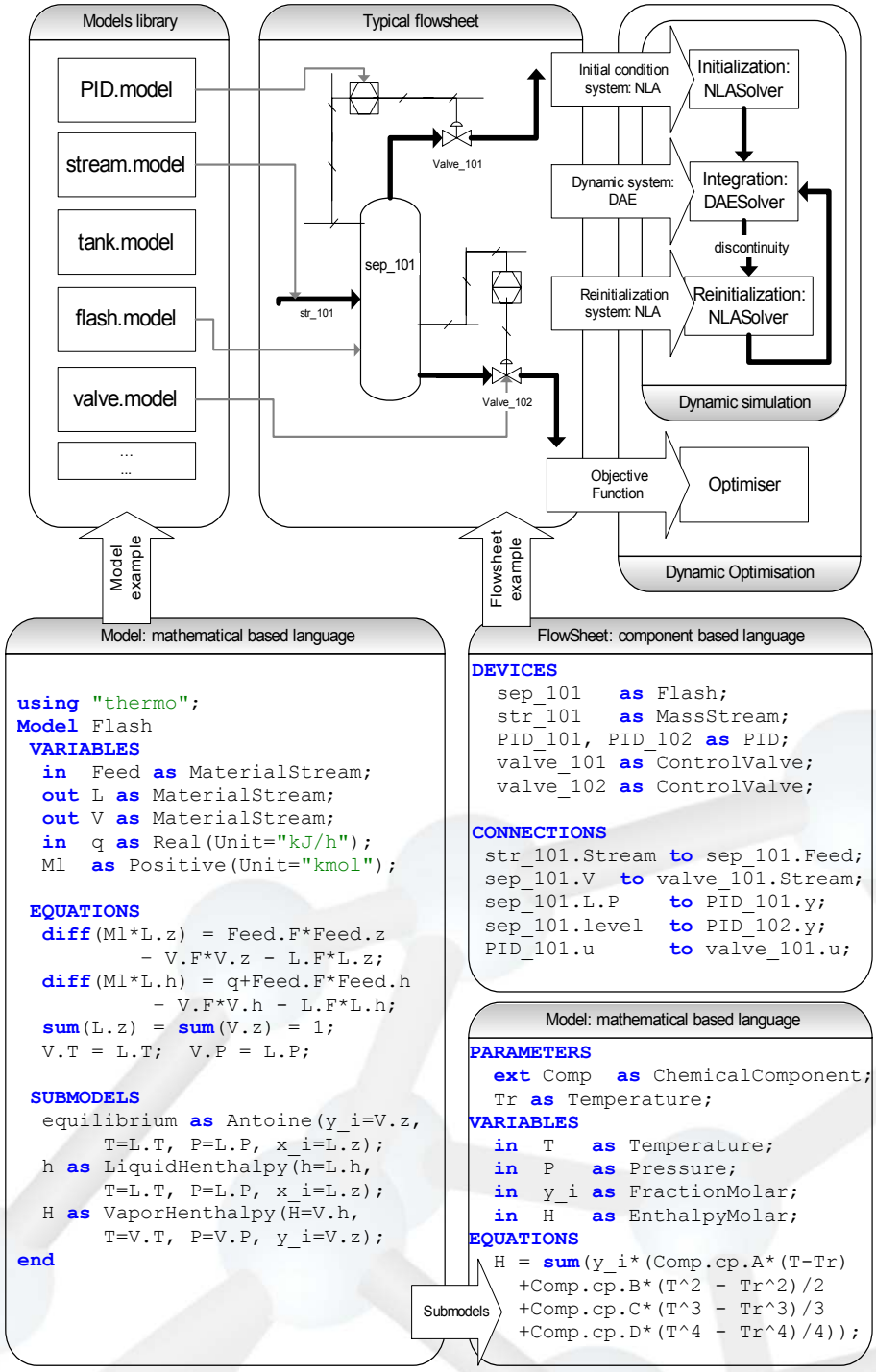

Figura 1 – Visão geral do simulador de processos EMSO.

2.MODELAGEM

No EMSO a descrição de um processo (chamado de *FlowSheet*) pode ser obtida pela simples conexão de unidades moduláres básicas chamadas de *Devices*. Cada *Device* é criado com base em uma descrição matemática, um *Model*. O usuário pode optar por utilizar os modelos existentes para a criação de seus *Devices*, derivar novos modelos a partir dos existentes ou criar seus próprios modelos. Assim, as três principais entidades do sistema, ordenadas em nível de abstração, são: *FlowSheet*, *Device* e *Model*.

Na parte inferior da Figura 1 podem ser observados trechos de código da linguagem de modelagem do EMSO.

2.1FlowSheet

Um *FlowSheet* é a principal entidade do EMSO pois representa o fluxograma processo a ser resolvido. Sobre esta são executadas as tarefas de simulação. Para a descrição de um *FlowSheet* é necessária apenas a listagem dos equipamentos que o compõem (seus *Devices*) e as conexões entre estes.

Na parte inferior da Figura 1 pode ser observado um exemplo de *Flowsheet*. Como pode ser visto, a descrição de um *FlowSheet* se dá na forma textual, porém esta descrição é simples o suficiente para ser totalmente gerenciada por uma interface gráfica. Nesta interface um *FlowSheet* poderia ser construído pela seleção de seus componentes e conexão entre estes com auxílio de um dispositivo apontador (*mouse*).

2.2Model

Como citado anteriormente cada *Device* de um *FlowSheet* depende de um modelo matemático, chamado *Model*. Na linguagem do EMSO um *Model* consiste na abstração matemática de algum equipamento, trecho de processo ou até mesmo *software*. Exemplos típicos de *Model* são a descrição matemática de um tanque, reator ou controlador PID.

Um *Model* pode conter parâmetros, variáveis, equações, condições iniciais ou modelos internos (composição). Novos modelos podem ser construídos pela derivação de outros existentes (herança) e novas funcionalidades podem ser adicionadas (novos parâmetros, variáveis, equações, etc.).

Cada variável ou parâmetro de um modelo é baseado em um *tipo* predefinido. Cada *tipo* contém uma série de atributos, tais como limite inferior, superior, uma descrição breve, entro outros. Além disto novos *tipos* podem ser derivados dos tipos básicos, exemplos da declaração e utilização de *tipos* podem ser vistos na Figura 2.

```
Fraction as Real(Lower=0, Upper=1);
Positive as Real(Lower=0, Upper=inf);
EnergyHoldup as Positive(Unit="J");
```

Figura 2 – Derivação de *tipos*.

3.ANÁLISES DE CONSISTÊNCIA

Em sistemas baseados em equações, a solução de um processo é feita pela concatenação de todas as variáveis e equações de todos os equipamentos do processo em um único sistemas de equações. Uma vez obtido este sistema resultante a solução pode ser obtida utilizando-se métodos apropriados.

Entretanto, a aplicação de análises de resolubilidade antes de uma tentativa de solução numérica pode revelar grande parte das causas de falha na solução. Existem diversos tipos de análises que podem ser aplicadas aos sistemas de equações algébrico-diferenciais (DAE - *Differential Algebraic Equations*) e sistemas de equações não lineares (NLA - *Nonlinear Algebraic*) que surgem na solução de problemas dinâmicos e estáticos, respectivamente. Exemplos destas possíveis análises que estão presentes no EMSO são: consistência de condições iniciais, consistência estrutural e consistência de unidades de medida.

4.INTERFACES

A habilidade de carregar *software* externo em tempo de execução é implementada por diversos simuladores comerciais. Esta característica é normalmente implementada utilizando-se algum sistema próprio de interface com outros *softwares*, o que leva a sistemas heterogêneos e incompatíveis.

4.1Interfaces CAPE-OPEN

Recentemente, o projeto CAPE-OPEN (CO-LAN, 2002) publicou um conjunto de interfaces padrões objetivando uma diminuição da heterogeneidade das interfaces dos simuladores de processos. O EMSO implementa um conjunto de interfaces que permite o compartilhamento dos sistemas de equações que resultam de seus modelos e também de seus códigos para solução de problemas dinâmicos e estáticos.

4.2Interfaces Próprias

Sem dúvida o sistema de interfaces CAPE-OPEN traz um grande avanço no sentido de compatibilidade entre simuladores de processos. Porém, o trabalho de Soares e Secchi (2004) mostrou que a utilização de interfaces CAPE-OPEN leva a perdas de eficiência. Esta perda se dá principalmente devido à grande complexidade envolvida na proposta de um sistema que trabalha de forma transparente em redes e/ou sistemas operacionais heterogêneos.

Por este motivo, além das interfaces CAPE-OPEN o EMSO apresenta um sistema próprio que permite carregar, em tempo de execução, códigos de terceiros encapsulados em bibliotecas dinâmicas de carregamento dinâmico (DLL em sistemas Win32 ou SO em sistemas Posix).

5.INTERFACE GRÁFICA

A interface gráfica do EMSO combina a construção de *FlowSeets*, o desenvolvimento de *Models* e a visualização de resultados. O cerne do simulador, assim como a sua interface gráfica são totalmente desenvolvidos em C++ e projetados de forma modular e padrão, tornado o simulador compatível com plataformas Win32, Unix e Linux.

Quando da execução de tarefas de simulação o EMSO converte a descrição do processo em sua linguagem de modelagem para sistemas de equações diretamente em memória, sem a geração de arquivos intermediários, necessidade de compilação ou *link* edição. O *software* permite execução de tarefas em paralelo permitindo simulações em tempo real, com atraso ou simulações concorrentes sem o bloqueio da interface. Além disto, o processo de obtenção das soluções pode ser pausado ou cancelado a qualquer tempo. A Figura 3 apresenta a interface gráfica do EMSO.

6.APLICAÇÕES

Nesta seção algumas aplicações apresentando a utilização de interfaces do simulador EMSO são apresentadas.

6.1Composição de modelos utilizando interfaces CAPE-OPEN

O principal objetivo do projeto CAPE-OPEN é tornar possível que componentes nativos de um simulador possam ser substituídos ou conectados a outros de uma fonte independente. A aplicação apresentada nesta seção faz a composição do modelo de um processo pela conexão de dois sub-processos de fontes diferentes.

Considere-se o processo da Figura 4, separado em dois sub-processos A e B. Um simulador CAPE-OPEN deve ser capaz de coletar e utilizar cada uma destas partes do processo mesmo que venham de fontes externas heterogêneas. Esta situação foi resolvida com sucesso em um experimento utilizando duas instancias do simulador EMSO. Uma descrição mais detalhada deste experimento pode ser encontrada em Soares e Secchi (2004).

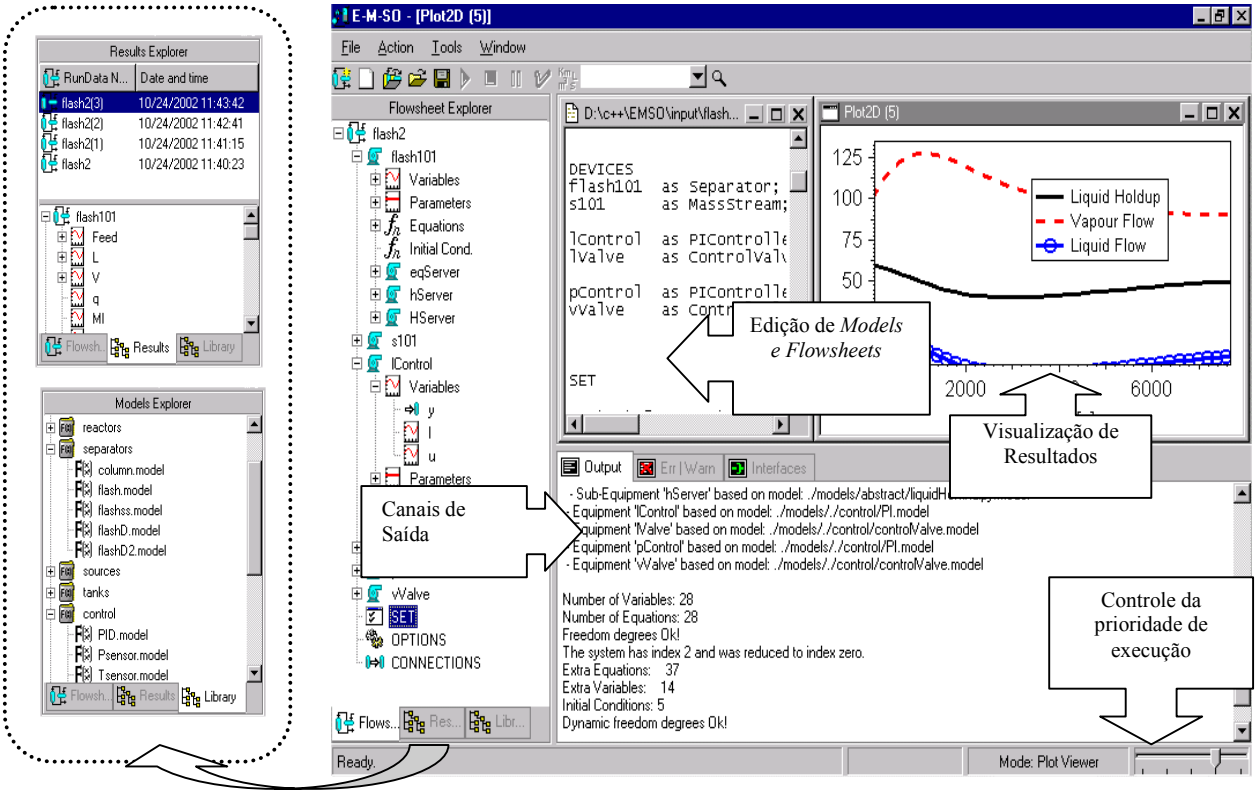

Figura 3 – Interface gráfica do simulador EMSO.

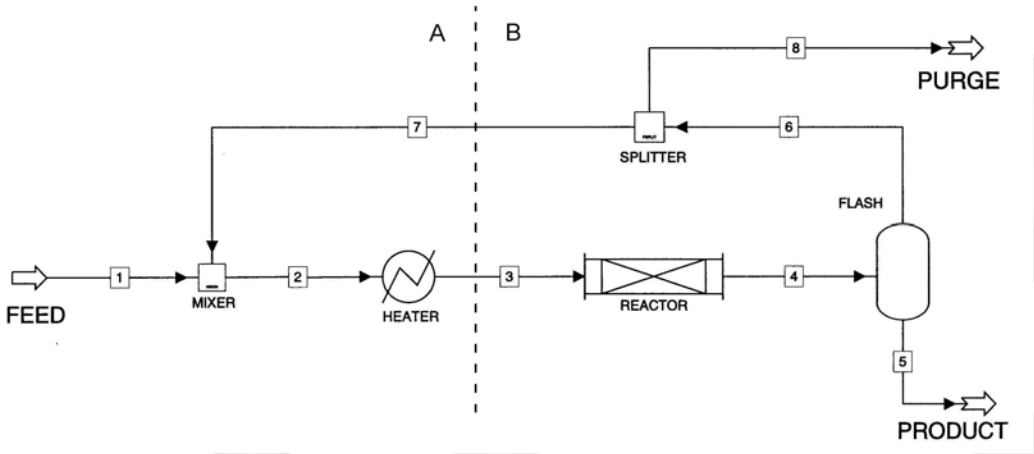

Figura 4 – Processo dividido em dois para a solução por interface CAPE-OPEN.

6.2 Bibliotecas de Propriedades Termodinâmicas

Como citado anteriormente, além das interfaces CAPE-OPEN o EMSO provê um mecanismo que permite carregar, em tempo de execução, bibliotecas dinâmicas (DLL ou SO) provendo cálculos de interesse. Exemplos típicos de aplicação seriam cálculos de propriedades termodinâmicas ou CFD (*Computational Fluid Dynamics*).

Com o objetivo de testar esta capacidade considere-se o ciclo de Rankine apresentado na Figura 5.

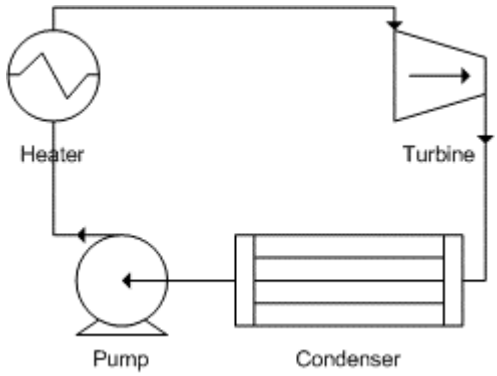

Figura 5 – Processo do ciclo de Rankine.

O *FlowSheet* para descrição deste processo no EMSO é apresentado na Figura 6.

```
FlowSheet Rankine
  PARAMETERS
    Prop as CalcObject(File="thermo.dll");
  DEVICES
    Turb       as Turbine;
    Cond       as Condenser;
    Pump       as Pump;
    GV         as Boiler;
    GE         as Electric_Power;
  CONNECTIONS
    GV.Fout       to  Turb.Fin;
    Turb.Fout     to  Cond.Fin;
    Cond.Fout     to  Bump.Fin;
    Bump.Fout     to  GV.Fin;
end
```

Figura 6 – *FlowSheet* do ciclo de Rankine.

No *FlowSheet* da Figura 6, o arquivo `thermo.dll` é utilizado como fonte de cálculos termodinâmicos. Esta biblioteca foi construída com rotinas escritas em FORTRAN-90, mas diversas outras linguagens podem ser utilizadas para implementação de serviços semelhantes. Detalhes sobre os *Models* utilizados como base para os *Devices* do *FlowSheet* da Figura 6 podem ser encontrados no Apêndice A.

7.CONCLUSÕES

Os principais avanços do ambiente integrado para modelagem, simulação e otimização de processos estáticos e dinâmicos, chamado EMSO foram apresentados.

Esta ferramenta implementa uma linguagem de modelagem orientada a objetos juntamente com métodos para checagem de consistência, redução de índice e derivação automática e simbólica. O pacote numérico de interfaces CAPE-OPEN já está implementado juntamente com um sistema próprio que permite a utilização de códigos externos dentro dos modelos. Isto permite, por exemplo, que *softwares* consagrados para cálculos de propriedades termodinâmicas sejam utilizados.

Códigos de otimização dinâmica estão sendo estudados para serem incorporados, espera-se que a arquitetura modular do simulador permita esta adição sem a necessidade de refazer trabalhos executados anteriormente.

8.BIBLIOGRAFIA

CHE-COMP, The State of Chemical Engineering Softwares. www.che-comp.org, 2002.

CO-LAN, Conceptual Design Document for CAPE-OPEN Project. www.co-lan.org, 2002.

HLUPIC, V., Simulation Software: User´s requirements. *Comp. Ind. Engrg,* 37, 185-188, 1999.

SOARES, R. de P and SECCHI, A. R., EMSO: a New Tool for Modeling, Simulation and Optimization, ESCAPE-13, 2003.

SOARES, R. de P and SECCHI, A. R., Modifications, Simplifications, and Efficiency Tests for the CAPE-OPEN Numerical Open Interfaces. *Comp. Chem. Engrg*, (in press), 2004.

APÊNDICE A

Seguem abaixo os *Models* utilizados no *FlowSheet* da Figura 6.

```
# Type declarations
Enthalpy       as      Real(Default=2,Lower=1e-3,Upper=10,Unit="MJ/kg");
Entropy        as      Real(Default=5,Lower=1e-3,Upper=15,Unit="kJ/kg/K");
Power          as      Real(Default=10,Lower=0,Upper=1e3,Unit="MW");
Pressure       as      Real(Default=1,Lower=5e-4,Upper=100,Unit="MPa");
Temperature    as      Real(Default=600,Lower=273.16,Upper=1073.15,Unit="K");
Diff_Temp      as      Real(Default=0,Lower=-1073.15,Upper=1073.15,Unit="K");
MassFlow       as      Real(Default=50,Lower=0,Upper=1e3,Unit="kg/s");
SpecificVolume as      Real(Default=1,Lower=1e-6,Upper=1e3,Unit="m^3/kg");
Fraction       as      Real(Default=0.5,Lower=0,Upper=1);
Efficiency     as      Real(Default=0.75,Lower=0,Upper=1);

#* The model Stream has no equations. Its only function is to hold the data
* to be shared between the devices of the flowsheet (an stream).
*#
Model Stream
 VARIABLES
        F as MassFlow;
        P as Pressure;
        T as Temperature;
        S as Entropy;
        H as Enthalpy;
end

#* Model of a Turbine.
* This model has an external parameter Prop which provides the
* thermodynamic calculations.
*#
Model Turbine
 PARAMETERS
  ext Prop      as CalcObject;
 VARIABLES
  in  Fin       as Stream;
  out Fout      as Stream;
      H_IS      as Enthalpy;
      EF_T      as Efficiency;
      POT_TURB  as Power;
EQUATIONS
      H_IS = .Prop.propPS(Fout.P,Fin.S);
      Fout.H = (H_IS - Fin.H) * EF_T + Fin.H;
      [Fout.S,Fout.T] = Prop.propPH(Fout.P,Fout.H);
      Fout.F * (Fin.H - Fout.H) = POT_TURB;
      Fin.F = Fout.F;
end

#* Model of a Condenser.
* This model has an external parameter Prop which provides the
* thermodynamic calculations.
*#
Model Condenser
 PARAMETERS
  ext Prop      as CalcObject;
 VARIABLES
  in  Fin       as Stream;
  out Fout      as Stream;
      Q_COND    as Power;
      G_S       as Diff_Temp;
 EQUATIONS
      Fout.P = Fin.P;
      Fout.T = Prop.Tsat(Fout.P) - G_S;
      [Fout.S,Fout.H] = Prop.propPTl(Fout.P,Fout.T);
      Q_COND = Fin.F * (Fin.H - Fout.H);
end
```

```
#* Model of a Pump.
* This model has an external parameter Prop which provides the
* thermodynamic calculations.
*#
Model Pump
 PARAMETERS
  ext Prop     as CalcObject;
      v_esp    as SpecificVolume;
 VARIABLES
  in  Fin      as Stream;
  out Fout     as Stream;
      H_IS     as Enthalpy;
      POT_BMB  as Power;
      EF_B     as Efficiency;
 EQUATIONS
      H_IS = Prop.propPS(Fout.P,Fin.S);
      (Fout.H - Fin.H) * EF_B = H_IS - Fin.H;
      [Fout.S,Fout.T] = Prop.propPH(Fout.P,Fout.H);
      POT_BMB * EF_B = Fin.F * v_esp * (Fout.P - Fin.P);
      Fin.F = Fout.F;
end

#* Model of a Boiler.
* This model has an external parameter Prop which provides the
* thermodynamic calculations.
*#
Model Boiler
 PARAMETERS
  ext Prop     as CalcObject;
 VARIABLES
  in  Fin      as Stream;
  out Fout     as Stream;
      Q_GV     as Power;
      EF_GV    as Efficiency;
 EQUATIONS
      Fin.P = Fout.P;
      [Fout.S,Fout.H] = Prop.propPTv(Fout.P,Fout.T);
      Q_GV * EF_GV = Fin.F * (Fout.H - Fin.H);
      Fin.F = Fout.F;
end

#* Model of an Electric_Power.
* This model has an external parameter Prop which provides the
* thermodynamic calculations.
*#
Model Electric_Power
 PARAMETERS
  ext Prop     as CalcObject;
      EF_GE    as Efficiency;

 VARIABLES
      Pot      as Power;
end
```