ThermoFlow

Thermo-hydraulic Modelling and Development of a Modelica Standard Library

by

Jonas Eborn

Falko Wagner

Hubertus Tummescheit

LUND INSTITUTE OF TECHNOLOGY
Lund University

DTU

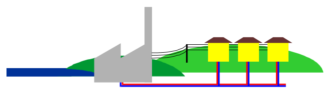
Contents

Introduction

- Thermodynamics
  - large field, “mother of science”
  - well-founded theory (first & second law)
  - empirical knowledge for materials & machines
  - expert: years of experience
  - model libraries can aide the non-expert with system simulation

Introduction - Applications

- Power (thermal, nuclear, solar ...)
- Refrigeration
- Chemical processes
- Building heating & ventilation
- ... many more ...



Introduction - Software

- Specialized thermo-hydraulic software
  - SINDA/FLUINT
- Static design tools
  - ProSim (power plants etc.)
  - Aspen Tech. (chemical process design)
- Computational Fluid Dynamics, CFD
  - FEMLAB, Combustion & 3D flow models
- General simulation packages
  - Dymola

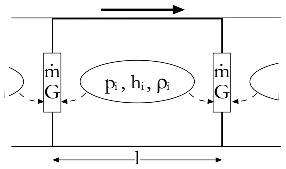
Contents

Physical Modelling - Features

- State of pure fluid described by 2 variables
  - p, ρ, v, T, u, h and s are possible
  - choice affects numerical properties
  - impossible combinations, e.g. $\{p,T\}$
- Balance eqns in mass, energy and momentum
- Constitutive relations
- Medium property models, functions of state

Physical modelling - Balances

Euler view: Integrate partial diffs. over a finite control volume



- Mass balance
  $$\frac{d\rho V}{dt} = \frac{dm}{dt} = \sum_i \dot{m}_i$$
- Energy balance
  $$\frac{dE}{dt} = \dot{Q} + \dot{W}_s + \sum_i \left(\dot{m}_i h_i + \frac{v_i^2}{2} + g z_i \right)$$
- Momentum balance
  $$V \frac{d\rho v}{dt} = l \frac{d\dot{m}}{dt} = \left(G_A - G_B \right) + A(\Delta p - \Delta p_f)$$

Physical modelling - State eqn

Rewrite basic balances into state eqns. for increased efficiency

- State equations in pressure and enthalpy:

$$\frac{dp}{dt} = \frac{\rho}{\alpha_p\rho + \alpha_h}\left(\left(\frac{\rho + \alpha_h h}{\rho V}\right)\Delta m - \frac{\alpha_h}{\rho V}\Delta e\right) \qquad \Delta m = \sum_i \dot{m}_i$$

$$\frac{dh}{dt} = \frac{\rho}{\alpha_p\rho + \alpha_h}\left(\left(\frac{\rho - \alpha_p h}{\rho V}\right)\Delta m - \frac{\alpha_p}{\rho V}\Delta e\right) \qquad \Delta e = \sum_i \dot{m}_i h_i + \sum_j \dot{q}_j$$

ρ, α_p, α_h and T are calculated by property functions.
Similar structure has been derived for (p, s) and (ρ, T).

Physical modelling - Empirics

Constitutive relations for different units are found by empirics,
Dimension free numbers used: *Re, Pr, Nu, Fr, ...*

- Pressure drop correlations

$$\Delta p_f = f(Re, l, d)\frac{l}{d}\frac{\dot{m}}{\rho A}$$

- Heat flow correlations

$$\dot{q} = h(Re, Pr, \dot{m})A\Delta T$$

- Unit specific characteristics

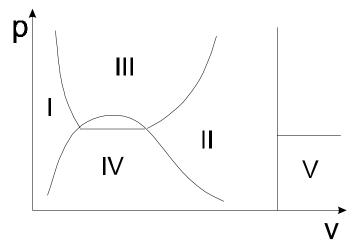
Physical modelling - Medium

- Medium properties are calculated from functions
  - Water (thermal power plants, steam heating)
  - Refrigerants (cooling in industry, ventilation...)
  - Gas mixtures (ideal or non-ideal, combustion)

Implemented as look-up tables or by approximation functions.
Iterative solutions ⇒ too slow for dynamic simulation.
Forward expressions in state give fast evaluation.

Physical modelling - Medium

- Steam properties standard: IAPWS-IF97
  - explicit functions in p, h
  - 5 different regions, switching



Contents

ThermoFlow - Library goals

- User-friendly, for basic/advanced users
- Cover wide area of interest
- Unified component library
  - lumped and discretized models
- Flexibility through parameterization

ThermoFlow - Applicability

- Many different media, water, gas, refrigerant
  - also multi-component mixtures
- Handle all cases of flows
  - uni- or bidirectional
  - one- or two-phase
  - homogeneous or inhomogeneous
  - with/without dynamic momentum balance
- Units: valves, heat exchangers, turbines, ...

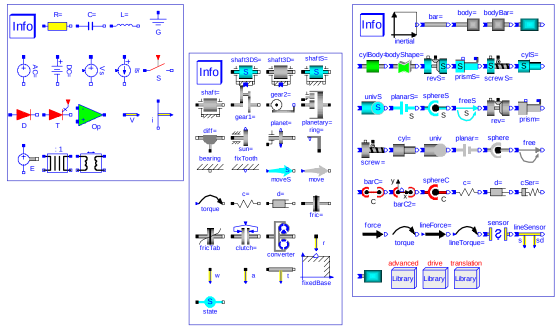
ThermoFlow - Flexibility

- Use parameters for choosing
  - level of discretization
  - medium type, class parameters
  - if dynamic or static flow is used
- Empty “shell” models allow the user to create new compatible models

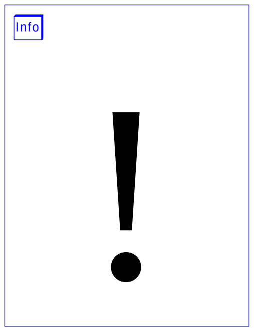
Contents

- Introduction
- Physical Modelling
- ThermoFlow model library
- Library design considerations
  - decomposition
  - library structure
- Modelica constructs
- Examples
- Conclusions

Existing Libraries



New Library



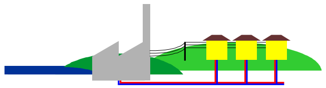
What do we want to model?

- Power plants
- Refrigeration systems

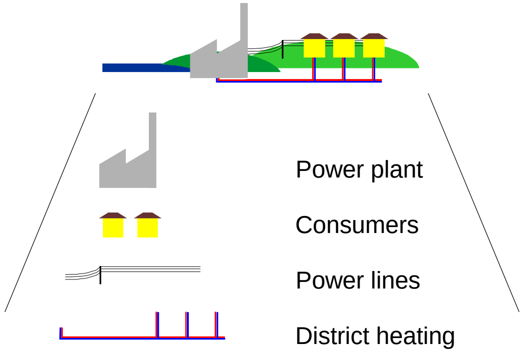
- Component examples :

  compressors, turbines, condensers, heat exchangers, boilers, flow in pipes, valves, etc.

Example : Power Plant

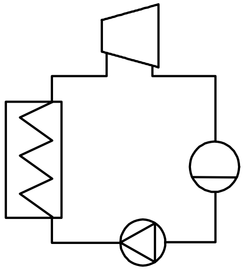


Hierarchical decomposition

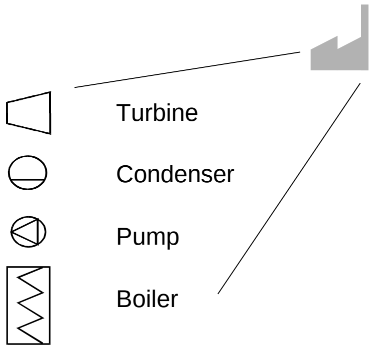


Component models

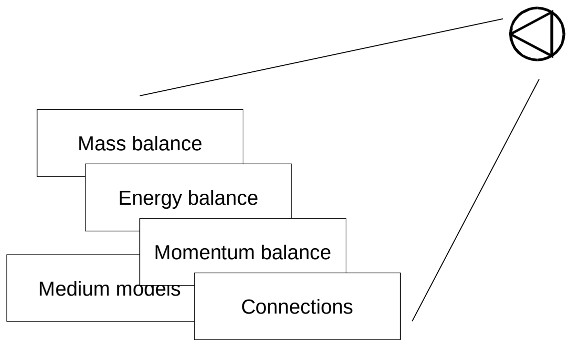
Power plant



The power plant



The pump



The momentum balance

Momentum balance

Modelica model :

```
G_norm[1]=mdot[1]^2/d[1]/A;
G_norm[2]=mdot[2]^2/d[2]/A;
dG[2]=G_norm[1]-G_norm[2];

dz*mdot[2]=dG+(p[2]-p[1])*A-Ploss;
```

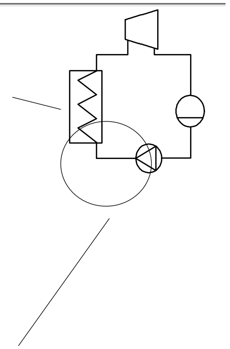

The connections

```
connector SingleFlow
    SIunits.Pressure            p;
    SIunits.SpecificEnthalpy    h;
    flow SIunits.MassFlowRate   mdot;
    flow SIunits.Power          q_conv;
    SIunits.Density             d;
end SingleFlow;

model pump
  SingleFlow in,out;

model boiler
  SingleFlow in,out;

connect(pump.out,boiler.in);
```



Important aspects

- Level of details
  - Purpose of library vs. modeling principles
- Nomenclature of research field
  - Use of known symbols/formulations
- Do not overkill !
  - Just because something seems to be general, it doesn't need its own class.

Key elements in library (1)

- Interfaces
  - all components have same interfaces (or same class of interfaces)
- Base components
  - build new components from basic
  - shell components (e.g. TwoPort)
- Advanced components

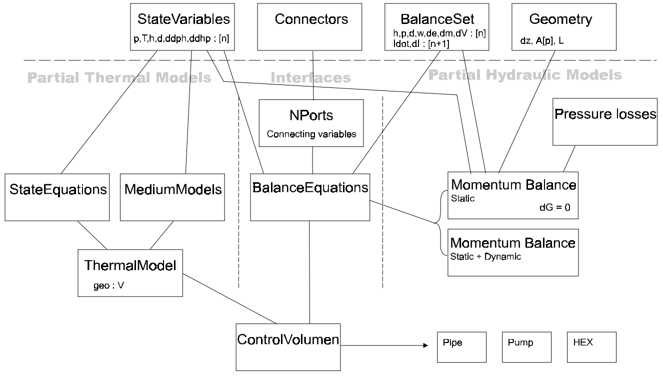
Key elements in library (2)

- Media property routines
  - fast! => for dynamic simulation
  - fast means explicit in state variables
  - fast means also pure Modelica functions
    - due to symbolic manipulation capabilities

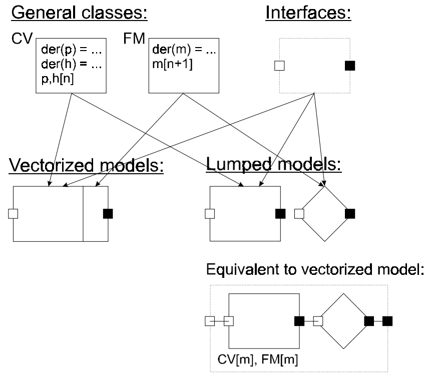
Library structure

- Classes :
  - Interfaces (connectors, variable sets)
  - n-Ports (contain connectors)
  - State equations
  - Media property routines
  - Thermal model (state equations + media)
  - Balance equations (mass + energy)
  - Hydraulic model (momentum + pressure)
  - Geometry data

Class diagram



Details



Interfaces

- Connectors contain:
  - p, h, mdot, q_conv, d, G_norm, dG
  - replaceable elements:
    - gamma, T
    - mdot for multiflow
- n-Ports
  - OnePort, TwoPort, ThreePort

Contents

- Introduction
- Physical Modelling
- ThermoFlow model library
- Library design considerations
- Modelica constructs
  - class parameterization
  - n-discretization
- Examples
- Conclusions

Class parameterization

- Used to implement alternate/exchangeable
  - Connectors
  - Media properties
  - Momentum balances
- Construct:

```
replaceable model Exchangeable=DefaultClass;
extends Exchangeable;
```

Example

- Class parameterization of connectors:

```
connector GExtension
    flow MomentumFlow G_norm;
    flow MomentumFlow dG;

connector GPExtension
    flow MomentumFlow G_norm;
    flow MomentumFlow dG;
    Real gamma;
    Temperature T;

connector SingleFlow
    Pressure         p;
    SpecificEnthalpy h;
    flow MassFlowRate mdot;
    flow Power        q_conv;
    Density           d;
    replaceable model ConnectorExtension = GExtension ;
    extends ConnectorExtension;

SingleFlow a(redeclare model ConnectorExtension= GPExtension);
```


Generic n-discretization

- Classes are discretized to contain n control volumes
- Possibility for using higher order derivatives
  - easier access to neighboring control volume

```
for i in 1:n loop
  dm[i]=mdot[i]-mdot[i+1];
  de[i]=edot[i]-edot[i+1]-p[i]*dV[i]+q_source[i];
end for;
```


Switching terms

- Switches used in momentum balances
  - big "IF" construction
- Generally done to support bi-directional flow

```
G_norm[i] = if mdot[i] > 0 then
              mdot[i]*mdot[i]/d[i-1]/A
            else
              mdot[i]*mdot[i]/d[i]/A;
```


Library implementation

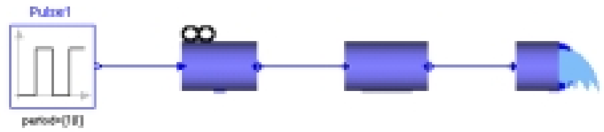
- Sub tree
  - `/library/Modelica/ThermoFlow`
  - results in "dot" notation : Modelica.ThermoFlow
- Top level contains
  - package.mo (description of library)
  - Interfaces.mo
  - PartialThermalModels.mo
  - PartialHydraulicModels.mo

Contents

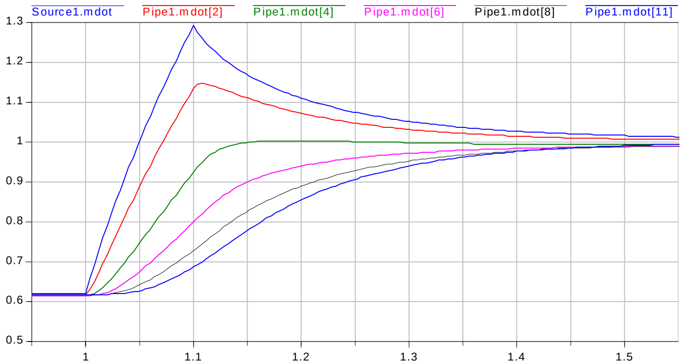
- Introduction
- Physical Modelling
- ThermoFlow model library
- Library design considerations
- Modelica constructs
- Examples
- Conclusions

Example

- Discretized pipe model :



Results



Contents

- Introduction
- Physical Modelling
- ThermoFlow model library
- Library design considerations
- Modelica_constructs
- Examples
- Conclusions

Conclusions

- Modelica supports library development
- Modelica constructs used to
  – increase flexibility
  – reduce number of component variants
  – choose complexity level
- Wide area of applicability with different medium models