D6.1.1 – Requirements for static model analysis

WP 6.1: Static model analysis

Work Package 6: Modeling and simulation services

MODRIO (11004)

Version 0.5

Date 30/08/2013

Authors

Eric Thomas	Dassault-Aviation
Emmanuel Ledinot	Dassault-Aviation

Summary

Executive summary

This document takes part of D6.1.1 – it describes needs for static model analysis for business aircrafts.

Contribution: Dassault Aviation: Provide requirements and assessment of WP results. The WP results will be tested, if possible, in the demonstrator (WP8.4).

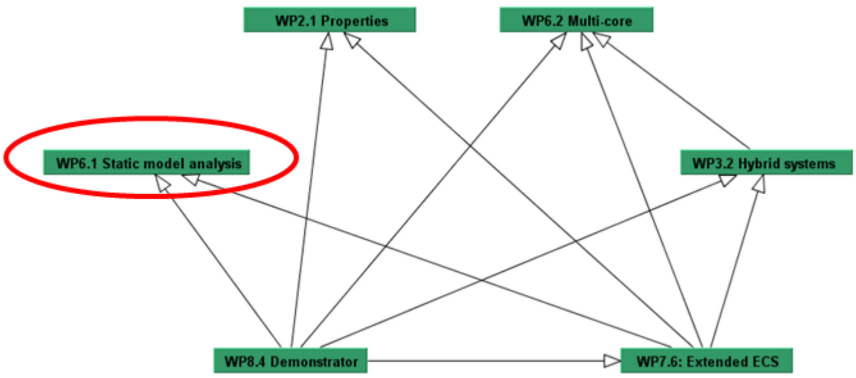
Delivery: Requirements for static model analysis.

1. SUMMARY

This document describes needs for static model analysis for business aircrafts.

The purpose is to be able to use an important feature of acausal Modelica model that is the possibility to inverse a model by searching freed parameter values when adding additional constraints. But, to be able to get results, the new generated equation system should be solvable. The purpose is then in this Work Package to analyze if help could be given to designers when the tool fail to get results and guide them to detect the cause and find solutions.

Dependencies with other Work-Package:



This work package doesn’t depend on other Work-Packages, but should give improvements that will be used within WP 7.6 and 8.4.

1.1 Acronyms

- BIZJET : Business aircraft
- CAU : Cold Air Unit
- CFD : Computer Fluid Dynamics
- ECS : Environmental Control System
- FMI : Functional Mock-up Interface
- Thermo-fluid
    - m : mass (kg)
    - qm : mass flow rate (kg/s)
    - qmh : masse flow rate multiplied by inflowing specific enthalpy of the Medium (W)
    - Φ : thermal power (W)
    - C.ω : mechanical power (W)
    - u : specific internal energy
    - U : internal energy

2. USAGE OF STATIC ANALYSIS

The purpose is to be able to analyze the behavior and performances of systems as such as those represented in figure n°1.

Such system is design with different partners, which supply part of the system, and can provide models. The aim of Dassault Aviation, as systems integrator, is to be able to master the whole system behavior.

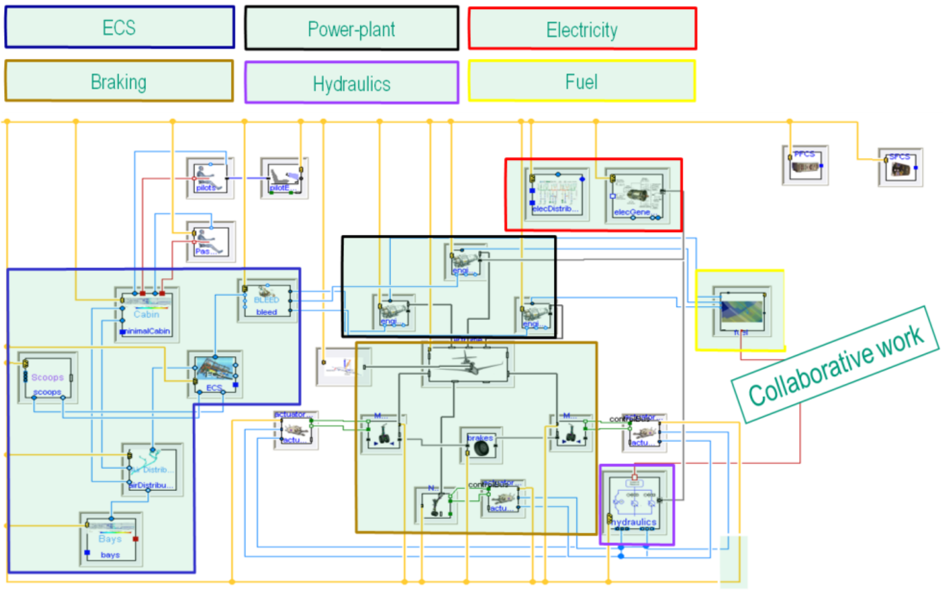


Figure 1 : conventional architecture with several aircraft systems

Physical behavior assessment of architecture is often performed by direct simulations. With models associated to components of the architecture, and sets of parameters, we can obtain respective sets of behavior results, and analyse them.

The purpose here is to be able to use an important feature of acausal Modelica model which is the possibility to inverse a model by searching freed parameter values when adding additional constraints. In fact, values of particular parameters may be sometimes not known, but behavior or constraints can be got from other analysis or tests.

These values could be obtained by "state" estimation as studied in WP3. They could be also determined by solving the inverse problem.

But, to be able to get results, the new generated equation system should be solvable. The purpose is then to analyze if help could be given to designers when the tool fail to get results and guide them to detect the cause and find solutions.

3. REQUIREMENTS

Debugging based on causality analysis is motivated by:

- The introduction of synchronous state-machines :
  - Boolean algebraic loops will become more frequent and need support to be fixed on large models,
  - Synchronous continuous time state machines used for control mode modeling, or for plant dysfunctional modeling, will enhance the occurrence of dynamic causality structures. Potentially: as many different DAEs to be integrated (and associated causality structures) as reachable logical states of the synchronous control skeleton.
- Providing support to non expert Modelica users: support to physical causality awareness, to prevent newcomers from introducing physical inconsistencies into models, would be nice to have for cost-effectiveness of modelers,
- Coping with large models: the larger, the harder to fix causality issues (physical and computational),
- Inversion: support to analyze the feasibility conditions of inversion.

Causality analysis is envisioned as a background task:

- At model level: graphical display of the causality cycles or inconsistencies,
- At ".mof" level: dedicated browser of linked dependencies between variables.

Hereafter the document is dedicated to causality analysis supporting inversion.

Inverse models should be:

- As quick as possible (see WP6.2)
- Freed of parameters and defining constraints should be as easy as possible
- Compatible with
  - ♦ Large systems
  - ♦ Concealed models (encrypted, encapsulated within FMI …)
  - ♦ Static or dynamic models
  - ♦ Mixed continuous, discrete variables and equations, and synchronous features

4. PROCESS

To analyze the invertibility of the system and by taking into account existing knowledge on bond graph (pure or pseudo bond graph), several steps seem to be required:

1. Select the architecture to study
2. Define the problem: constraints, inputs and outputs, environment and scenario
3. Define the part of the architecture involved in the analysis
4. If needed (efficiency of the analysis, time computation), restrain the system to the minimum/smaller part of the architecture for the analysis (i.e. replacing surrounding components by boundary components.)
5. Replace virtually or physically components from acausal to causal models
6. Make the invertibility analyze

- Identify the paths in the architecture which links variables (causality analysis), equations of components to outputs (design parameters) and constraints of the problem
- If possible, highlight these links and components along the paths to the users (designers, architecture analysts);
- Diagnose the problem
    a. Identify components which "breaks" the paths, and prevent to solve the problem
    b. Show diagnosis, involved equations and variables.
    c. Give advice to change part of the models to allow inversion

7. Replace components with better causal models
8. Check the new causal paths
9. Try again to inverse the system and solve the problem

5. EXPLANATIONS

To test the process defined below, a Modelica library is being developed. The purpose is to implement step by step the process in a use-case. In particular, bond graph components are implemented to replace the acausal components by causal ones, and obtain an equivalent architecture with the same variables and the same "type of" relations between these variables.

But the aim is really not to use such a library in the future, which should be a step backwards regarding capabilities of Modelica. But, the aim is to analyze the way is done the diagnosis with bond graph and try to understand if such a diagnosis could be done directly on the acausal model.

6. STATUS

Currently, the library is under development. Components are described in Appendix "Use case". It is not yet completed, and several issues are still to be solved; but it is sufficient to discuss with partners on the required implementations, and define the best way to step forward on invertibility analysis.

7. REFERENCES

[R01] Modelica Conference 2012: Collaborative complex system design applied to an aircraft system; Eric Thomas, Michel Ravachol, Jean Baptiste Quincy and Martin Malmheden

[R02] DGT 113.239 Eurosyslib Report "State of the art for Fluid modeling with Modelica", E. Thomas

[R03] DGT 116083A Eurosyslib Report "Properties Modeling"

[R04] DGT139136_D6 2 1 "Requirements for multi-core compilation"

[R10] MultiBonLib Modelica library, Dirk Zimmer, ETH Zürich, Prof. Dr. Francois E. Cellier

[R11] ThermoBondLib Modelica Library, Prof. Dr. François E. Cellier

[R12] BondLib Modelica Library, Prof. Dr. François E. Cellier

8. APPENDIX

8.1 ECS use case

8.1.1 Description of the system

An environmental control system (ECS) was selected because this system is particularly interesting. It combines several demonstrative features which can be applied to other systems afterwards.

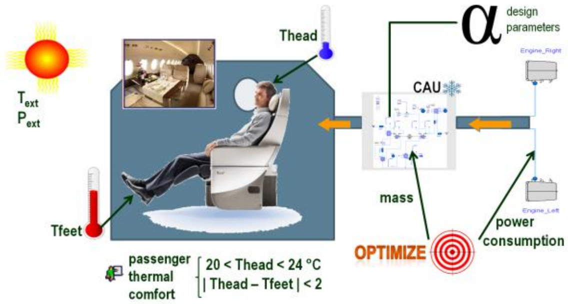


Figure 2: ECS Sizing engineering problem

For this reason, this generic model of ECS was previously used as a base during ITEA2 Eurosyslib for properties modelling (see [R03]), and collaborative system design during CSDL (see [R01]). In ITEA2 Modrio it is used to investigate static analysis modelling.

This generic model is written in Modelica. It is composed of basic sub-systems. Air flow comes from two engines modelled as boundaries with fixed pressure and temperature. A bleed mixes the two flows and provides the resulting flow to the Cold Air Unit (CAU) which regulates mass flow and energy given to the Cabin. Usually the energy flow rate coming from the CAU is provided to the different parts of the Cabin and to the Bays through a complex piping system. In this example, only a Cabin is taken into account.

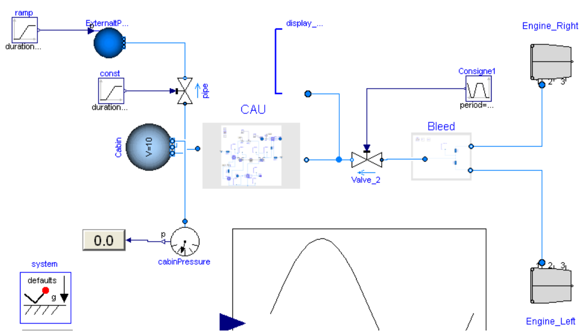


Figure 3: Generic ECS

The CAU is composed of a compressor, a turbine, heat exchangers, pipes and a regulating valve controlled by a PI controller which uses the measured Cabin temperature and a temperature set point for the regulation, as shown in figure 4.

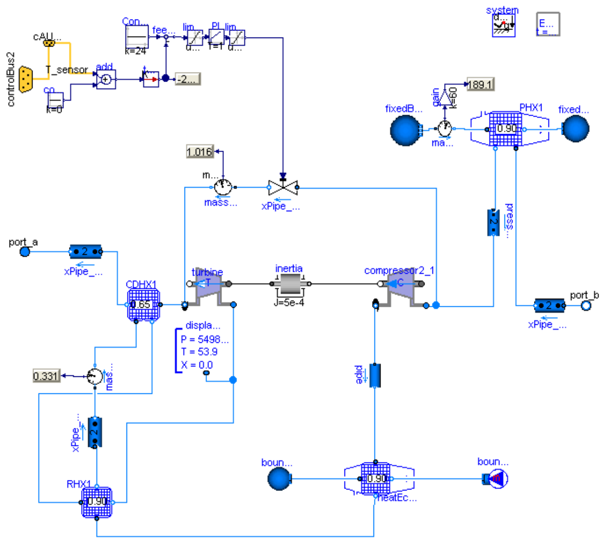


Figure 4 : Cold Air Unit

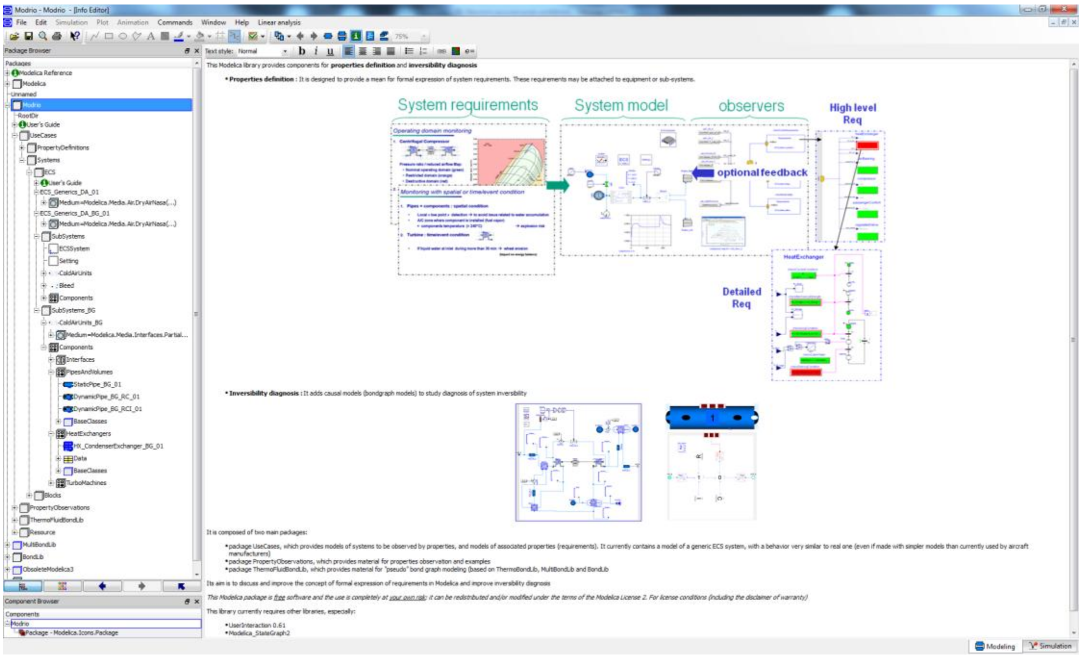
The control system, which is a currently an analogical control device, but could be partially digital (or with synchronous components)

8.1.2 Problem to solve

The purpose is to consider that condition on performances will be set (engine mass flow rates, system mass), and parameters (exchanger efficiencies, component sizes …) would be released, with may be guess values. The aim is to find these parameters.

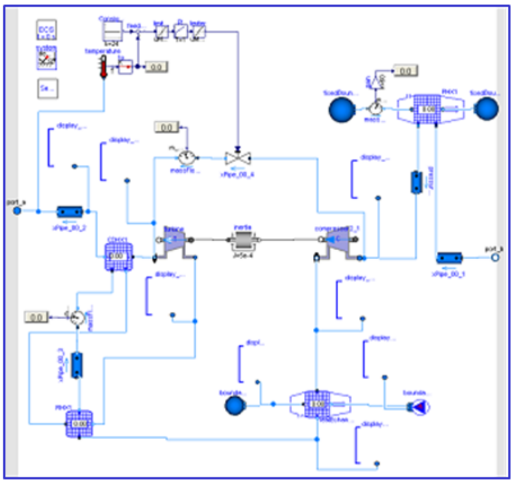
8.2 Modrio library

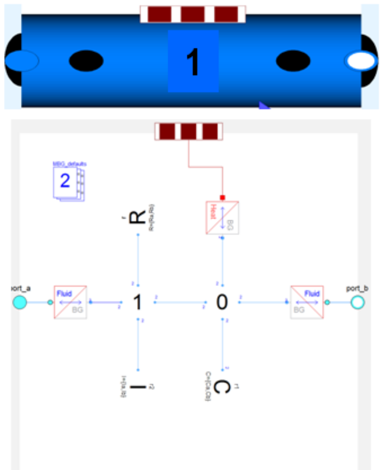
A new library called Modrio is being developed to test and illustrate needs regarding invertibility.



It was created from the Properties library (made during the ITEA2 project Eurosyslib for properties modeling investigations). Use cases and bond graph components have been added to the library. The components are described in the following paragraph.

The purpose is to be able to change acausal components by causal bond graph components and allow the first steps of the invertibility analysis.





8.2.1 Models of components

8.2.1.1 Causal models

The only interface parts which change are the connectors. The purpose is to have the same parameters, but model described internal with bond graph components.

The following paragraph defines Modelica components with acausal ports made to be able to connect internally to bond graph ports of the MultiBonLib library, through a new package called ThermoFluidBondLib, which replace the library ThermoBondLib, too far from the need.

8.2.1.2 Assumptions

The following causal models are described hereafter only for dry air. For use with, moist air additional variables (ratio of water in air, liquid water amount …) and equations should be used in addition.

8.2.1.3 Connectors

The connector of the acausal model is the one of Modelica.Fluid:

```
connector FluidPort
  "Interface for quasi one-
dimensional fluid flow in a piping network (incompressible or compressible, one or more phases, one or more s
ubstances)"

  replaceable package Medium = Modelica.Media.Interfaces.PartialMedium
    "Medium model" annotation (choicesAllMatching=true);

  flow Medium.MassFlowRate m_flow
    "Mass flow rate from the connection point into the component";
  Medium.AbsolutePressure p "Thermodynamic pressure in the connection point";
  stream Medium.SpecificEnthalpy h_outflow
    "Specific thermodynamic enthalpy close to the connection point if m_flow < 0";
  stream Medium.MassFraction Xi_outflow[Medium.nXi]
    "Independent mixture mass fractions m_i/m close to the connection point if m_flow < 0";
  stream Medium.ExtraProperty C_outflow[Medium.nC]
    "Properties c_i/m close to the connection point if m_flow < 0";
end FluidPort;
```

The original connector of ThermoBondLib is made of pure bond graph variables:

```
connector ThBondCon "Bi-directional thermo-bond graph connector"
  Modelica.SIunits.Temperature T "Temperature";
  Modelica.SIunits.Pressure p "Pressure";
  Modelica.SIunits.SpecificEnthalpy g "Gibbs potential";
  Modelica.SIunits.ThermalConductance Sdot "Entropy flow";
  Modelica.SIunits.VolumeFlowRate q "Volume flow";
  Modelica.SIunits.MassFlowRate Mdot "Mass flow";
  Modelica.SIunits.Entropy S "Entropy";
  Modelica.SIunits.Volume V "Volume";
  Modelica.SIunits.Mass M "Mass";
  Real d "Directional variable";
  Boolean Exist "True if substance exists";

end ThBondCon;
```

Variables of this connector are very far from the Modelica.Fluid connector.

Therefore, a new connector, for “pseudo” bond graph has been created, to be closer to the acausal connector and allow fewer modifications in relations within the models:

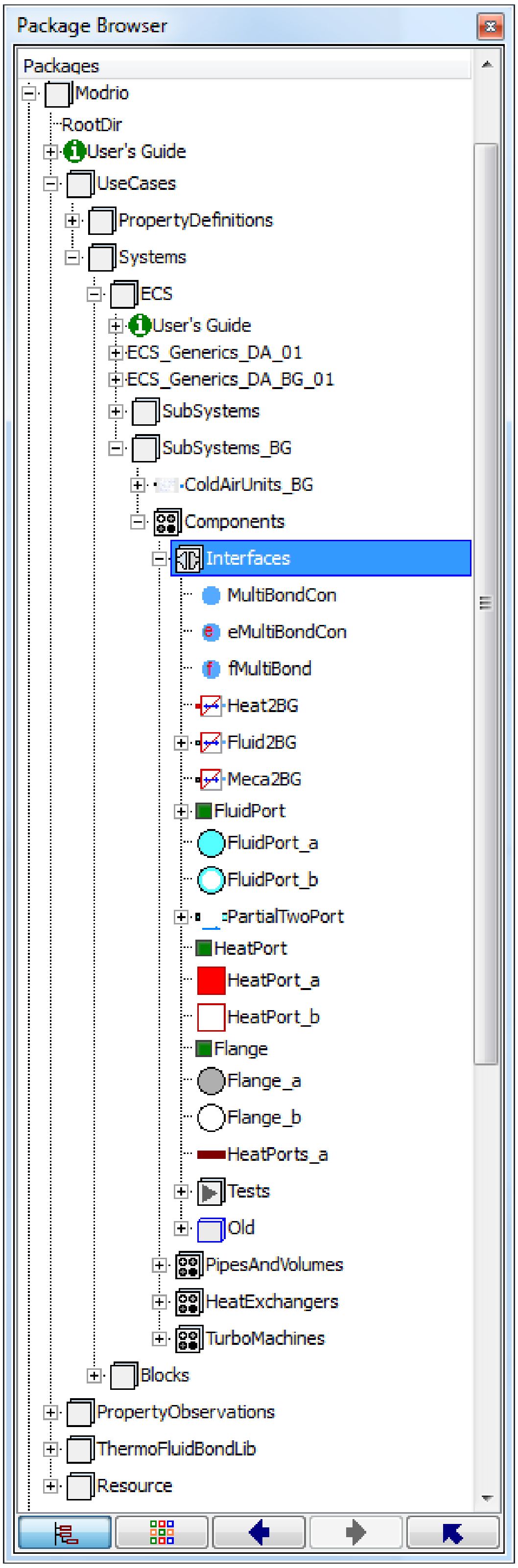
```
connector FluidPort
  "Interface for quasi one-
dimensional fluid flow in a piping network (incompressible or compressible, one or more phases, one or more s
ubstances)"

  import SI = Modelica.SIunits;

  replaceable package Medium = Modelica.Media.Interfaces.PartialMedium
    "Medium model" annotation (choicesAllMatching=true);

  Medium.AbsolutePressure p "Thermodynamic pressure in the connection point";

  Medium.Temperature T "Thermodynamic temperature in the connection point";
  //flow
  Medium.MassFlowRate m_flow
    "Mass flow rate from the connection point into the component";

  //flow
  SI.Power H_inflow
    "Thermodynamic enthalpy mass flow rate close to the connection point if m_flow < 0";

end FluidPort;
```

MultibondLib bondgraph main connector:

```
connector MultiBondCon "bi-directional bondgraphic connector"
  parameter Integer n=1 "Cardinality of Bond connection";
  Real e[n] "Bondgraphic effort variable";
  //flow
  Real f[n] "Bondgraphic flow variable";
  Real d "Directional variable";

end MultiBondCon;
```

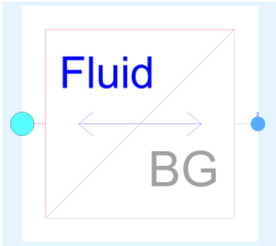
8.2.1.4 Interfaces between physical and Bond graph connectors

To be defined



Components make the interface connections between internal bond graph connectors (from MultibondLib) and the “pseudo” bond graph connectors:

For fluid connector

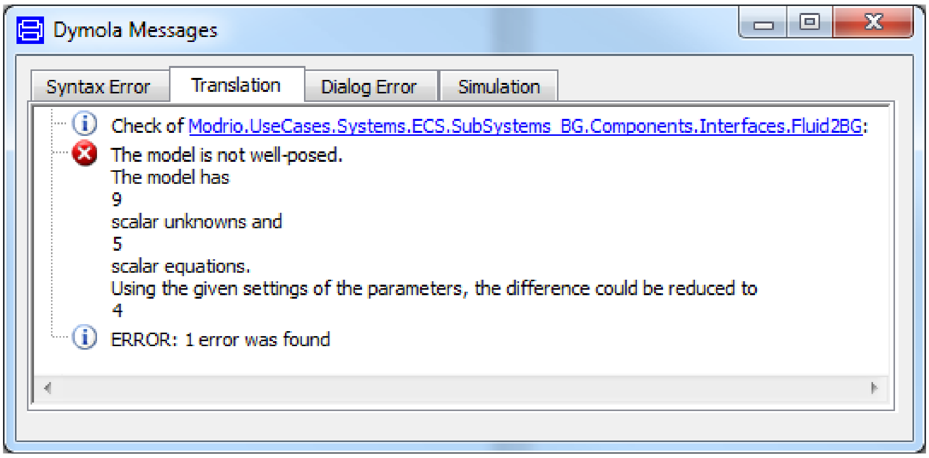


- Code:

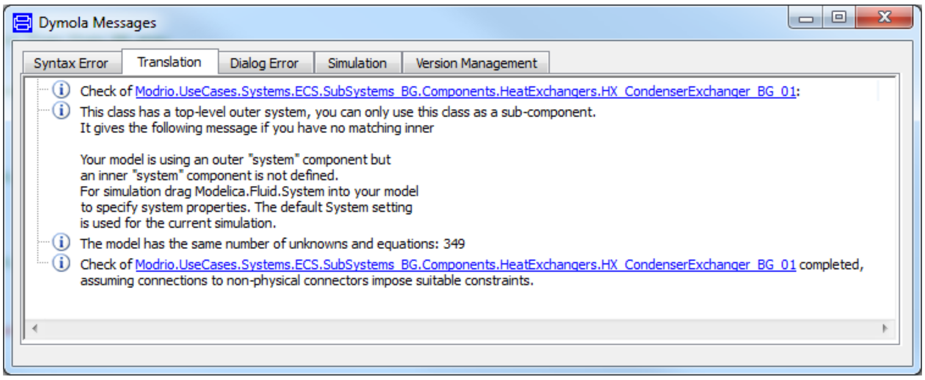
```
model Fluid2BG "acausal Fluid(P,T) to/from bond graph conversion"

  import SI = Modelica.SIunits;

  replaceable package Medium = Modelica.Media.Interfaces.PartialMedium
    "Medium in the component" annotation (choicesAllMatching=true);

  Modrio.UseCases.Systems.ECS.SubSystems_BG.Components.Interfaces.MultiBondCon
    MultiBondCon1(final n=2) "Bond graph connector" annotation (Placement(
        transformation(extent={{90,-10},{110,10}}, rotation=0)));

  FluidPort_a port_a(redeclare package Medium = Medium)
    "Thermal fluid connector" annotation (Placement(transformation(extent={{-110,
          -10},{-90,10}}, rotation=0)));

  // parameter Integer n=1 "Cardinality of Bond connection";
  // Real e[n] "Bondgraphic effort variable";
  // Real f[n] "Bondgraphic flow variable";
  // Real d "Directional variable";

  // Medium.AbsolutePressure p
  // Medium.SpecificEnthalpy h
  // flow Medium.MassFlowRate m_flow
  // flow SI.Enthalpy H_inflow

equation
  MultiBondCon1.e[1] = port_a.p;
  MultiBondCon1.e[2] = port_a.T;
  MultiBondCon1.f[1] = -port_a.m_flow;
  MultiBondCon1.f[2] = -port_a.H_inflow;

  MultiBondCon1.d = -1;

end Fluid2BG;
```

- Current issues

Components to transform physical variables to bond graph variables have physical connectors without type flow to be compatible with the MultibondLib library. Individually, they don't check because there are too many variables.



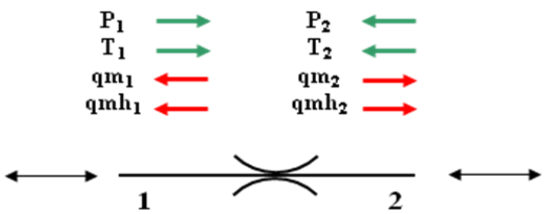
Even if check of component which integrates previous interfaces is true, it is not completely correct. It is not really important because the library is only developed for test, but it is not satisfactory.



8.2.1.5 Pressure loss

A pressure loss corresponds to the momentum balance equation. It gives relations between boundary variable (pressures, temperatures) and average mass flow rate going through the component.

- Causal boundary variables:



- Equations

For compressible media (like dry air), it is often given by the following equation:

$$qm_{12} = f(P_1, P_2, T_1, T_2)$$

The mass flow rate can be blocked if the flow is chocked (Mach = 1).

The pressure drop used with the components of the CAU is given by the following formula valid when Mach < 0.3:

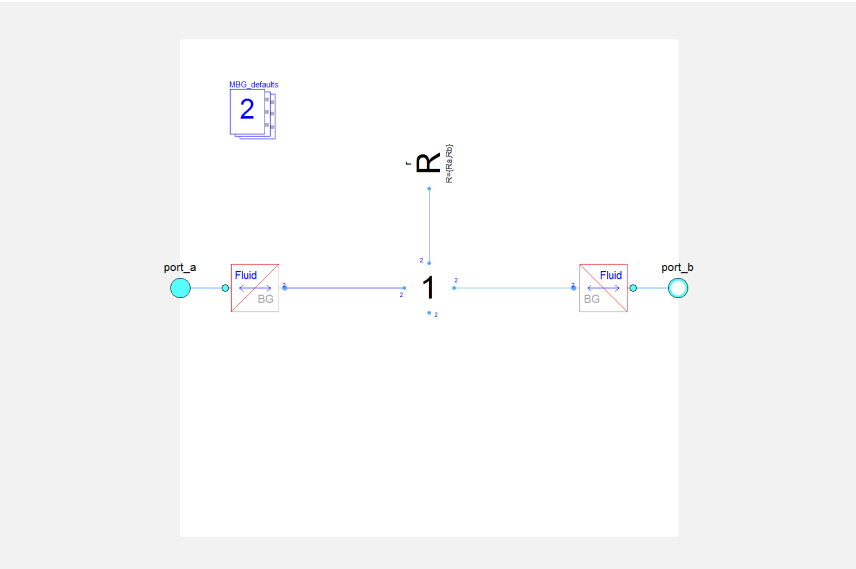
$$\left|P_2^{\,2} - P_1^{\,2}\right| = 2.K.qm^{\alpha}.Tm$$

With $Tm = \dfrac{T_1 + T_2}{2}$ or $Tm = T_{up}$

And K is the pressure drop coefficient, which is different when the flow is laminar or turbulent. α is a constant coefficient close to 2.

- Bond graph representation

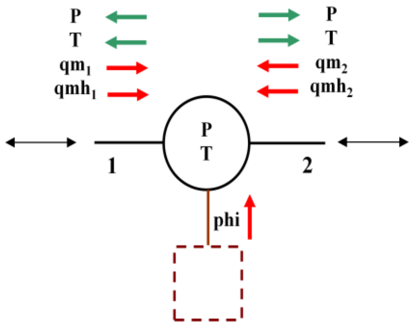
  The bond graph representation of the pressure drop encapsulated in a Modelica component, also called Static Pipe, is represented in the following figure. The external ports of the component are Modelica connectors



8.2.1.6 Volume

A volume is used to calculate the mass and energy balances. It gives static variables pressure (P) and temperature (T) within the volume.

- Causal boundary variables:



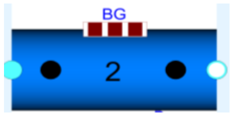
- Equations
  - Mass balance: $$\frac{dm}{dt} = qm_2 + qm_i$$

    m is the mass within the volume
  - Energy balance: $$\frac{dU}{dt} = qmh_2 + qmh_i + \Phi$$

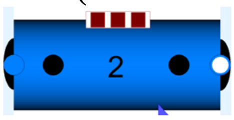
    U is the total internal energy within the volume

8.2.1.7 Dynamic pipes

- Icon of the causal model



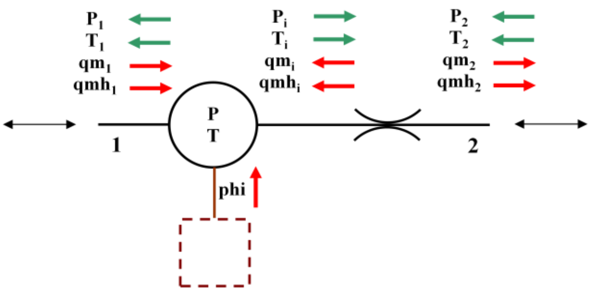
- Icon of corresponding acausal model (similar to which of Modelica.Fluid):



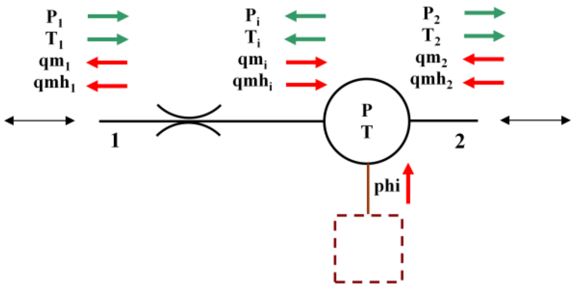
According to the Finite Volume method, pipes can be in cells which can have the behavior split in volume and resistance elements. To fulfill the three balance equations of a cell, it is sufficient to describe the pipe with a volume and a pressure drop.

The simple pipe is composed with only one volume and one pressure drop, as defined above.

- Causal boundary variables, illustrated for a simple pipe:
  - Element defined with a volume and a resistance (if with consider that the nominal flow is from port 1 to port 2):



- Element with a resistance and a volume resistance (if with consider that the nominal flow is from port 1 to port 2):

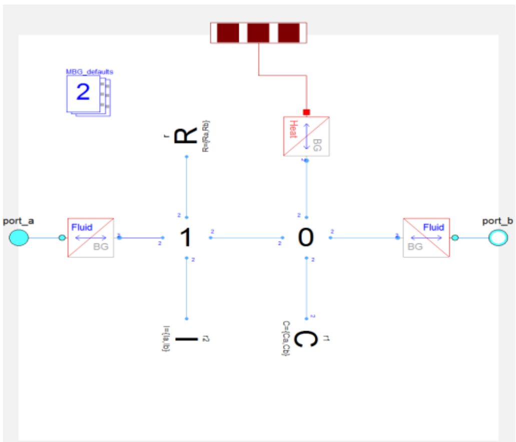


Remarks:

- ♦ Pipes may have reverse flow during a simulation.
- ♦ In an acausal world, these components would be the same. Connections and simulation ability depends of the internal structure of the component.

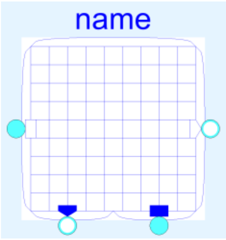
- Bond graph representation

  The bond graph representation of the pressure drop encapsulated in a Modelica component, also called Static Pipe, is represented in the following figure. The external ports of the component are Modelica connectors

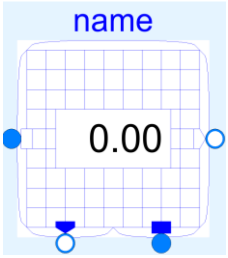


8.2.1.8 Heat Exchangers

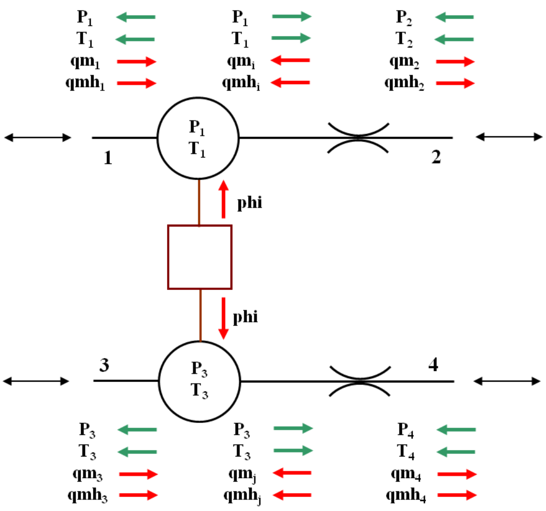
- Icon of the causal model (condenser heat exchanger)



- Icon of corresponding acausal model:



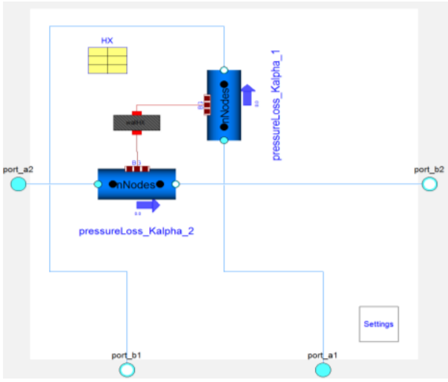
- Causal boundary variables, illustrated with a simple heat exchanger composed of simple pipes and a simple heat transfer:



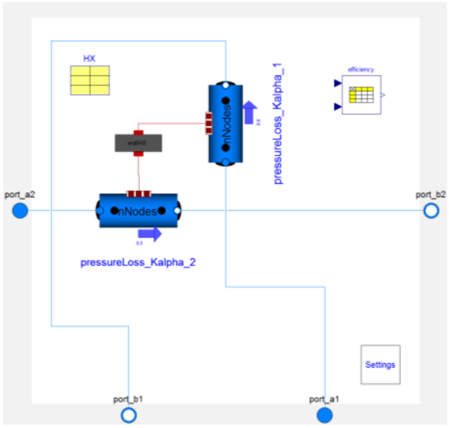
- Equations

$$Tc_s - Tc_e = \varepsilon.(Tf_e - Tc_e)$$

- Internal composition of the causal model



- Internal composition of the acausal model

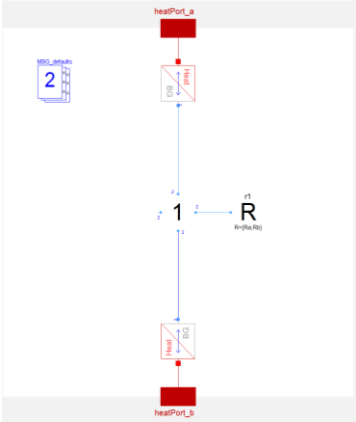


- Heat transfer component

  The bond graph representation of the heat transfer between pipes encapsulated in a Modelica component is represented in the following figure. The external ports of the component are Modelica connectors

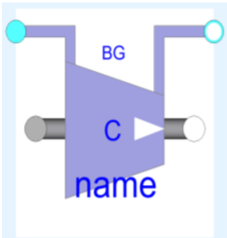
  Icon:

Internal composition:

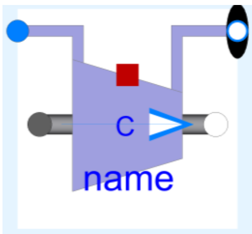


8.2.1.9 Compressor

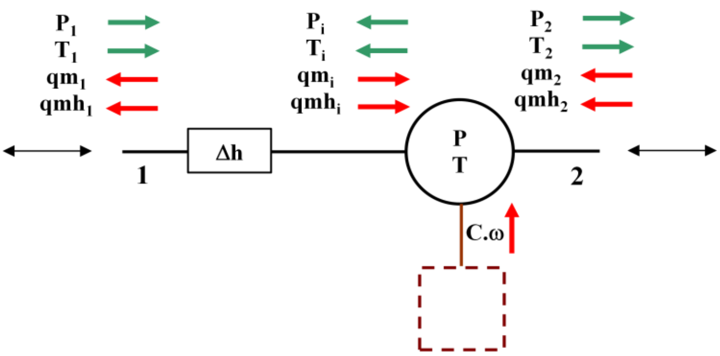
- Icon of the causal model



- Icon of the corresponding acausal model:



- Causal boundary variables for a simple compressor:



- Equations
  - Assumptions :
    - No internal leaks
    - No external heat transfer
  - Equations
    - Mass balance

$$\frac{dm}{dt} = qm_2 + qm_i$$

    - Energy balance

$$\frac{dU}{dt} = qmh_2 + qmh_i + C.\omega \qquad qmh_2 + qmh_i = -qm_1 \cdot \Delta h \qquad \Delta h = f(\eta_{is}, h_1, P_1, T_1, P_2)$$

    - Mass flow rates

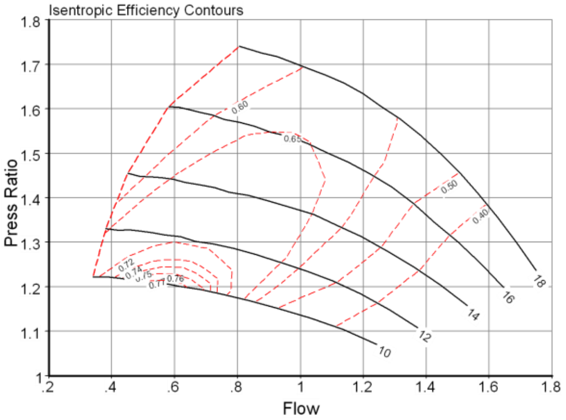
$$qm = qm_2 = f(\omega, P_1, T_1, P_2)$$

    - Efficiency (isentropic)

$$\eta = f(\omega, P_1, T_1, P_2)$$

    - Pressure ratio

$$\frac{P_2}{P_1} = f(qm, \omega, T_1, P_1)$$



8.2.1.10 Turbine

To be defined

8.2.1.11 Control system

To be defined

8.2.1.12 Cabin

To be defined

8.2.1.13 Media

To be defined … $u = f(P,T)...$

8.2.1.14 Assumptions and real system

- Differences between static and stagnation characteristics

To be defined

8.3 FPP extract

Work Package 6: Modeling and simulation services (LIU)
WP6 start date: M0 WP6 end date: M39
Objectives and expected results: The new functionalities added in WP2, WP3, WP4 and WP5 will result in larger models, a priori more difficult to build and simulate. An important part of the problem should be solved by the ability to separate the different operating modes of a given component, resulting in enhanced modular modeling methods (WP4 and WP7). The goal of WP6 is to provide the complementary methodologies (WP6.1, WP6.4, WP6.5, WP6.7) and the computing means (WP6.2, WP6.3, WP6.6) to assist the modeler in tackling larger models.

WP 6.1: Static model analysis (CNRS-Ampère)	
Detailed activity: Extend the techniques already in use for the structural analysis of mechatronics systems to perform static analysis and diagnostics on physical models in order to state whether they are well-posed and suitable for each step of the system's lifecycle. The analysis will localize the problem in the model and will give the physical interpretation of the error. The objective is not to be able to diagnose all possible physical errors, but to add another type of analysis in order to improve error detection before running the model. Coupled with advances in the field of Synchronous modelling, structured approaches will be used to statically analyse systems, and provide useful information to users before models are actually run in order to prevent modelling errors frequently made by users of unstructured solutions. **Starting point:** Dynamic and energy sizing methodology of mechatronic systems based on inverse modelling and bond graph technology developed by CNRS-Ampère. LMS Imagine contribution is based on its OPENPROD project's results and its internal developments related to bond graph static model analysis. Preliminary research on structured approaches coupled with advances in the field of Synchronous modelling: • Enforcing Model Composability in Modelica, Modelica Int' Conference'2009 • Enforcing Reliability of Discrete-Time Models in Modelica, Modelica Int' Conference'2011	
Partner	**Contribution**
CNRS-Ampère	Starting from an industrial process (e.g. a power plant) and the application of the tools developed at laboratory Ampère for system design, the contribution will be to extend these tools and to set up a methodology that helps analyse the model requirements and improve/correct the existing models in view of their use during system operation. The methodology will be developed in the context of the common environment basis of the project (Modelica, FMI...).
EDF	Provides requirements for static model analysis and evaluates results.
LMS Imagine	AMESim's static model analysis capabilities study and software mockup development (experimentation).
Dassault Aviation	Provide requirements and assessment of WP results. The WP results will be tested in the demonstrators (WP8.4).

Date	PU/CO	Deliverable
M12 M39	PU	D6.1.1 – Requirements for static model analysis (Dassault-Aviation, EDF)
		Document which will describe needs for static model analysis for business aircrafts and power plants.
M18 M39	PU	D6.1.2 – Intermediate and final report static model analysis (Dassault-Aviation, EDF)
		Document which will describe results of the WP for business aircrafts and power plants.
M18 M39	CO	D6.1.3 – Static model analysis module based on bond graph and synchronous approaches (LMS Imagine)
		Prototype software component developed upon the D6.1.4 recommendations, interfaced with AMESim to check structural consistency of system models and to deliver user-friendly error messages based on the physical properties of models.
M18 M39	PU	D6.1.4 – Methodology and specification report for static model analysis (CNRS-Ampère, LMS Imagine)
		The report will describe the theoretical material for the concepts and the tools of the developed methodology (physical structural analysis, model inversion...). It will clearly give the different steps to undertake the model analysis and furnish a guide for a system operator to take operating decisions. Finally, the report will specify the project environment tools to implement the developed methodology.