The Dark Side of Object-Oriented Modelling: Numerical Problems, Existing Solutions, Future Perspectives

Francesco Casella

(francesco.casella@polimi.it)

Dipartimento di Elettronica e Informazione

Politecnico di Milano

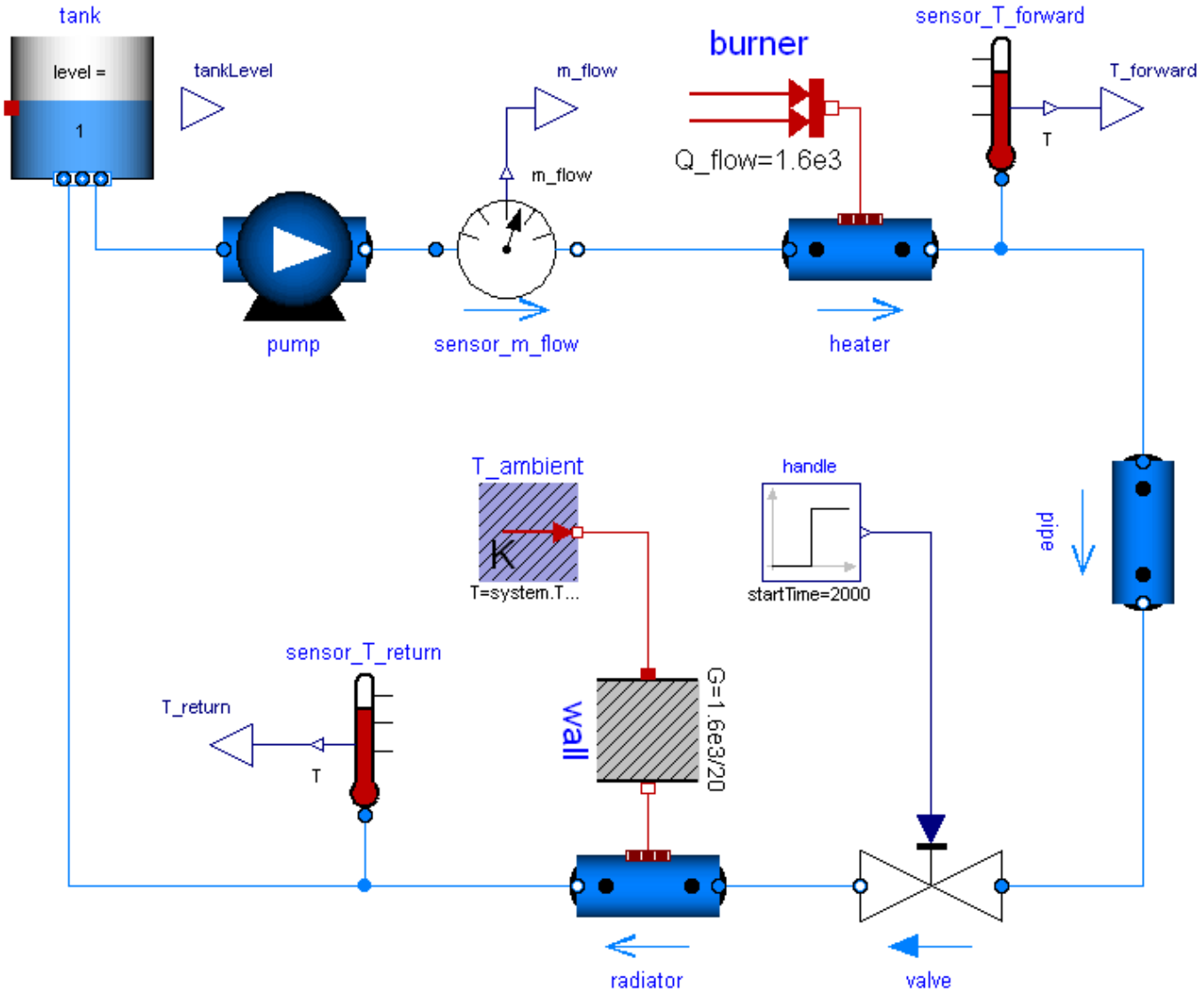
OpenOffice.org

Introduction

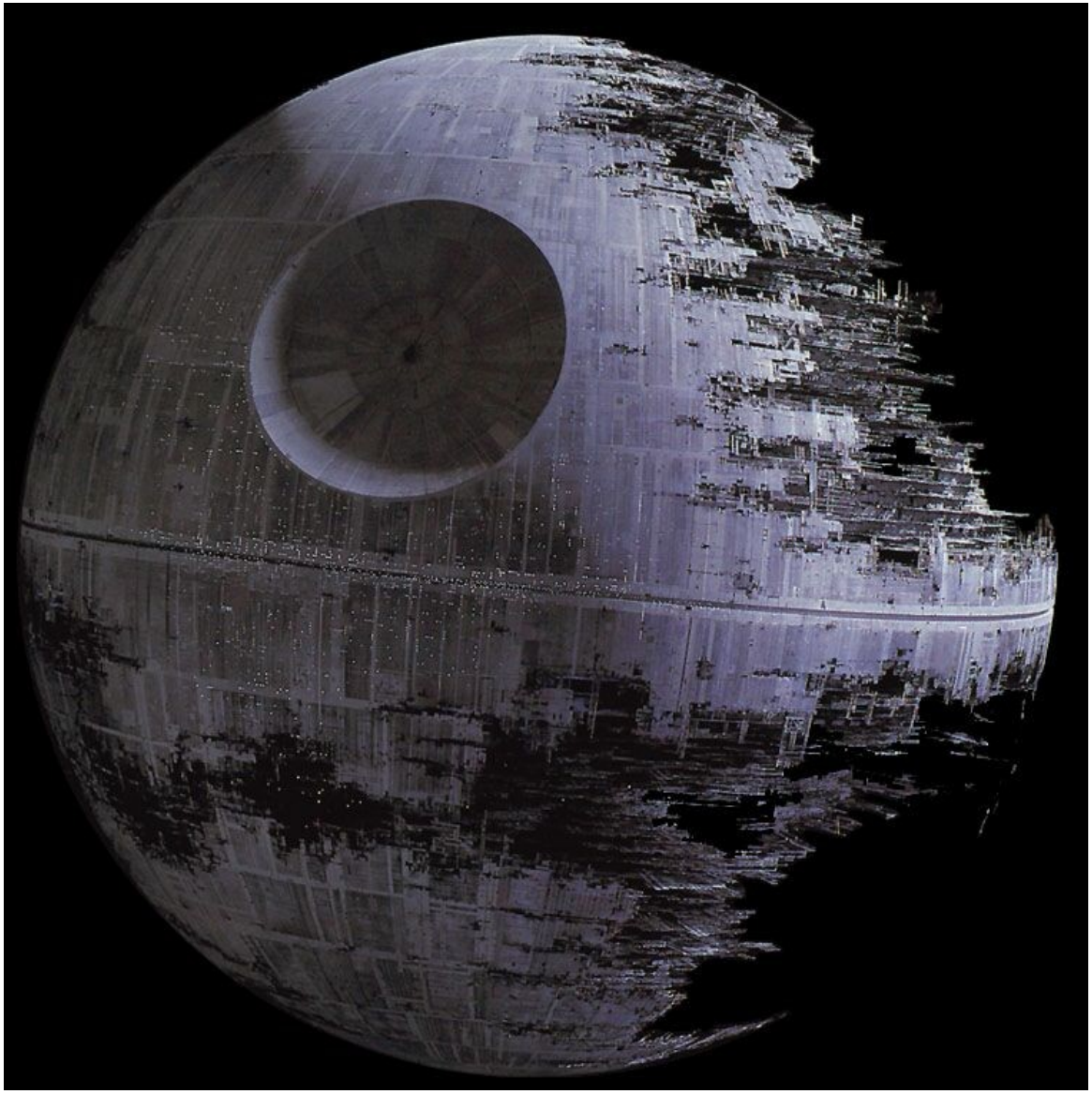
- Equation-Based Object-Oriented Modelling (OOM) approach well established in industry and academia
- Modelica Language emerging
- A-causal approach (write equations, not how they are solved) makes fully modular modelling of multi-domain physical system possible
- Hierarchical modelling, handling of complex systems
- Replaceable models, handling of reconfigurable systems
- Fancy GUIs, model management tools, ...

but...

A (not so nice) anecdote



The dark side of O-O modelling



Numerical Errors in OOM: A Taboo Topic?

- *All users* of OOM encounter this kind of problems
- High-level expertise in simulation techinques required for troubleshooting
- Domain experts use the model!
- These errors are a blocker (no result obtained at all unti. solved)
- They can scare off people from OOM technology

Numerical Errors in OOM: A Taboo Topic?

- *All users* of OOM encounter this kind of problems
- High-level expertise in simulation techinques required for troubleshooting
- Domain experts use the model!
- These errors are a blocker (no result obtained at all unti. solved)
- They can scare off people from OOM technology

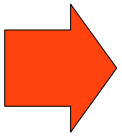
- People cope somehow, eventually
- Fact never mentioned in technical or scientific literature
  (something not worth mentioning, or even to be somwhat ashamed of?)

- Little progress in solving these problems in a sound and systematic way
- Error handling and debugging techniques much less developed
  than simulation techniques

Numerical Problems in OOM Simulation

Numerical Problems in OOM Simulation

OO Model
(Modelica)



Dynamic equations
System behaviour

$$F\left(x, \dot{x}, v, p, t\right) = 0$$

Initial equations
Initial states
& unknown parameters

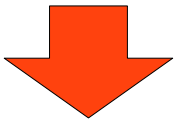
$$F_i\left(x, \dot{x}, v, p\right) = 0$$

Simulation Problem

- At each time step,
  solve $F(x, \dot{x}, v, p, t) = 0$ given x, p, t
- Behaviour equations
  - well-tested model library
  - physically meaningful connection equations
- Known states problem broken into many small subproblems
- Advanced numerical/symbolic techniques used for efficient solution
- Modelling errors wrong behaviour wrong trajectories

Simulation Problem

- At each time step,
  solve $F(x, \dot{x}, v, p, t) = 0$ given x, p, t
- Behaviour equations
  - well-tested model library
  - physically meaningful connection equations
- Known states problem broken into many small subproblems
- Advanced numerical/symbolic techniques used for efficient solution
- Modelling errors wrong behaviour wrong trajectories



Errors can be investigated
by domain-expert modeller

Initialization problem

- At time $t = 0$, solve $F\left(x, \dot{x}, v, p, 0\right) = 0$

$$F_i\left(x, \dot{x}, v, p\right) = 0$$

- Typically large systems of highly non-linear equations
- In case of solver errors: no trajectories available for inspection

Initialization problem

- At time $t = 0$, solve $F\left(x, \dot{x}, v, p, 0\right) = 0$

$$F_i\left(x, \dot{x}, v, p\right) = 0$$

- Typically large systems of highly non-linear equations
- In case of solver errors: no trajectories available for inspection

Error scenarios

(1) Well-posed problem, solver fails

(2) Underconstrained/overconstrained problem

(3) System-level singular problem, solver fails

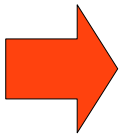
(4) Structurally well-posed problem, wrong parameters (no solution)

(1) Well-Posed Problem, Simulation Fails

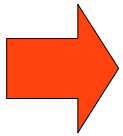
- Nonlinear solver needs initial guess for all iteration variables
- Providing close enough values often unfeasible or very tedious
- Can't tell whether the guess values are wrong, or the problem has no solution at all

(1) Well-Posed Problem, Simulation Fails

- Nonlinear solver needs initial guess for all iteration variables
- Providing close enough values often unfeasible or very tedious
- Can't tell whether the guess values are wrong, or the problem has no solution at all



Homotopy-based initialization



High-level debugging

Homotopy-Based Initialization

- Define a simplified problem which is easier to solve

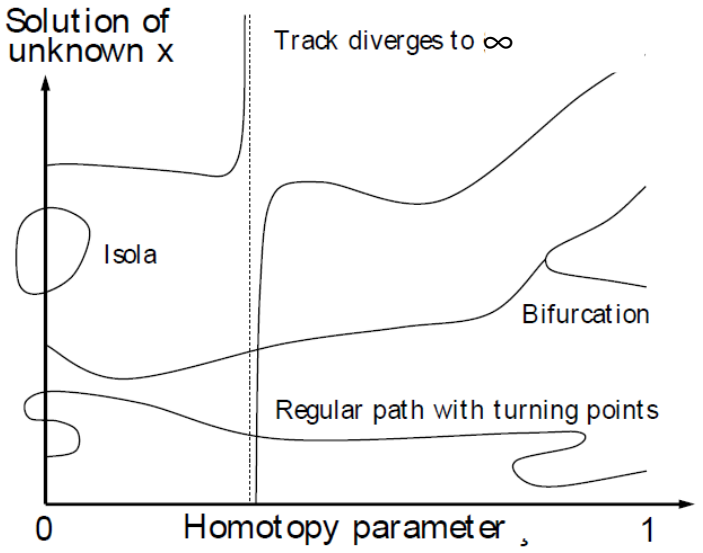
$$F_s(x) = 0$$

- Continuously transform into the actual problem

$$(1 - \lambda)\, F_s(x) + \lambda\, F_a(x) = 0$$

- New Modelica operator **`homotopy`**`(actual, simplified)` introduced in 2011

- Beware of singularities!



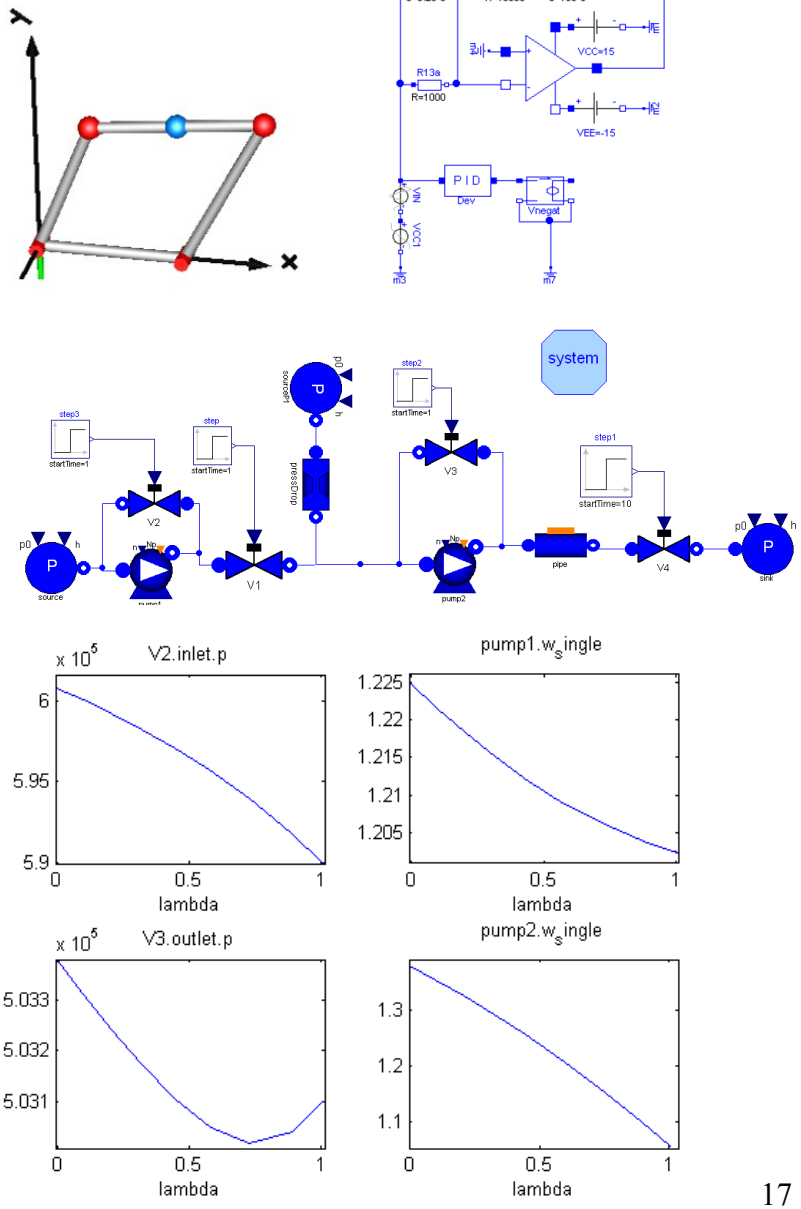
Homotopy-Based initialization – First Results

Modelica Conference 2011
(Casella, Sielemann et al.)

- Multibody systems with multiple configurations
- Analog electronic circuits
- Hydraulic networks
- Calibration of air-conditiong systems
- Large power plants

Modelica Conference 2012
(Sielemann)

- Probability one homotopy



Homotopy-Based initialization – First Results

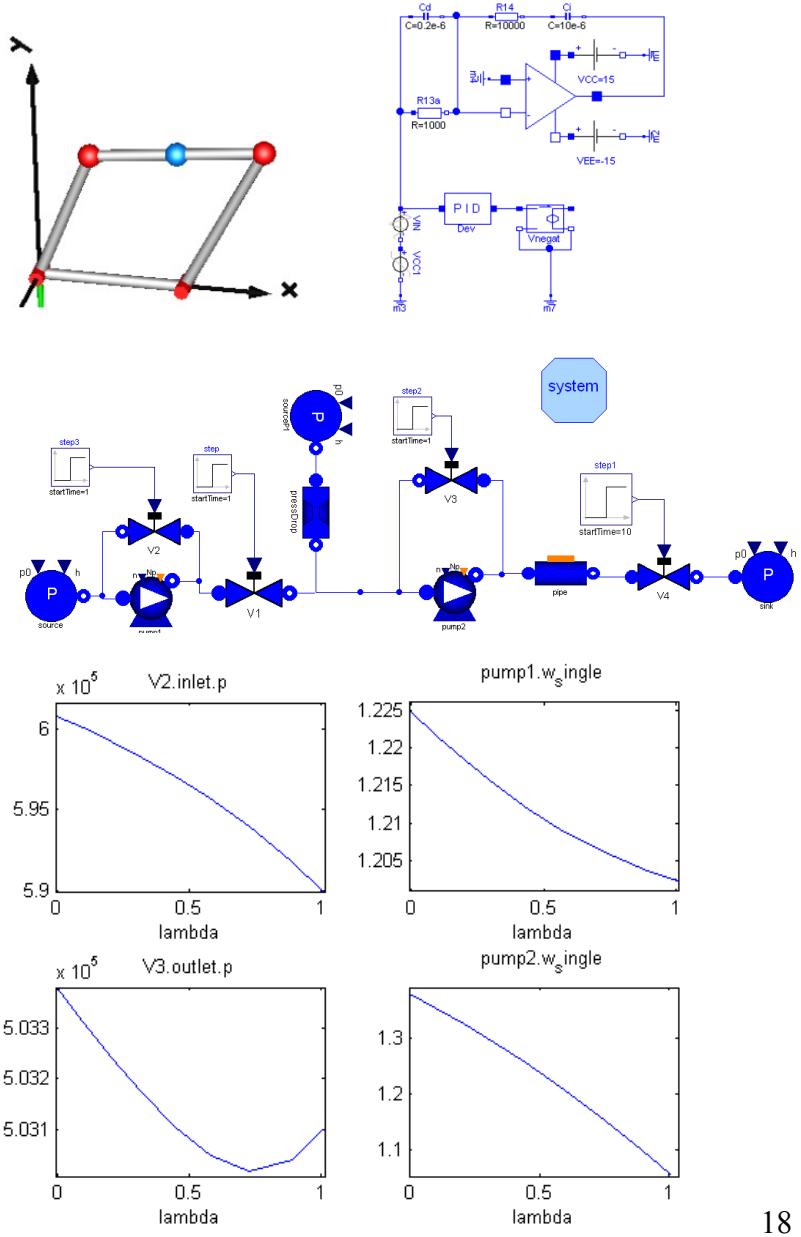
Modelica Conference 2011
(Casella, Sielemann et al.)

- Multibody systems with multiple configurations
- Analog electronic circuits
- Hydraulic networks
- Calibration of air-conditiong systems
- Large power plants

Modelica Conference 2012
(Sielemann)

- Probability one homotopy

Promising approach
more operational experience
needed before getting mature



High-Level Debugging: the Open Modelica Debugger

- Idea #1: keep track of all model transformations in the generated code
- Idea #2: explore this information graphically, by traversing graphs

High-Level Debugging: the Open Modelica Debugger

- Idea #1: keep track of all model transformations in the generated code
- Idea #2: explore this information graphically, by traversing graphs

- The solver fails when trying to solve an equation in the BLT

  or

  A result is found, clearly wrong from a physical perspective

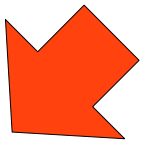
High-Level Debugging: the Open Modelica Debugger

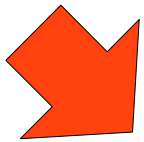
- Idea #1: keep track of all model transformations in the generated code
- Idea #2: explore this information graphically, by traversing graphs

- The solver fails when trying to solve an equation in the BLT

  or

  A result is found, clearly wrong from a physical perspective





Code Dependency Graph

- where does the equation come from in the original model?
- which transformations have been applied?

Variable Dependency Graph

- from which other variables it depends?
- what values they have?

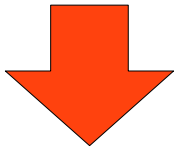
- see paper by Pop et al., Modelica Conference 2012

(2) Underconstrained/Overconstrained problem

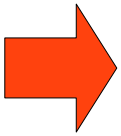
- Initial conditions are usually set at the single component level
- Initialization problem is a system-level problem
  (exp. with control systems and steady-state conditions!)
- Additional initial equations added determine unknown parameters
  (trimming problem)
- Initial equations made redundant in case of index reduction

(2) Underconstrained/Overconstrained problem

- Initial conditions are usually set at the single component level
- Initialization problem is a system-level problem
  (exp. with control systems and steady-state conditions!)
- Additional initial equations added determine unknown parameters
  (trimming problem)
- Initial equations made redundant in case of index reduction



- Often init problem turns out to be underconstrained or overconstrained
- Underconstrained tool adds extra initial equations based on heuristics
  - might not reflect the end user's intention
- Overconstrained the tool asks to remove initial equations
  - not clear which one is “the right one”
  - not clear how to do it on structured model via GUI



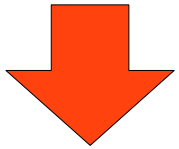
No satisfactory solution available yet

(3) System-level singular problem, solver fails

- The initialization problem can be square but singular for system-level structural reasons
- Examples:
  - Closed thermohydraulic systems with steady-state initial equations
  - Closed hydraulic systems with incompressible fluid
  - Electrical circuits without ground connection
- The Jacobian of the initialization problem equations is singular, no unique solution, solver fails

(3) System-level singular problem, solver fails

- The initialization problem can be square but singular for system-level structural reasons
- Examples:
  - Closed thermohydraulic systems with steady-state initial equations
  - Closed hydraulic systems with incompressible fluid
  - Electrical circuits without ground connection
- The Jacobian of the initialization problem equations is singular, no unique solution, solver fails



- Analysis of the nullspace of the Jacobian can reveal the subset of equations causing the singularity
- Meaningful annotations on key equations which are always part of the singular system can be used for high-level diagnostics to the end user
- Proposed by Casella, Modelica Conference 2012, not yet implemented

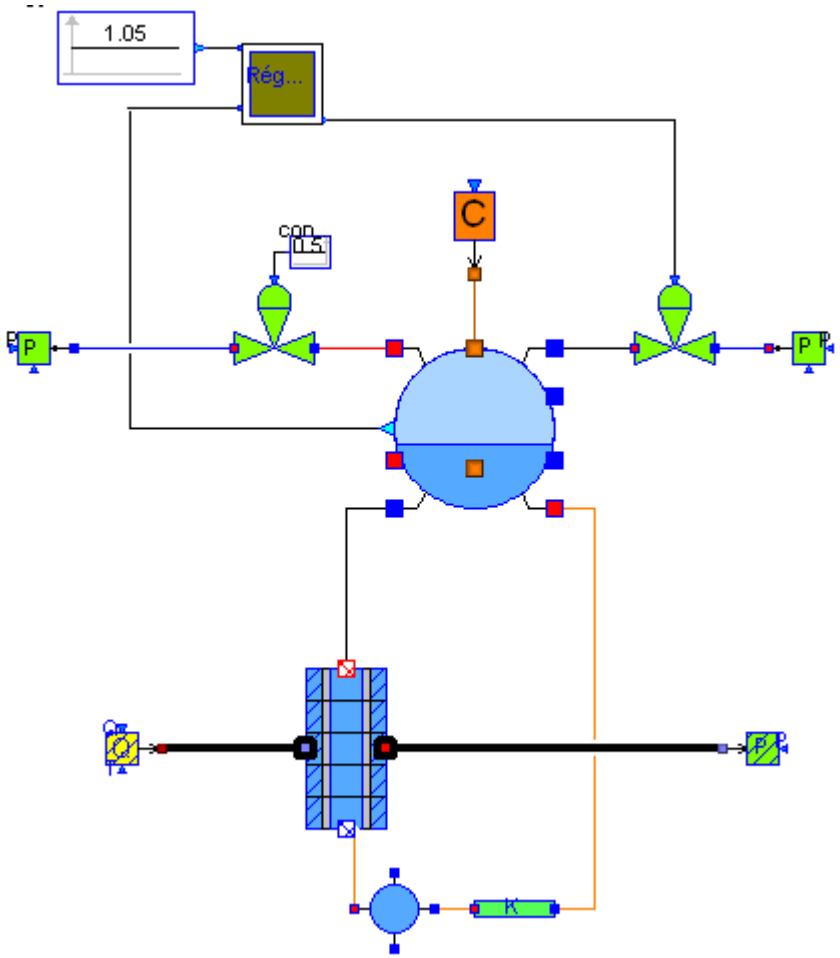
(4) Structurally well-posed problem, wrong parameters

- The strange case of the steam generator
- Steam valve Cv too small
- No steady state solution exists!

- Difficult to determine “the” wrong parameter
  - steam valve too small?
  - too much flue gas flow?
  - flue gas too hot?

- Traditional troubleshooting strategy: divide & conquer

- New ideas?



Numerical Problems in OOM Optimization

Dynamic optimization of OO Models

- OO models can be used for dynamic optimization (optimal control) of physical systems
- The OO modelling language is used to write the model DAEs which are the dynamic constraint of the optimization problem
- Extension of Modelica: Optimica (J. Åkesson PhD work)

```
optimization DIMinTime (
     objective=finalTime,
     startTime=0,
     finalTime(free=true,initialGuess=1))
  Real x(start=0,fixed=true);
  Real v(start=0,fixed=true);
  input Real u;
equation
  der(x)=v;
  der(v)=u;
constraint
  x(finalTime)=1;
  v(finalTime)=0;
  v<=0.5;
  u>=-1;
  u<=1;
end DIMinTime;
```

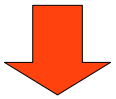
Dynamic optimization of OO Models

- OO models can be used for dynamic optimization (optimal control) of physical systems
- The OO modelling language is used to write the model DAEs which are the dynamic constraint of the optimization problem
- Extension of Modelica: Optimica (J. Åkesson PhD work)

```
optimization DIMinTime (
     objective=finalTime,
     startTime=0,
     finalTime(free=true,initialGuess=1))
  Real x(start=0,fixed=true);
  Real v(start=0,fixed=true);
  input Real u;
equation
  der(x)=v;
  der(v)=u;
constraint
  x(finalTime)=1;
  v(finalTime)=0;
  v<=0.5;
  u>=-1;
  u<=1;
end DIMinTime;
```

Model DAEs are constraints

Very large non-convex optimization problems



Convergence of solver is a major issue (more than in simulation problems!)

Symbolic Manipulation of Equations

for Simulation

- DAEs solved for state derivatives, then time integration
- alias elimination
- BLT ordering
- index reduction / dummy der
- symbolic solution of implicit equations
- tearing of implicit systems
- common subexpression elimination

- scaling of variables (via `nominal` attribute)
- state selection
- inverse annotations
- simplified expressions and homotopy for initialization

for Optimization

- DAEs used as constraints
- alias elimination

- scaling of variables (via `nominal` attribute)
- bounds on variables (via `min`/`max` attributes)

... ?

Consequences on Modelling

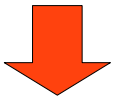
for Simulation

- The modeller can focus on clarity, readability, generality
- The modeller can (almost) ignore how equations will be solved
- Efficient and robust simulation code automatically generated regardless how the equations are written

for Optimization

- The modeller must focus on solvability and convergence issues when writing the equations
- Mathematically equivalent models

completely different optimization problems



completely different convergence properties

~~*Declarative OOM*~~

Example 1

```
optimization Example1A(
  objective = a1/b1*(x1 - x10)^2 +
              a2/b2*(x2 - x20)^2 +
              a3/b3*(x3 - x30)^2,
...
end Example1A;
```

```
optimization Example1B(
  objective = f1 + f2 + f3,
...
equation
  f1 = a1/b1*(x1 - x10)^2;
  f2 = a2/b2*(x2 - x20)^2;
  f3 = a3/b3*(x3 - x30)^2;
...
end Example1B;
```

Example 2

```
optimization Example2A(
  parameter Real PR = 10;
...
equation
  p_in/p_out = PR “Turbine pressure ratio”;
...
end Example1A;
```

```
optimization Example2A(
  parameter Real PR = 10;
...
equation
  p_in = PR*p_out “Turbine pressure ratio”;
...
end Example1A;
```

Research Goals

Investigate structural properties of model DAEs that help NLP solver convergence

Devise structural analysis and manipulation techniques that can transform DAEs into "more favourable" equations

Talk to NLP solver developers so they understand the structure of "our" problems in depth and help us solve them better

Bring the field of OOM Optimization from adolescence (or childhood?) into adulthood

Conclusions

- Object-Oriented Modelling concepts are getting more and more widely used in system engineering and control
- OOM has a huge potential to help closing the gap between theory and practice in optimal control

- Use of OOM for simulation is now fairly mature, but still has some dark areas related to initialization problems that need to be addressed for wider acceptance and use
- Use of OOM for optimization still quite young, comparably little effort spent in automatic (or assisted) streamling of model equations
- Some steps in the right direction have already been made in recent years
- More research is needed to
  - improve the numerical robustness and mitigate convergence problems in tools
  - develop effective and easy-to-use debugging tools for domain experts
  - possibly by embedding expert a-priori knowledge in the modelling code and exploiting it cleverly

Thank you for you kind attention!