Exergy Analysis of Thermo-Fluid Energy Conversion Systems in Model-Based Design Environment

Daniel Bender

German Aerospace Center (DLR)

Institute of System Dynamics and Control

The American Modelica Conference 2018

Boston, October 09th/10th, 2018

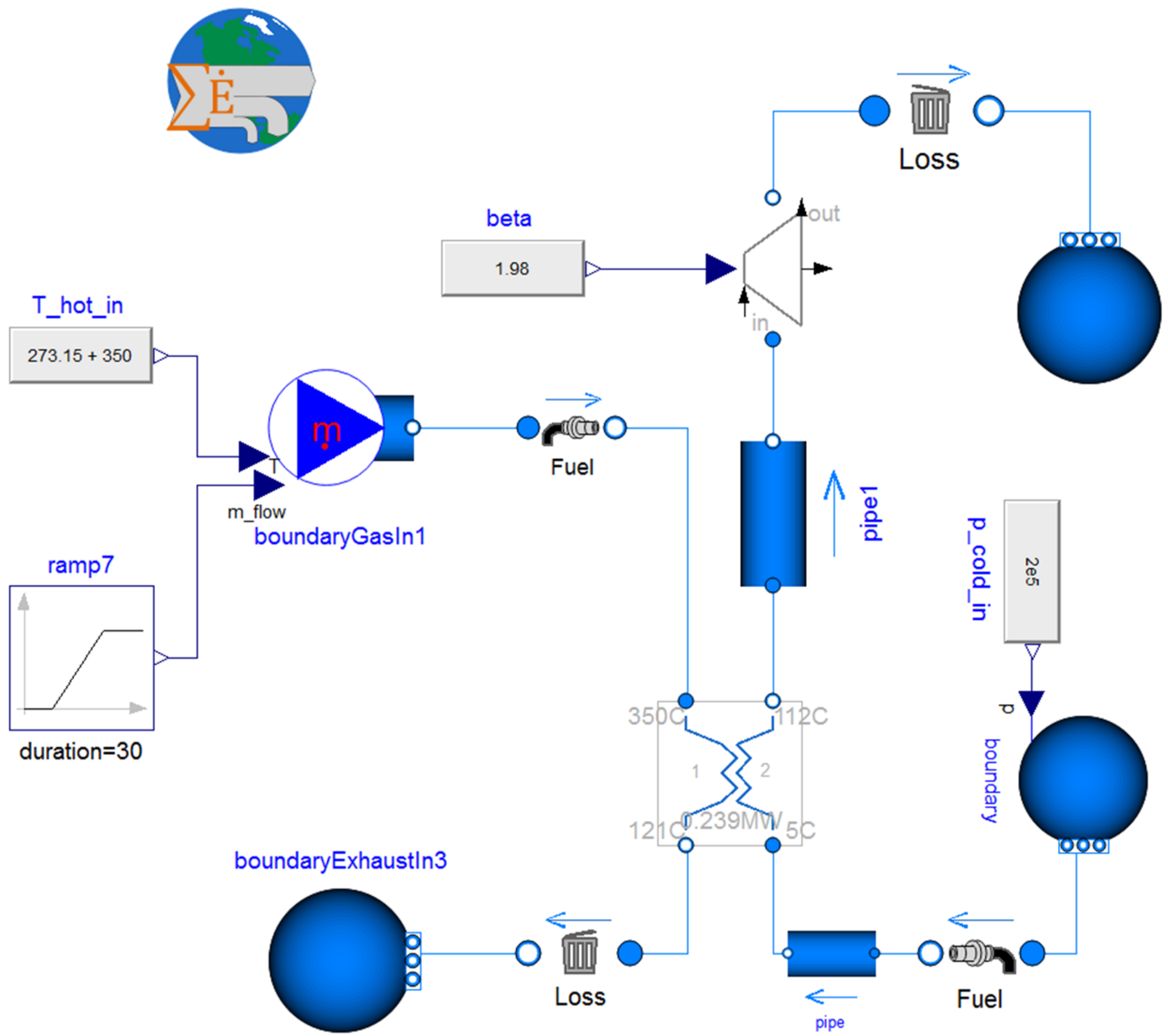
DLR

Definition of Exergy

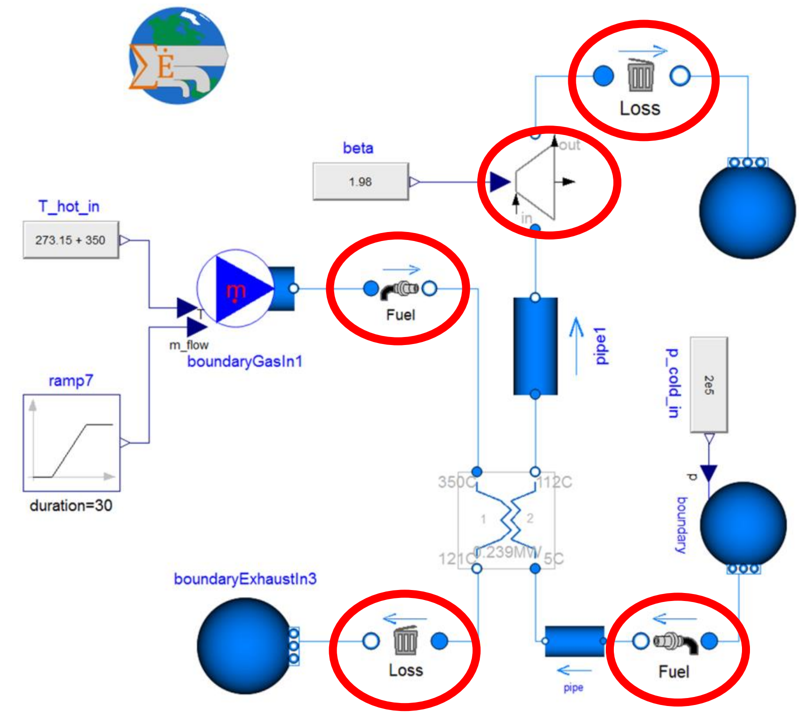
„Exergy of a thermodynamic system is the maximum theoretical useful work (shaft work or electrical work) obtainable as the system is brought into complete thermodynamic equilibrium with the thermodynamic environment while the system interacts with this environment only.“

(Tsatsaronis, 2007)

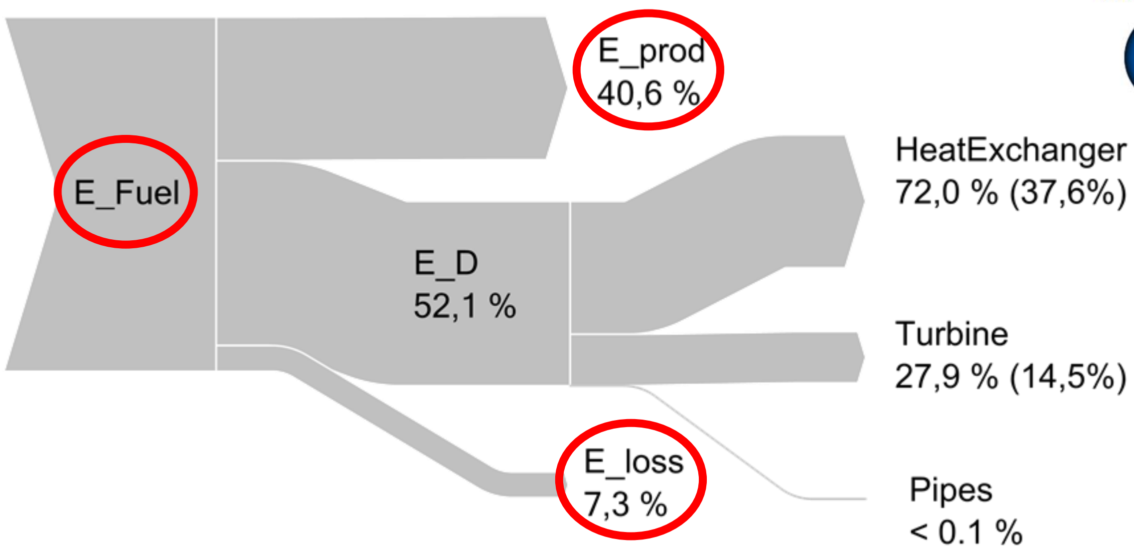
Motivation – Example Energy Conversion System



Motivation – Example Energy Conversion System



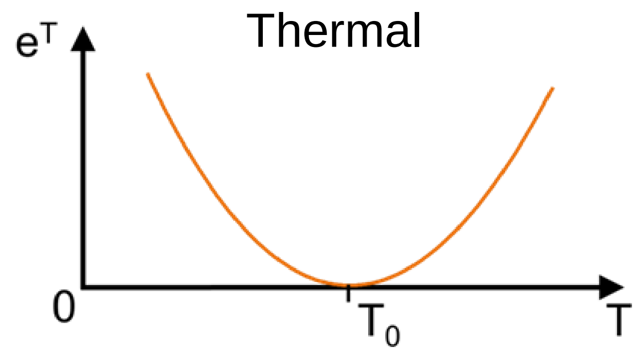
Exergetic Analysis

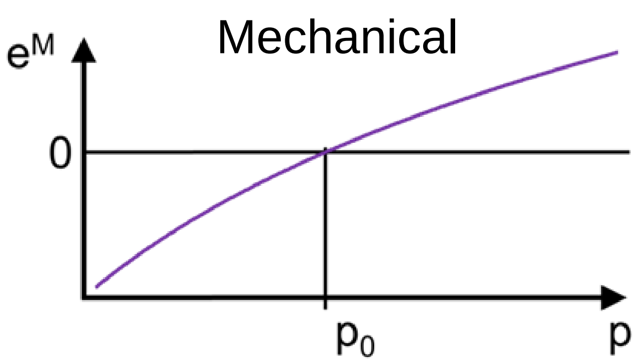


Introduction to Exergy Analysis
Physical and Chemical Exergy

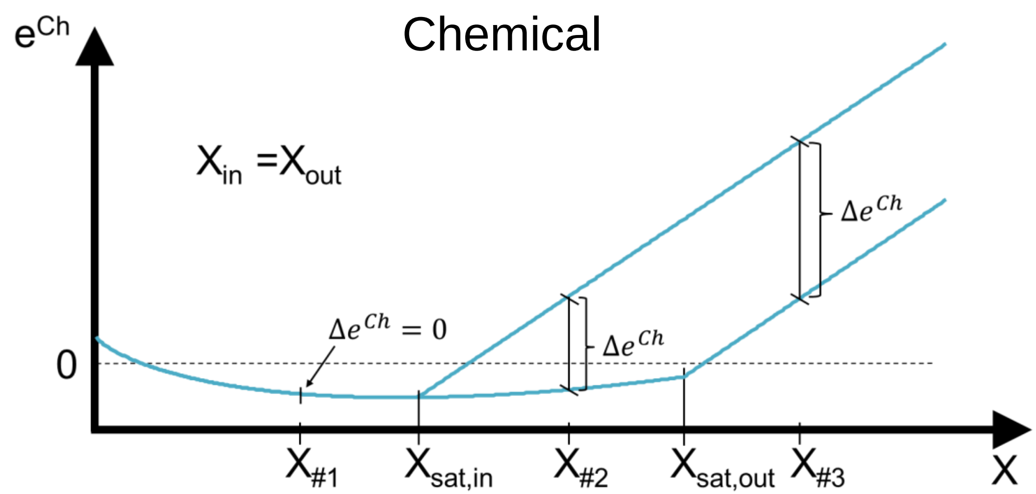
Exergy flow of fluid stream $$\dot{E}_{t,i} = \dot{m}_i \cdot [h_i - h_0 - T_0 \cdot (s_i - s_0)]$$

Splitting of specific exergy $$e_{t,i} = e_i^T + e_i^M + e_i^{Ch}$$



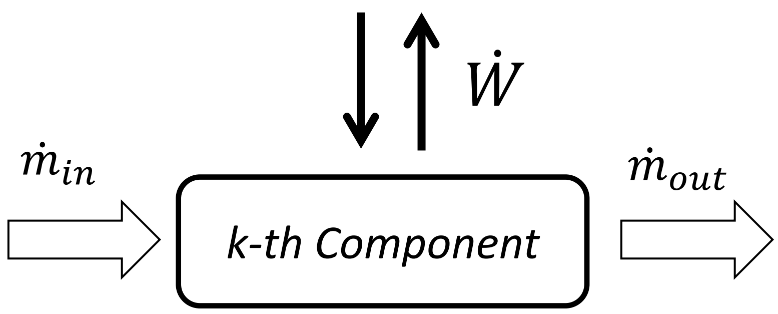


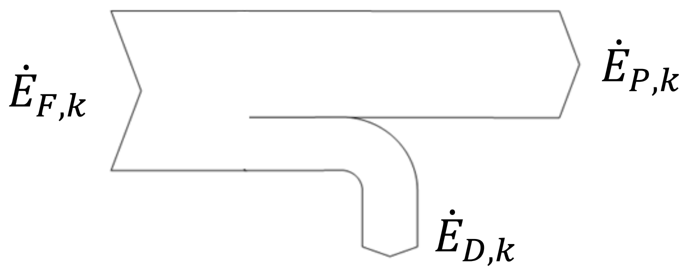
Assumptions:
- Moist air treated as ideal gas
- Chemical exergy with approach of Szargut (Szargut, 1988)



Introduction to Exergy Analysis - Exergy Balances

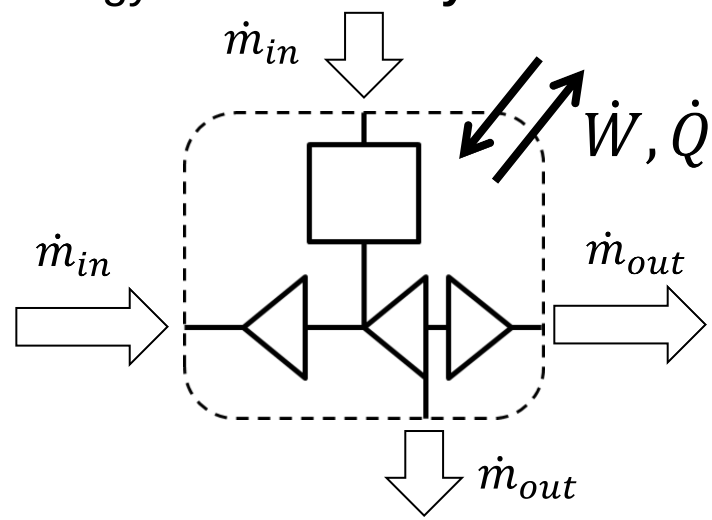
Exergy balance on **component** level

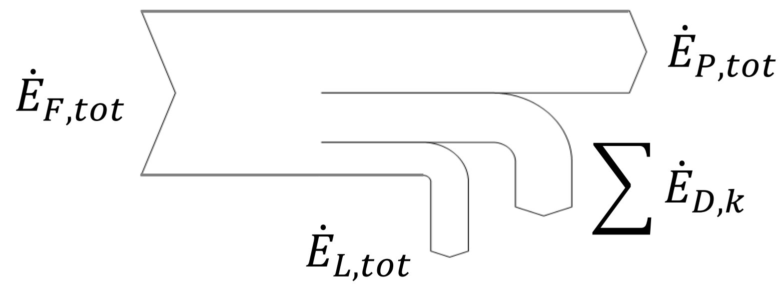




$$\dot{E}_{F,k} = \dot{E}_{P,k} + \dot{E}_{D,k}$$

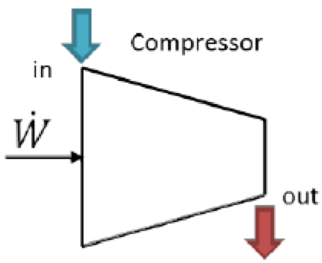
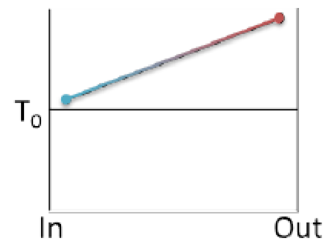
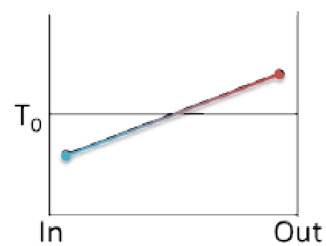
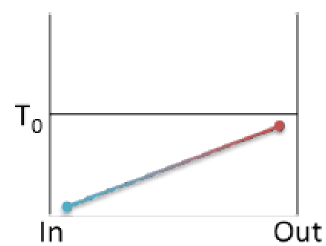
Exergy balance on **system** level





$$\dot{E}_{F,tot} = \dot{E}_{P,tot} + \sum \dot{E}_{D,k} + \dot{E}_{L,tot}$$

Introduction to Exergy Analysis - Exergy Balances

Component schematic	*Operation Conditions*	*Exergy Rates*
	 $T_{in} \geq T_0$	$\dot{E}_{Fuel} = \dot{W} + \dot{E}^{Ch}_{in} - \dot{E}^{Ch}_{out}$ $\dot{E}_{Prod} = \dot{E}^{Ph}_{out} - \dot{E}^{Ph}_{in}$
	 $T_{in} \leq T_0 \leq T_{out}$	$\dot{E}_{Fuel} = \dot{W} + \dot{E}^{T}_{in} + \dot{E}^{Ch}_{in} - \dot{E}^{Ch}_{out}$ $\dot{E}_{Prod} = \dot{E}^{Ph}_{out} - \dot{E}^{M}_{in}$
	 $T_{out} \leq T_0$	$\dot{E}_{Fuel} = \dot{W} + \dot{E}^{T}_{in} - \dot{E}^{T}_{out} + \dot{E}^{Ch}_{in} - \dot{E}^{Ch}_{out}$ $\dot{E}_{Prod} = \dot{E}^{M}_{out} - \dot{E}^{M}_{in}$

Exergy Analysis – Fuel and Product Balances

Component	Definition Cases
Heat Exchanger	6
Compressor	3
Turbine	3
Water Separator	1
Water Injector	3
TCV / Flow Resistance	1
Mixer	1

Integration into Model-Based Design Environment – Requirements

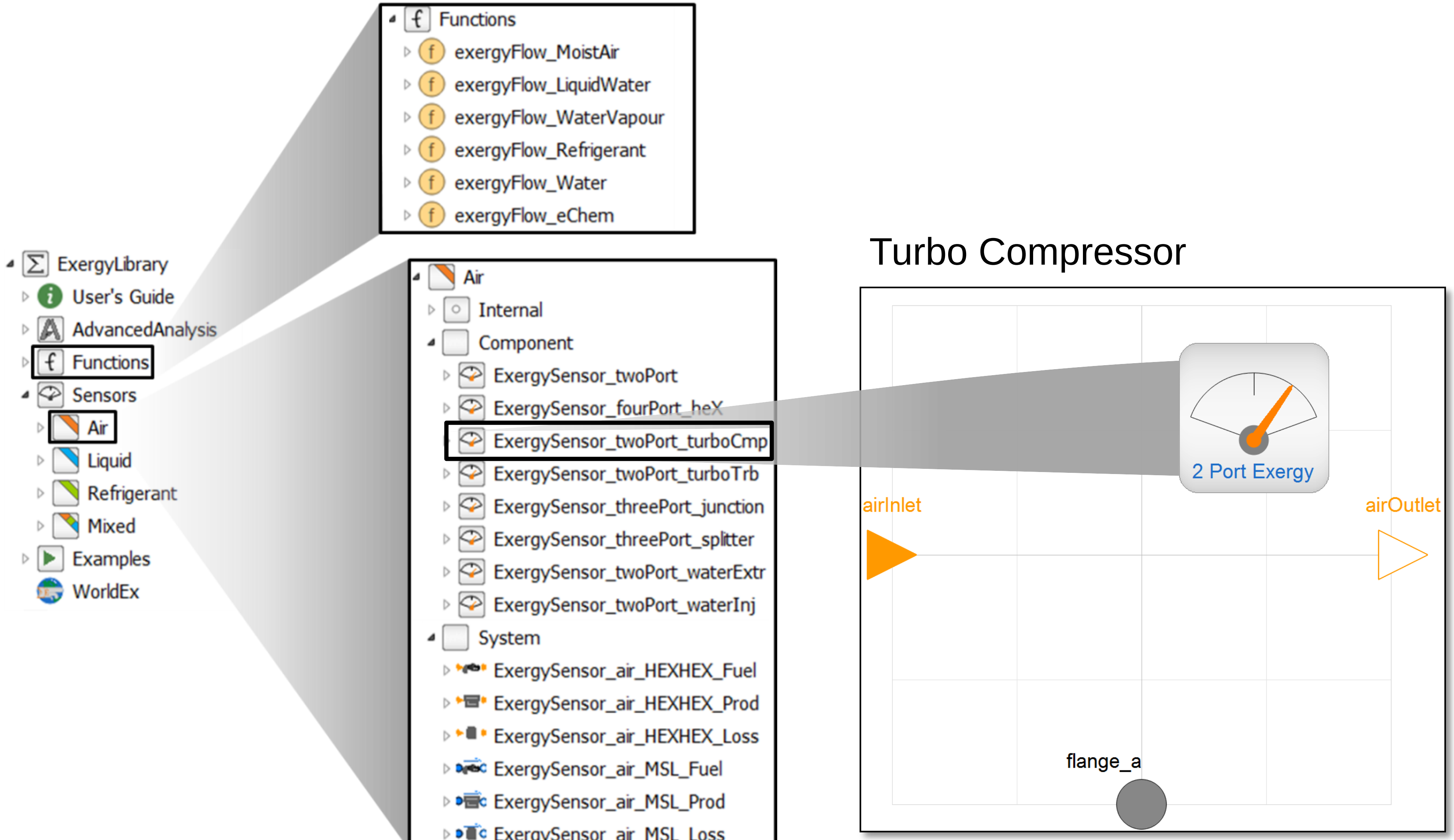
Exergy-Based Methods

- Retrieve thermodynamic state of all energy streams entering and exiting a component
- Identify the aim the component's energy conversion
- Select the appropriate exergy balance of fuel and product exergy rates depending of the operation condition and reference environment
- Allow a user defined exergy analysis on system level using the component's based analysis
- Centralized propagation of reference environment on system level among all components
- Media models must provide appropriate functions to calculate further thermodynamic data

Model-Based Environment

- Generic approach for easy integration into any thermo-fluid library
- Compliant with Modelica Standard Library (Usage of MSL Media models and connectors)
- Minor impact on numerical computation

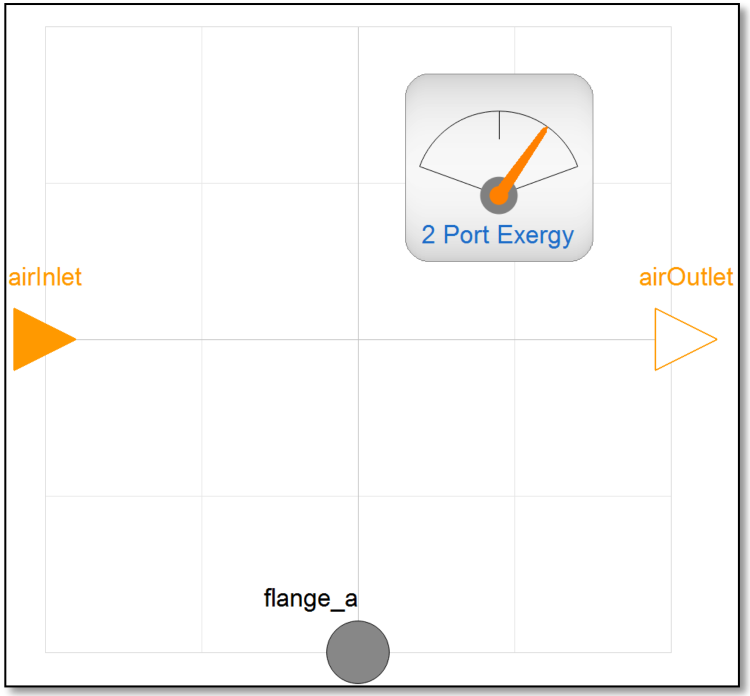
Integration into Model-Based Design Environment – Component Level



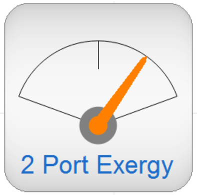
Integration into Model-Based Design Environment – Component Level

```
// Reference environment
// -----------------
SIunits.Temperature T_ref =
    worldEx.T_ref
"Reference Temperature for Exergyflow";
SIunits.Pressure p_ref = worldEx.p_ref
"Reference Pressure for Exergyflow";
SIunits.MassFraction X_ref[:] =
    worldEx.X_ref;
outer ExergyLibrary.World worldEx;

//*************Sensors*************
Sensors.Air.ExergySensor_twoPort_turboCmp
     exergySensor_twoPort(
airMediumA(state=AirMedium.setState_phX(
            portA.p,
            portA.h,
            portA.Xi),
            redeclare package AirMedium
    = AirMedium),
airMediumB(state=AirMedium.setState_phX(
            portB.p,
            portB.h,
            portB.Xi),
            redeclare package AirMedium
    = AirMedium),
 m_flow=m_flow,
 power=power,
 T_ref = T_ref,
 p_ref = p_ref,
 X_ref = X_ref);
```

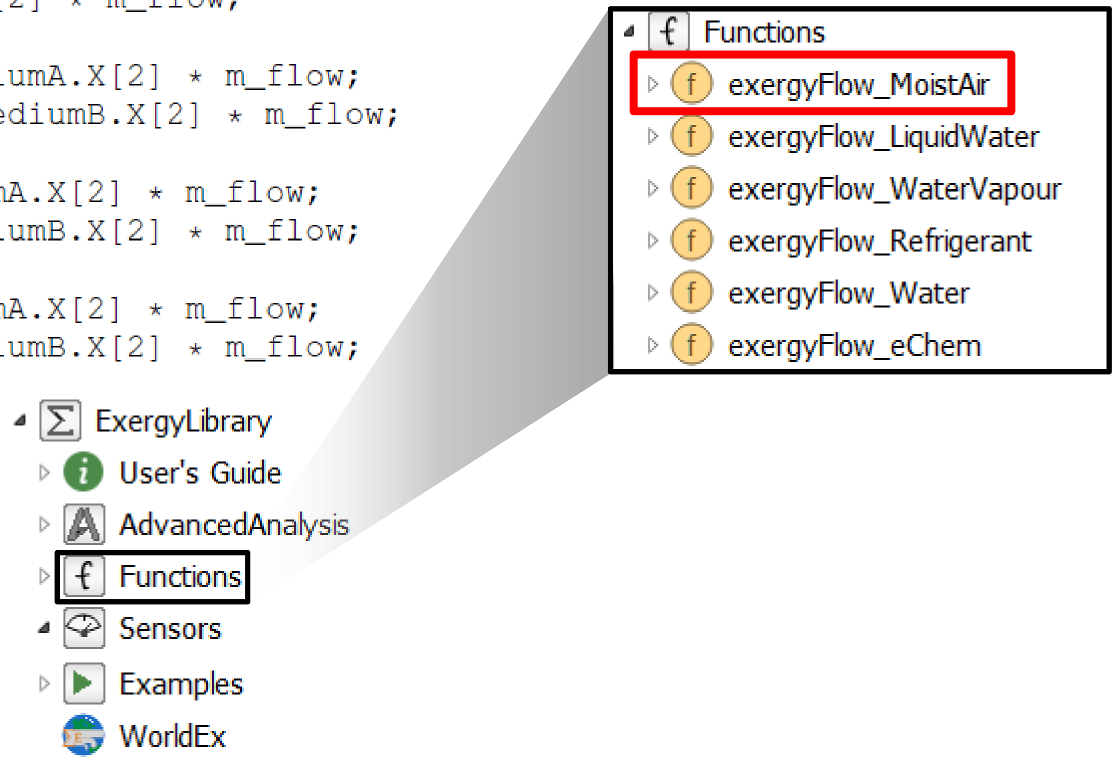
Turbo Compressor



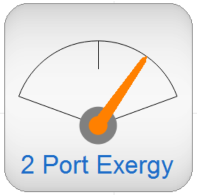
Integration into Model-Based Design Environment – Component Level




```
// Exergy implementation
// ----------------------
(e_t_in,e_therm_in,e_mech_in,e_chem_in) = Functions.exergyFlow_MoistAir
 airMediumA.state, state_ref);
(e_t_out,e_therm_out,e_mech_out,e_chem_out) = Functions.exergyFlow_MoistAir(
 airMediumB.state, state_ref);
E_t_in = e_t_in * airMediumA.X[2] * m_flow;
E_t_out = e_t_out * airMediumB.X[2] * m_flow;

E_therm_in = e_therm_in * airMediumA.X[2] * m_flow;
E_therm_out = e_therm_out * airMediumB.X[2] * m_flow;

E_mech_in = e_mech_in * airMediumA.X[2] * m_flow;
E_mech_out = e_mech_out * airMediumB.X[2] * m_flow;

E_chem_in = e_chem_in * airMediumA.X[2] * m_flow;
E_chem_out = e_chem_out * airMediumB.X[2] * m_flow;
```



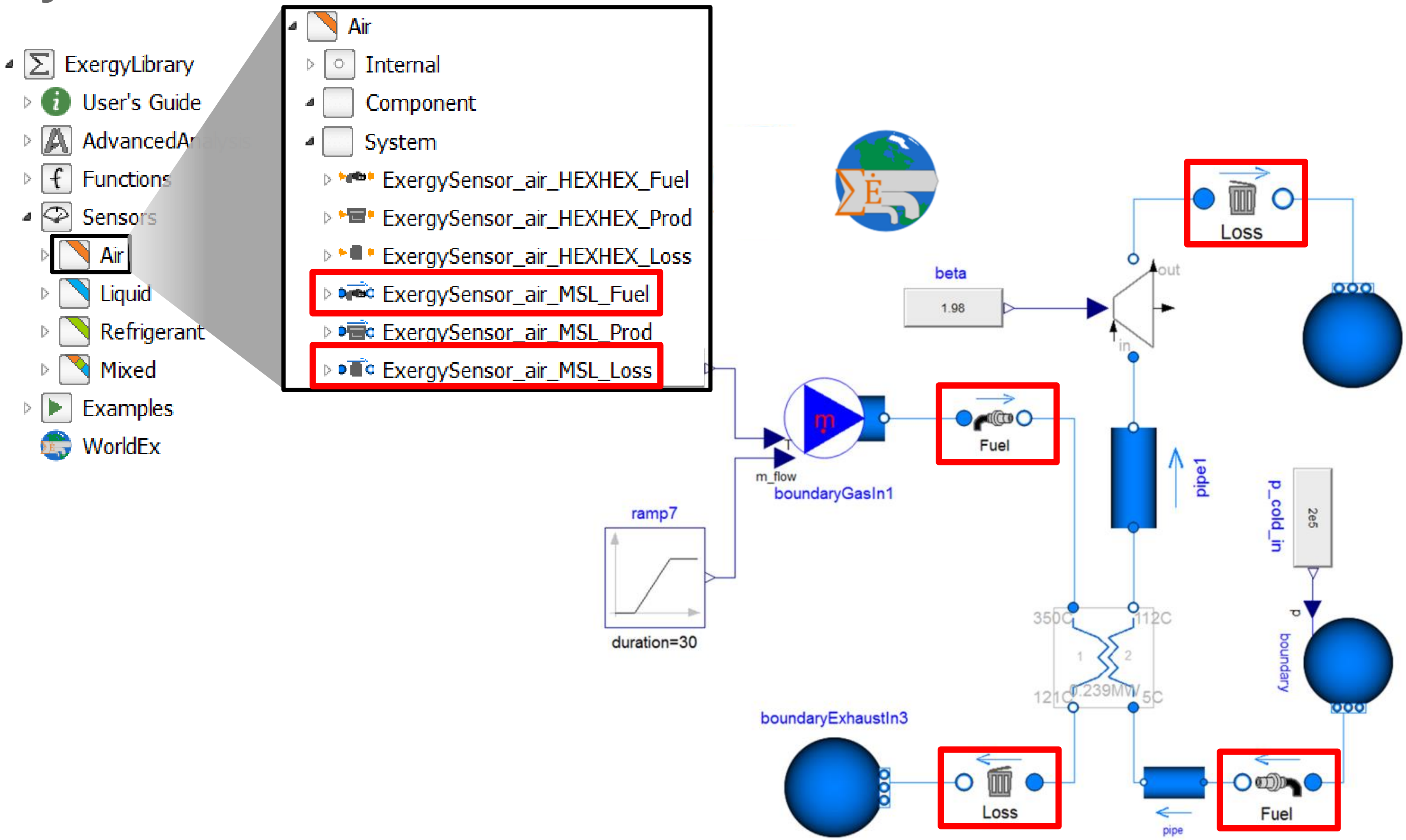
Integration into Model-Based Design Environment – Component Level



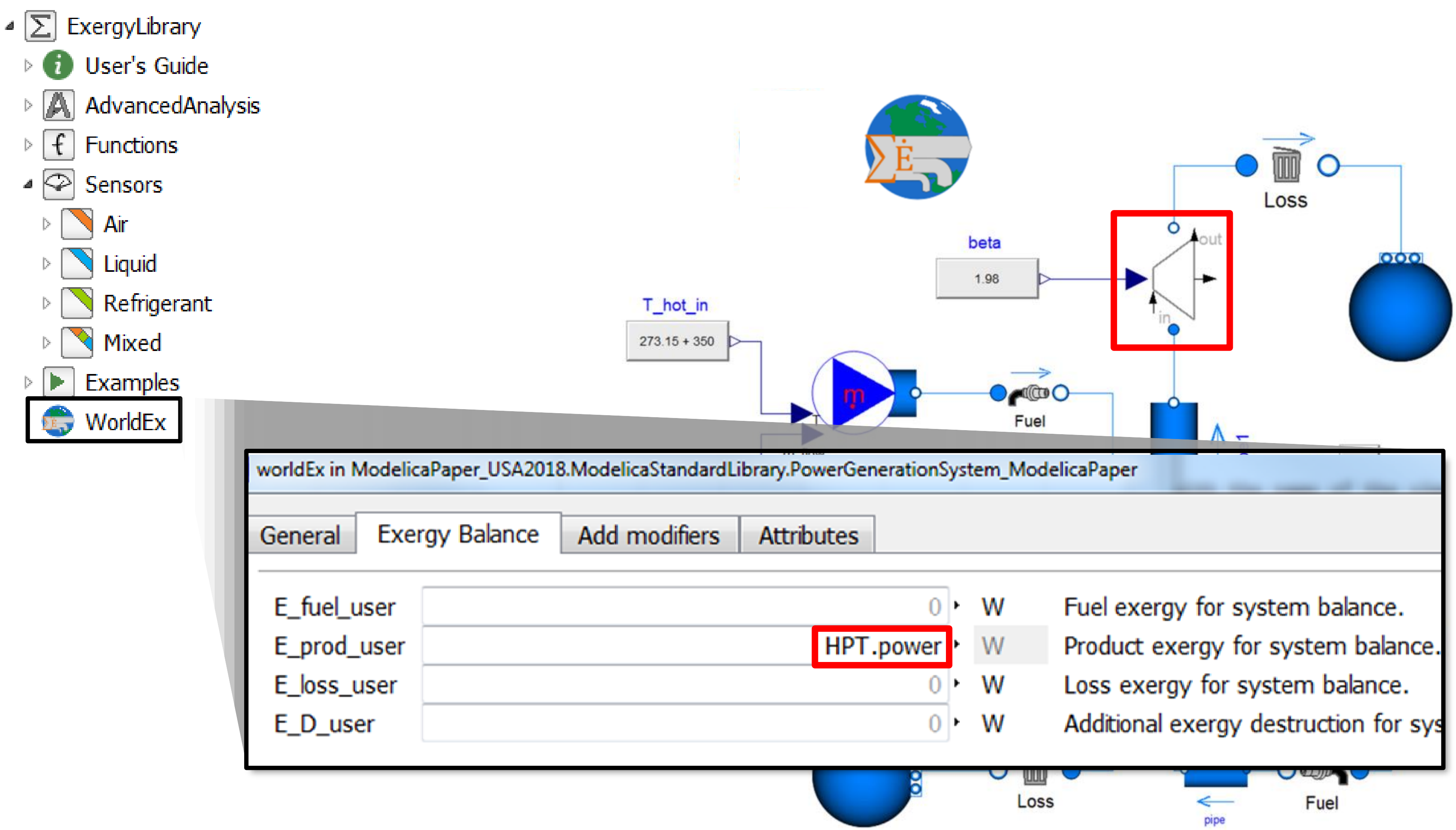

```
// Calculation of fuel and product Exergy
if m_flow <= 0 then
E_fuel = 0;
E_prod = 0;
case_T = 0;
else
if airMediumA.T > state_ref.T then
case_T = 1;
E_fuel = abs(power) + E_chem_in - E_chem_out;
E_prod = E_therm_out - E_therm_in + E_mech_out - E_mech_in;
elseif airMediumB.T >= state_ref.T and airMediumA.T <= state_ref.T then
case_T = 2;
E_fuel = abs(power) + E_therm_in + E_chem_in - E_chem_out;
E_prod = E_therm_out + (E_mech_out - E_mech_in);
elseif airMediumB.T < state_ref.T then
case_T = 3;
E_fuel = abs(power) + (E_therm_in - E_therm_out) + E_chem_in - E_chem_out;
E_prod = E_mech_out - E_mech_in;
else
case_T = 100;
E_fuel = 0;
E_prod = 0;
end if;
end if;
E_D = E_fuel - E_prod;
exergy_eff = E_prod / max(eps,E_fuel);
```

$$\dot{E}_{F,k} = \dot{E}_{P,k} + \dot{E}_{D,k}$$

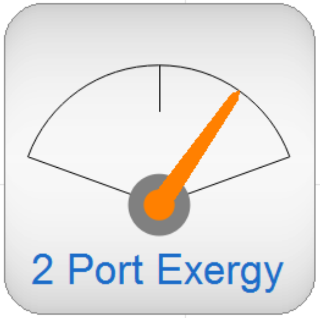
Integration into Model-Based Design Environment – System Level



Integration into Model-Based Design Environment – System Level



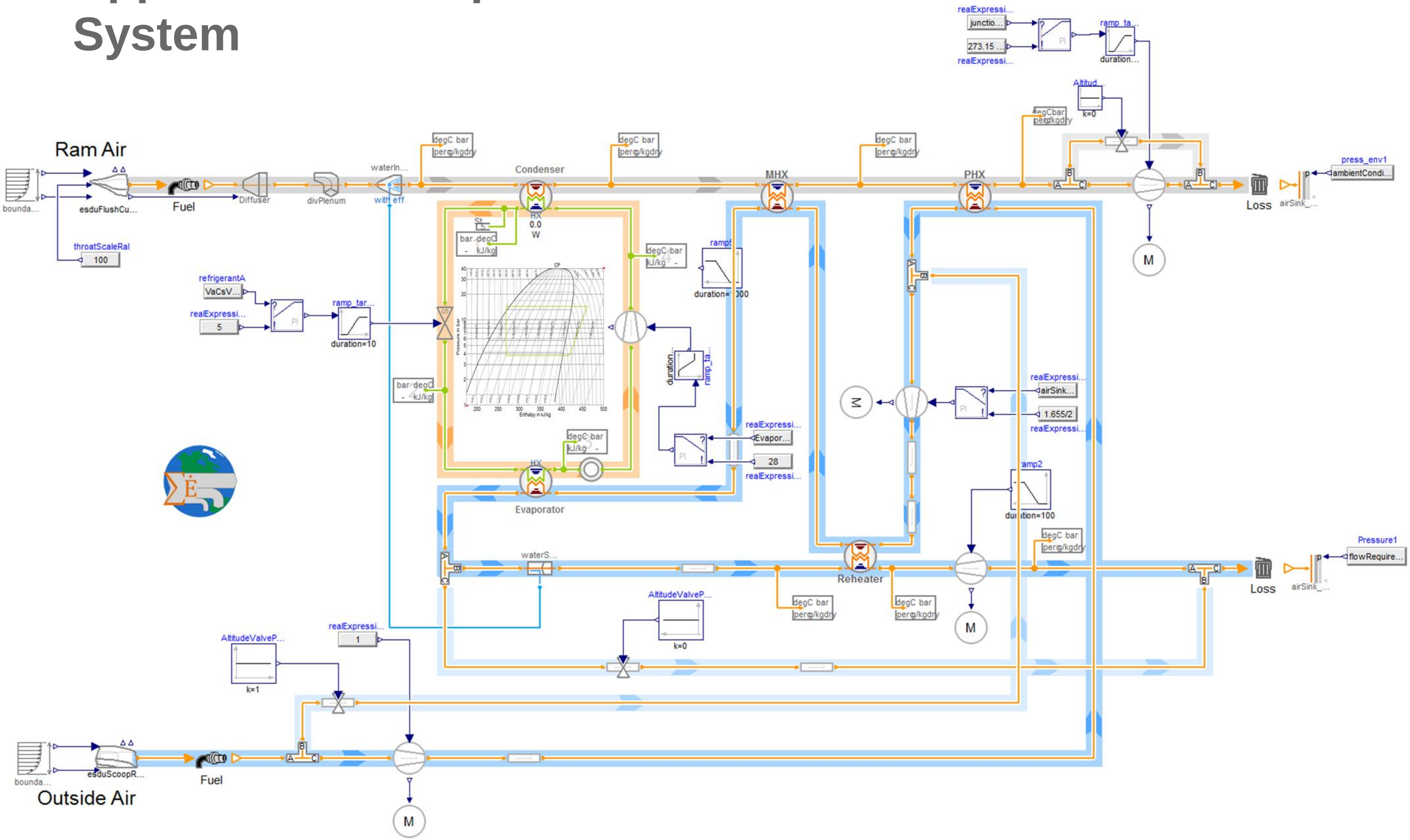
Integration into Model-Based Design Environment – Identifier



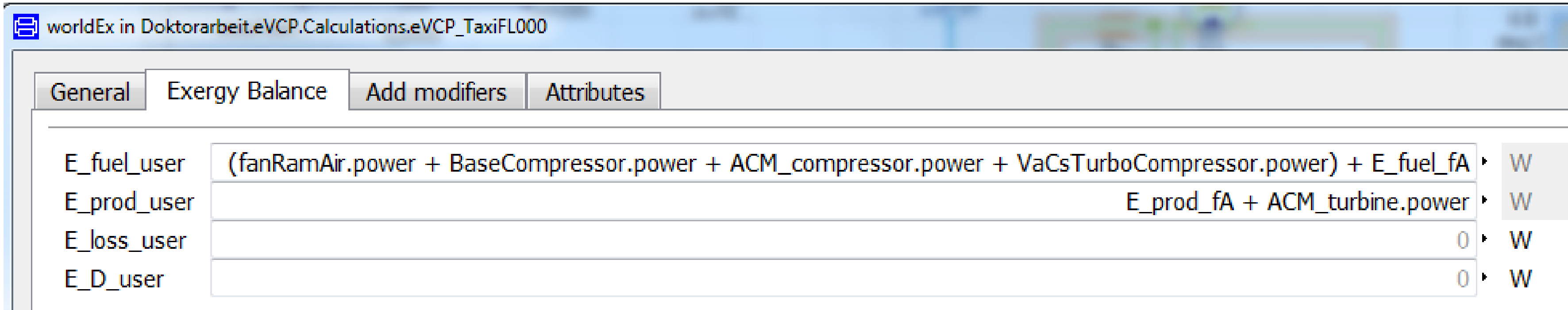

```
UID.UniqueID uniqueID(group="exergy") a;
parameter String instanceName =
    getInstanceName();
equation
   worldEx.E_D[uniqueID.uid+1] = E_D;
   worldEx.instanceName[uniqueID.uid+1]
    = instanceName;
```

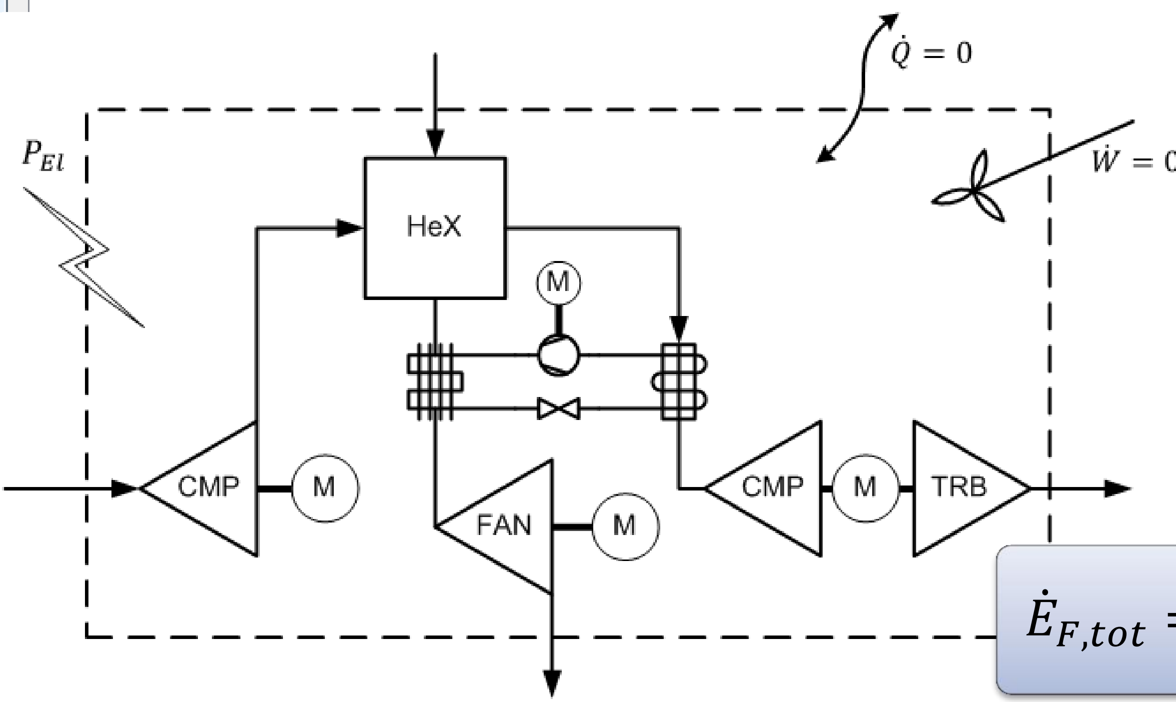
```
UID.GroupTotal groupTotal(group="
  exergy") a;
Modelica.SIunits.Power E_D[groupTotal.
  total];
Modelica.SIunits.Power E_D_total = sum
  (E_D) + E_D_user;
String instanceName[groupTotal.total];
```

Application Example – Aircraft Environmental Control System



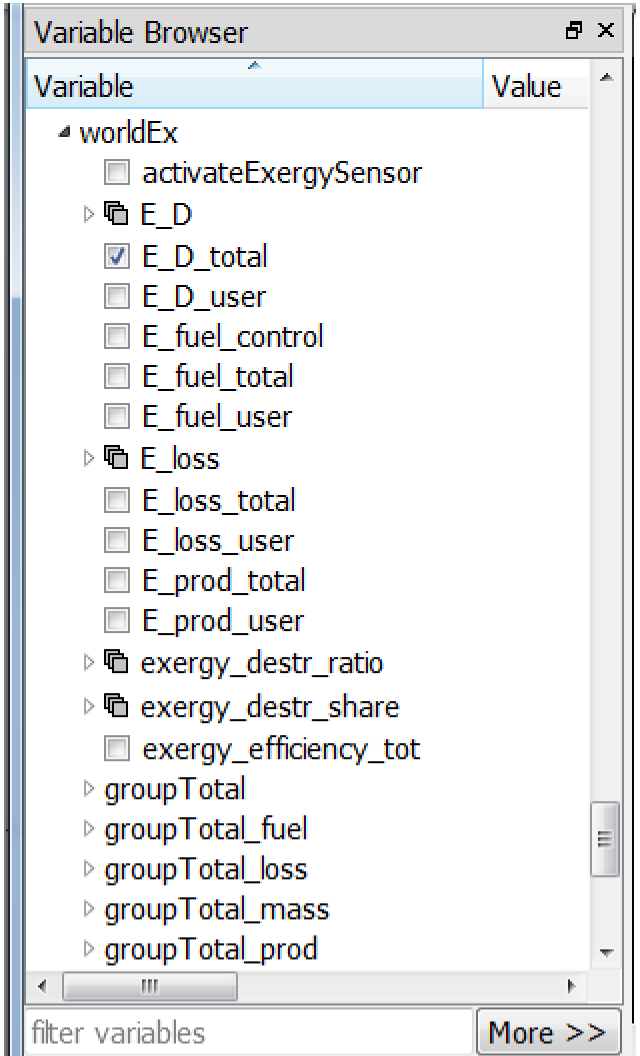
Application Example – Aircraft ECS
Definition of System Balances

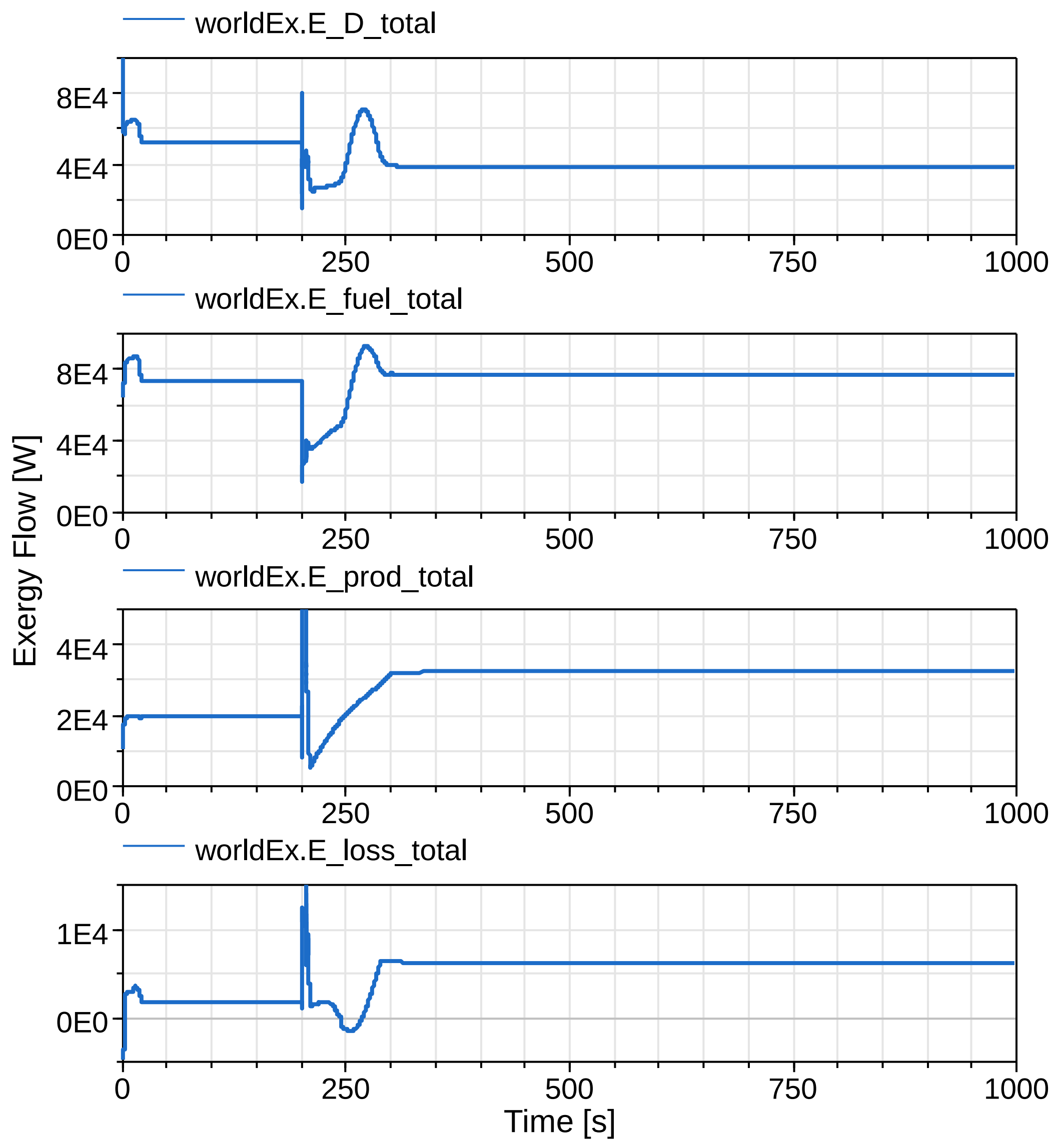




$$\dot{E}_{F,tot} = \dot{E}_{P,tot} + \sum \dot{E}_{D,k} + \dot{E}_{L,tot}$$

Application Example – Aircraft ECS Results





Application Example – Aircraft ECS
Results

ExComponentNames_eVCP_referenceArchitecture_VaC_TAXI_RAC.txt - Editor

Datei Bearbeiten Format Ansicht ?

```
Total Number of components containing exergy sensors = 30
Structure of vectors based on E_D:
[1]MHX
[2]Evaporator
[3]Reheater
[4]waterSeparator
[5]waterInjector
[6]fanRamAir
[7]BaseCompressor
[8]ACM_turbine
[9]PHX
[10]ACM_compressor
[11]Condenser
[12]VaCsTurboCompressor
[13]VaCsValve
[14]junction2
[15]junction13
[16]flowModelAirARamAir4
[17]flowModelAirARamAir1
[18]junction1
[19]Venturi_ACM_Compressor
[20]Ozone_Converter
[21]Pack_Venturi
[22]WaterExtractorAirFlowResistance
[23]AltitudeValveDuctAirFlowResistance
[24]Diffuser
[25]divPlenum
[26]junction4
[27]Base_Compresor_Check_Valve1
[28]junction5
[29]junction6
[30]Base_Compresor_Check_Valve
Full paths to exergy sensors:
eVCP_referenceArchitecture_VaC_TAXI_RAC.MHX.exergySensor
eVCP_referenceArchitecture_VaC_TAXI_RAC.Evaporator.exergyS
eVCP_referenceArchitecture_VaC_TAXI_RAC.Reheater.exergySen
eVCP_referenceArchitecture_VaC_TAXI_RAC.waterSeparator.exe
eVCP_referenceArchitecture_VaC_TAXI_RAC.waterInjector.exer
eVCP_referenceArchitecture_VaC_TAXI_RAC.fanRamAir.exergySe
```

Conclusion

- Development of an Exergy Library in Modelica
- Integration of exergy-based methods into model-based environment:
    - Exergy balances available for standard components of an aircraft ECS
    - Reference environment centralized propagated
    - User defined exergy balances on system level
- Library can be integrated into any thermo-fluid library
- Compliant with MSL
- Only minor impacr on numerical computation

Remarks

- Standardized formulation and naming of equations for the thermodynamic properties should be ensured within media models

- Integration of „identifier“ into MSL for automated collection of variables and data on system level

Thank you for your attention!

Daniel Bender
German Aerospace Center (DLR)
Institute of System Dynamics and Control

The American Modelica Conference 2018
Boston, October 09th/10th, 2018

DLR

DLR