Object-Oriented Modelling from a Control Engineer's Perspective: Past, Present, and Future

Francesco Casella

(francesco.casella@polimi.it)

Dipartimento di Elettronica e Informazione

Politecnico di Milano

The control engineer's perspective

Object-Oriented Modelling Tools

Computer-Aided Control System Design Tools

Requirements for control-oriented physical modelling

- *Modularity*
  - modular modelling
  - physical, a-causal interfaces
  - hierarchical system decomposition
- *Flexibility*
  - appropriate level of detail
  - replaceable component and sub-system models
- *Reusability*
  - do not reinvent the wheel
  - capitalize on past modelling investment
- *Generality*
  - multi-domain system modelling
- *Extensibility / Customizability*
  - overcome the “component not available” syndrome
  - including proprietary information
- *Plant-Control Integration*
  - closed-loop Modelica model
  - Modelica plant + external controller (Simulink, HIL, etc)

Common activities in control system design

- Open-loop analysis
  - steady-state operation
  - step responses
  - numerical linearization
- Closed-loop analysis
  - evaluation of different control strategies
  - verification of control system performance
  - empirical tuning of parameters
  - study of operating scenarios
- Identification of uncertain parameters
  - ad-hoc experiments (e.g. friction in motion control applications)
  - optimization-based fitting of transients

The ideal setting

Plant ⟺ Control

Co-Design

Early design stages

↘ Detailed design & engineering

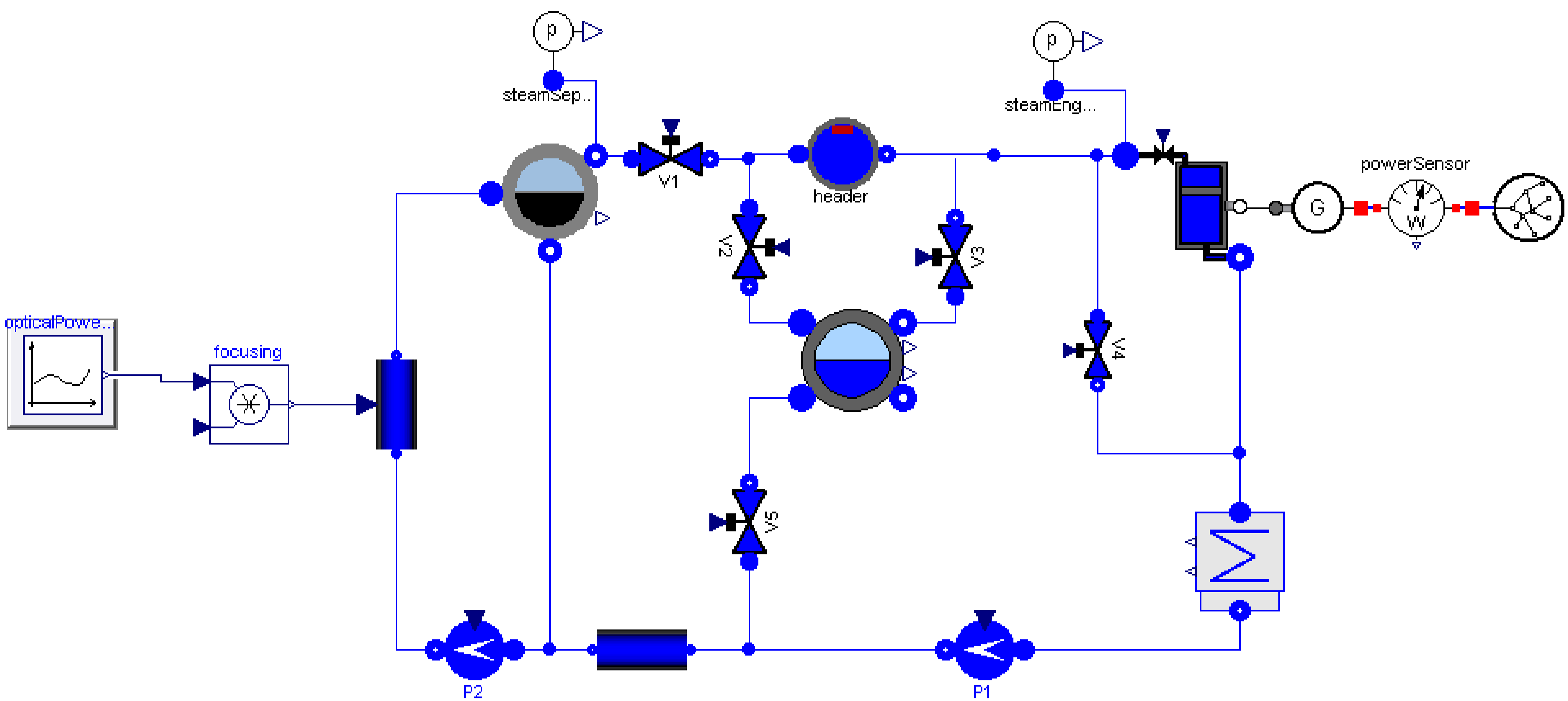
↘ Commissioning

↘ Operation → Revamping

Early design stages

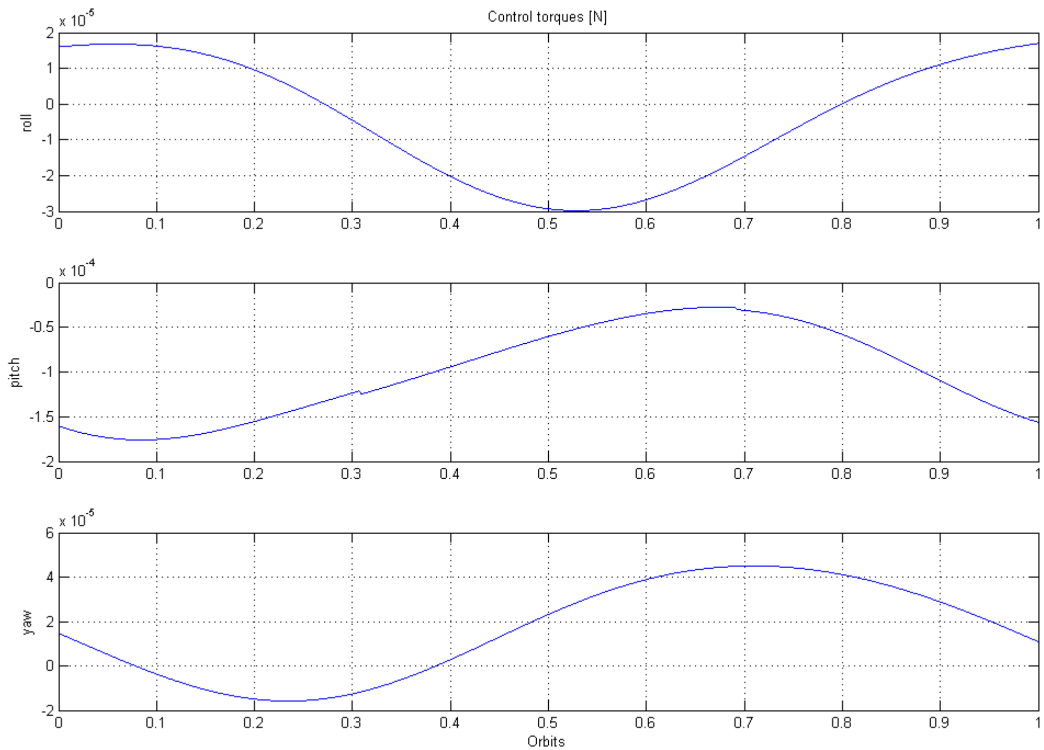
- Few details available
- Evaluation of alternatives
- Closed-loop testing

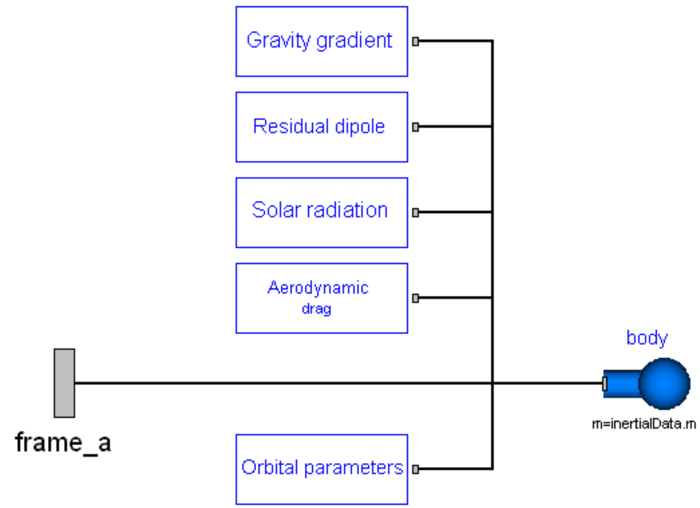
- Example: solar power plant

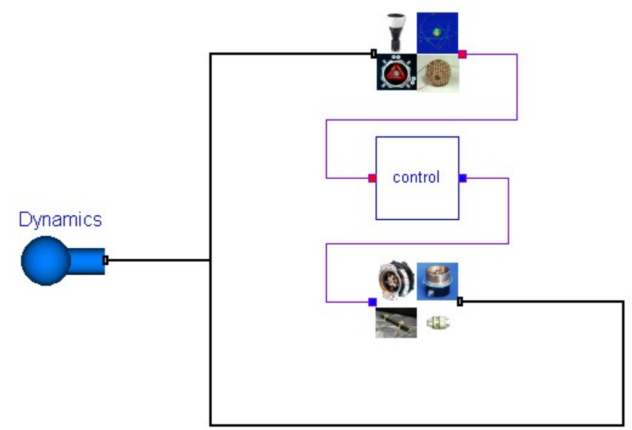


Early design stages – cont'd

- Model inversion for actuator sizing
- Example: satellite attitude control







Detailed design

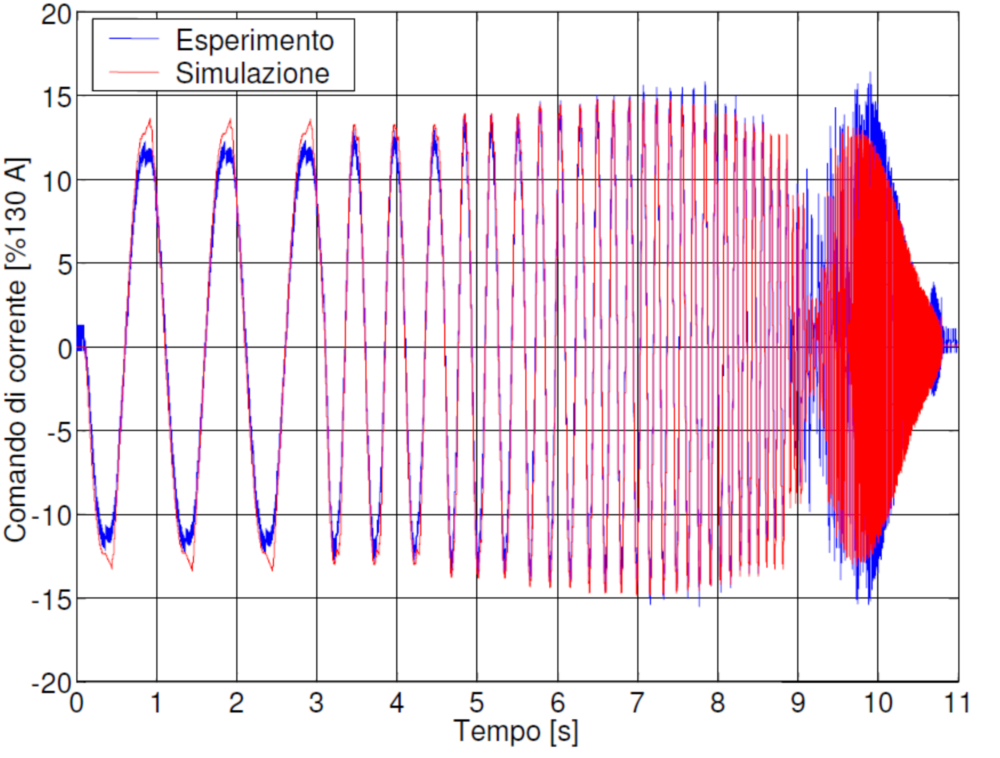
- Refinement of sensors and actuators
  - satellite attitude control
  - pump/valves in thermo-fluid systems

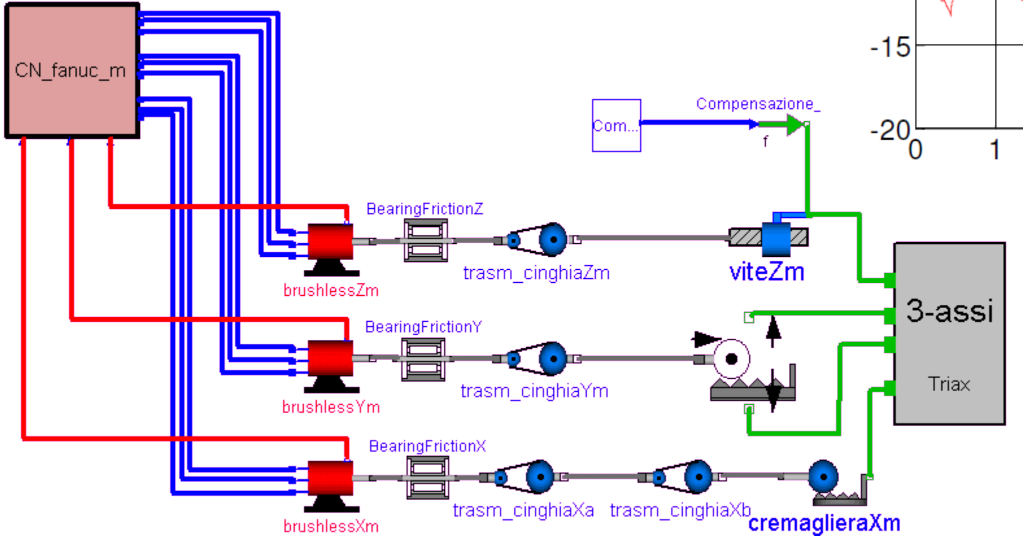
- Refinement of plant model
  - more accurate physical model
  - second-order physical phenomena

- Refinement of control system model
  - digital control algorithms
  - start-up / shutdown / trip handling

Detailed design – cont'd

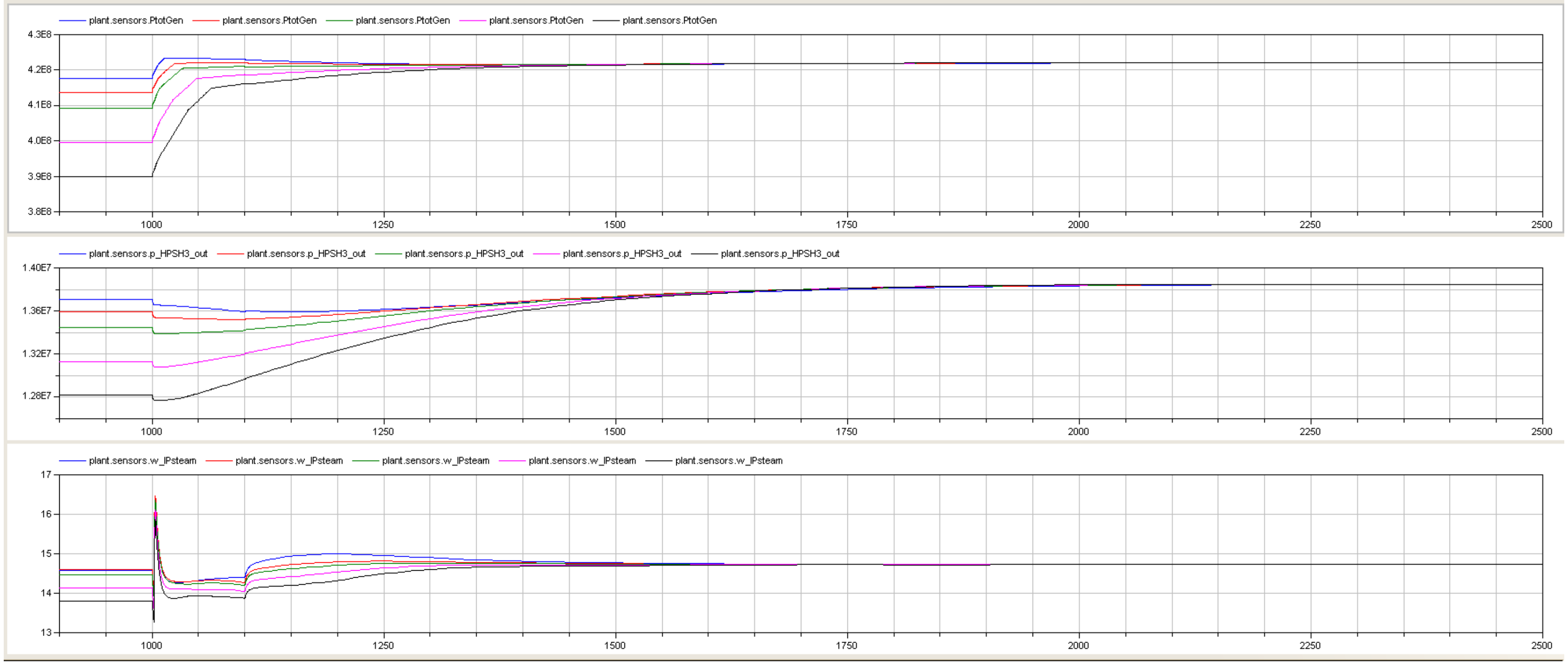
- Evaluation of critical plant dynamics
- Example: machining tool





Commissioning

- Anticipating problems that might arise on-site during commissioning
- Reducing number of tests for CS acceptance
- Help understanding results of tests
- Example: power/frequency control in a 400 MW power plant



How did all of this evolve?

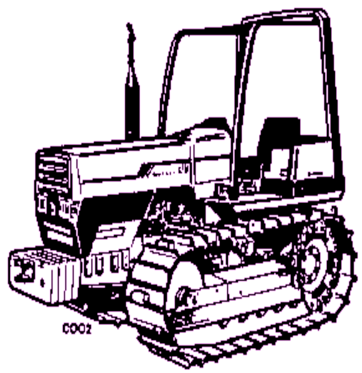
The Past
or
The Heroic '90s

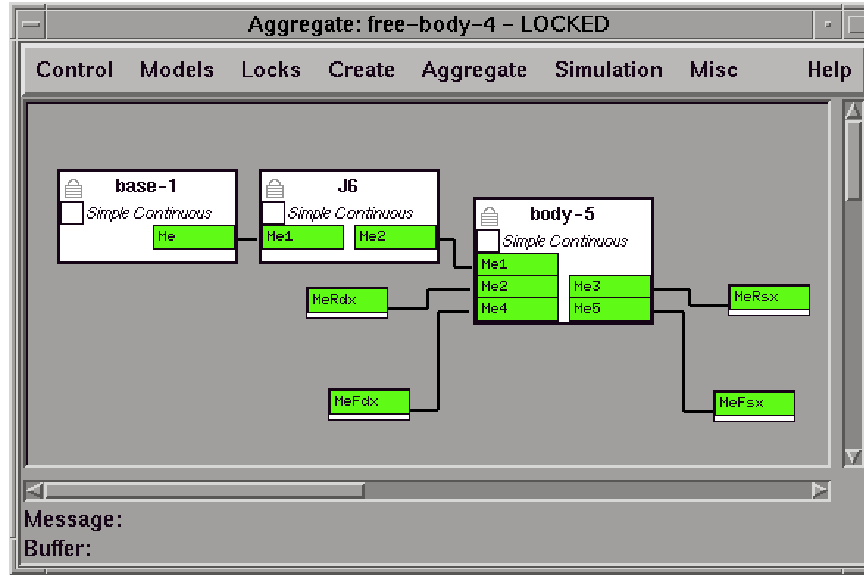
MOSES

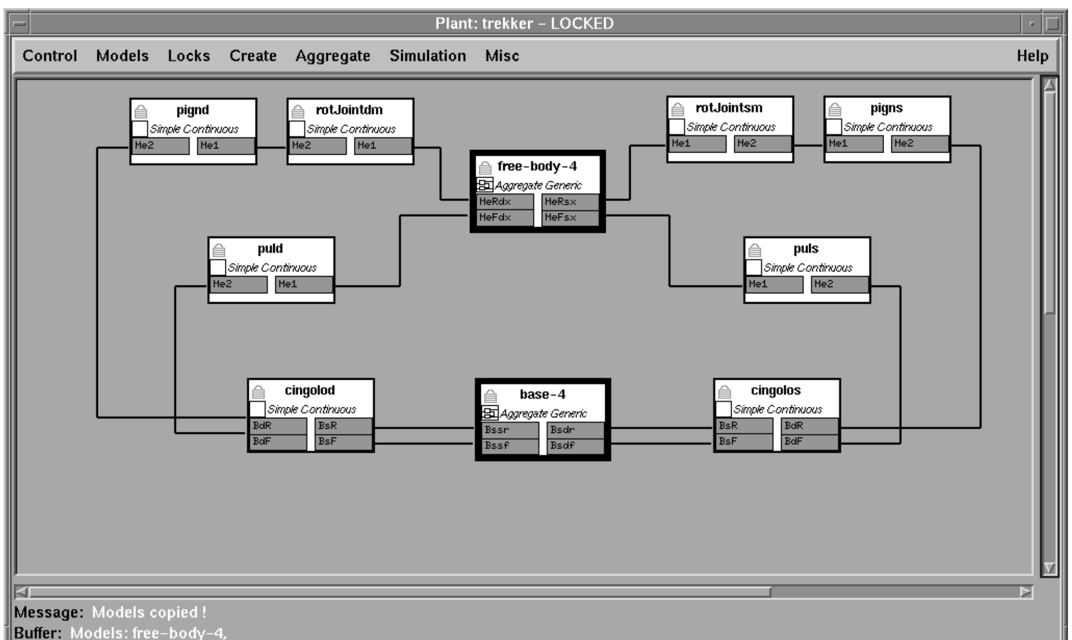
- O-O Modelling tool developed @ Politecnico di Milano 1993-1998
  - effort led by prof. Claudio Maffezzoni
  - Roberto Girelli's PhD thesis
  - contributions by Emanuele Carpanzano and others

- Features
  - Equation-based, O-O models in an O-O database
  - Hierarchical composition
  - Multibody mechanical systems
  - Index reduction and automatic differentiation
  - Tearing
  - DASSL as ODE integrator

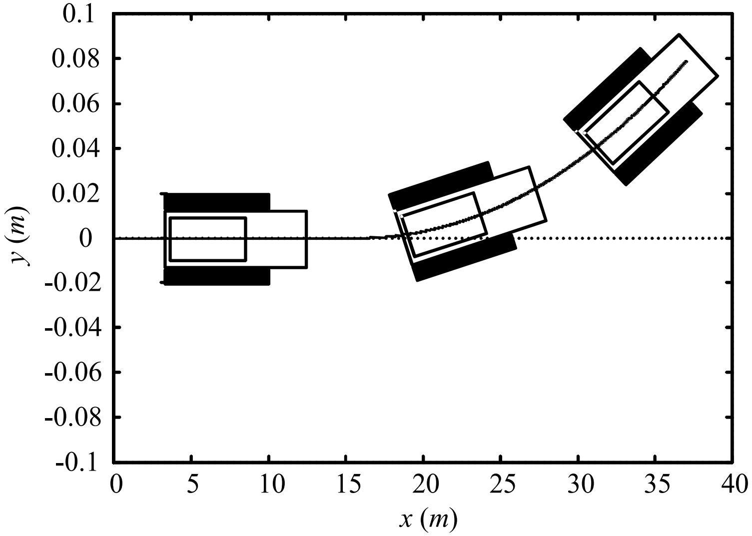
Modelling of an agricultural tracked vehicle

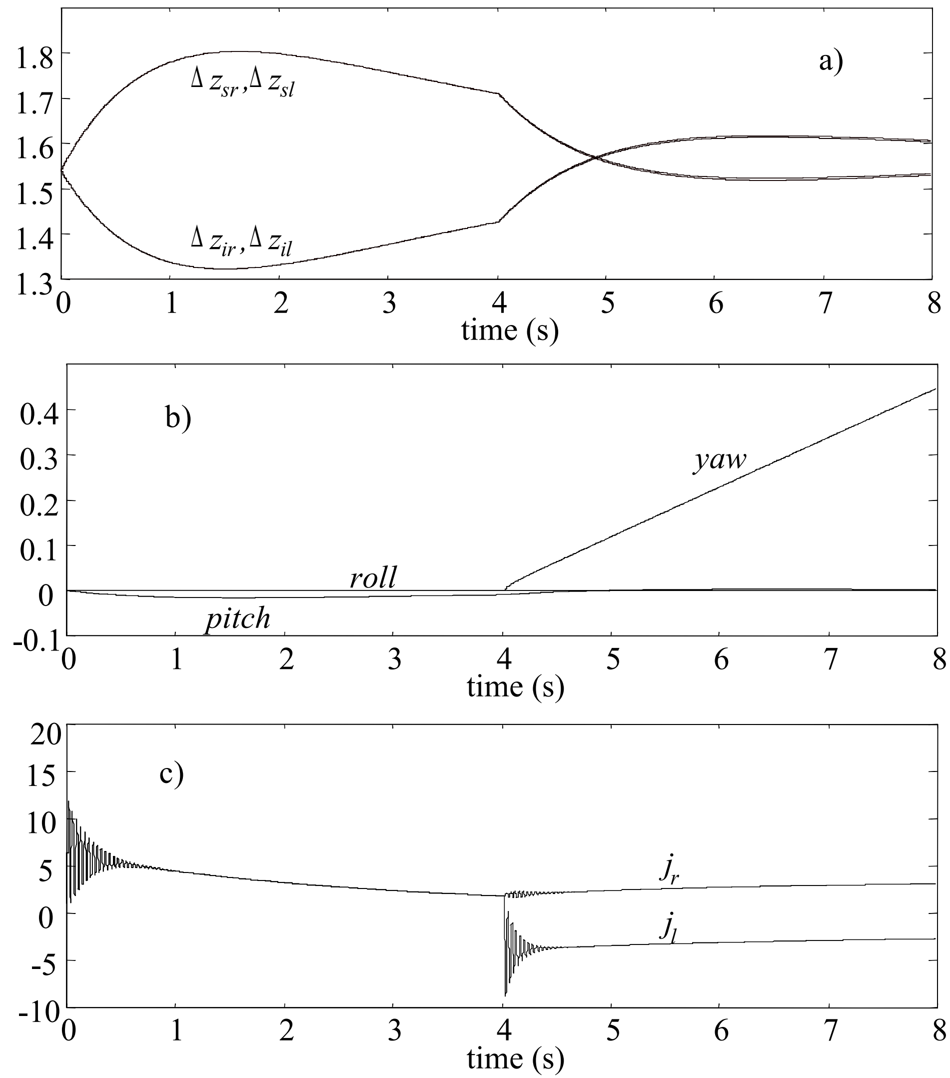






Modelling of an agricultural tracked vehicle – cont'd





The heroic early '90

- Equation-based, Object-Oriented modelling paradigm emerges
- Early tool implementations
- Prototype tools developed mostly in universities
- Proof-of-concept
- Heroic efforts in order to support real-life engineering projects

The Present

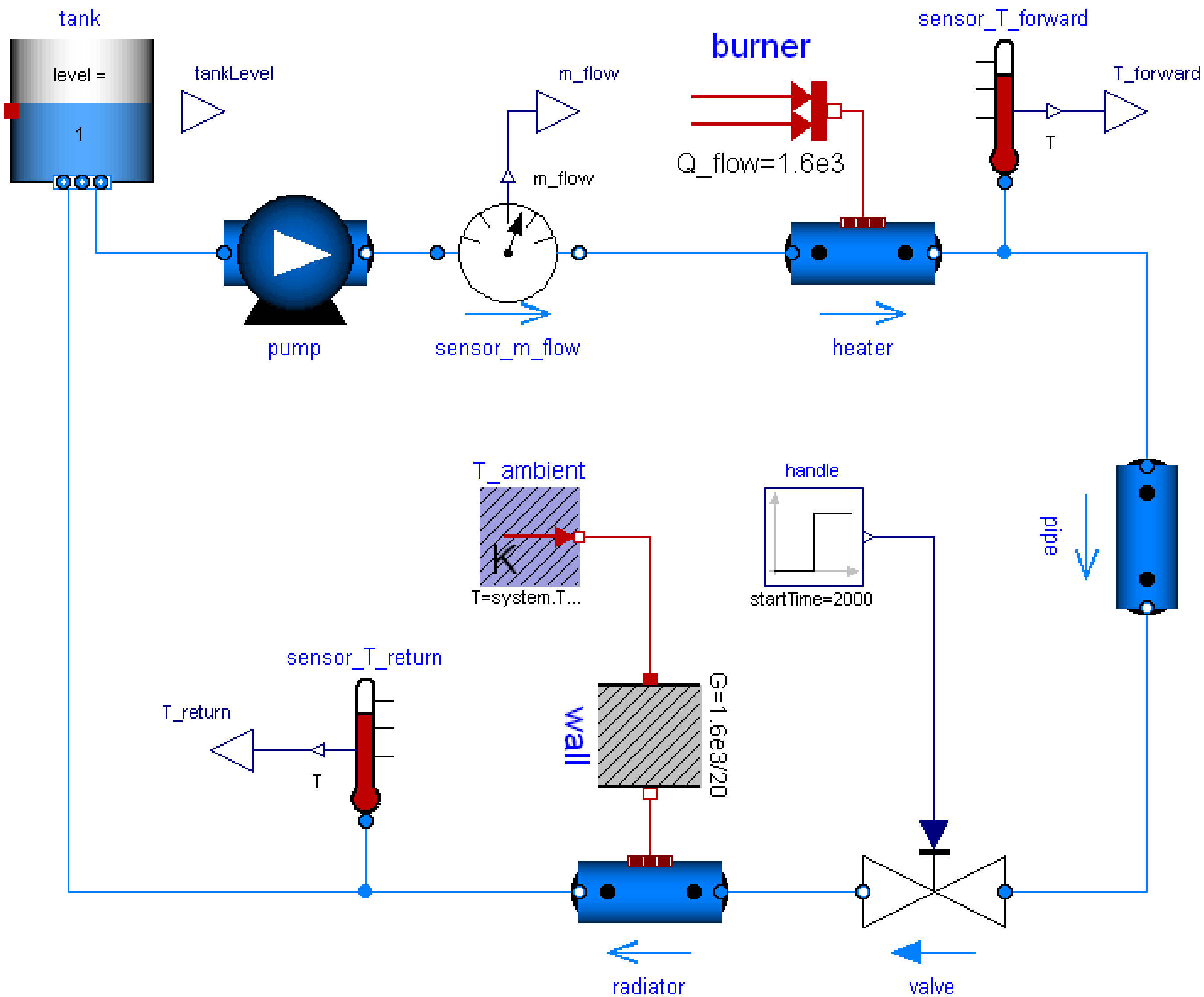
2000-2010

A well-established paradigm

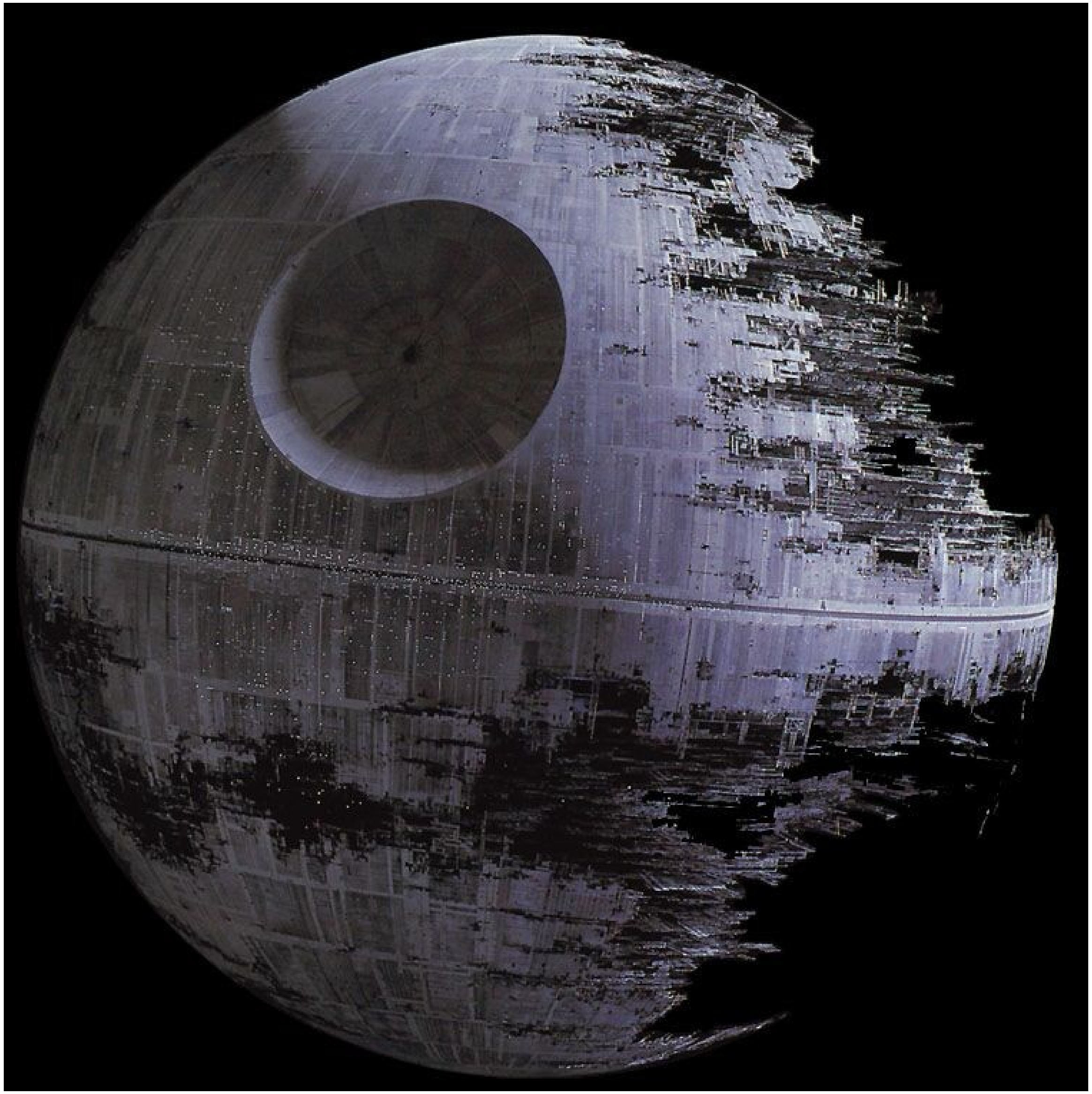
- Modelica language evolves and into gets mainstream
- Increasing number of simulation tools available
- Used for real-life system engineering (and control engineering)
- Success stories
  - Automotive
  - Power plants
  - Robotics
  - Mechatronic systems
  - Aerospace
  - …
- Reusable libraries
- European Projects
  - EuroSysLib
  - ModProd
  - Modelisar
  - …
- Link with Systems Engineering Community

but...

A (not so nice) anecdote



The dark side of O-O modelling



Dynamic modelling problems

Description of model behaviour

+

Description of initial state

Initial state descriptions

- Problem #1: the initialization problem is a *system-level* problem
  - issues with component-wise initial equation setting
  - singular configurations (e.g. closed fluid systems)
- Problem #2: the initial equations might be difficult to solve
  - nonlinear algebraic equations require initial guesses
  - steady-state initialization → large systems of equations
- Problem #3:given initial state values sometimes unsuitable
  - fluid and thermofluid systems: unphysical transient outside model validity range
  - “rest” state might be problematic (singular problem)
  - control studies often requires equilibrium as initial state

➡ How to specify initial conditions conveniently?

➡ How to solve the initialization problem reliably?

The Future
or
2011 and beyond...

Reliable automatic initialization

- Getting a meaningful initial state is a key ingredient in dynamic modelling
- Shouldn't require a PhD in simulation methods
- Should be easy for the average engineer
- Should be easy to specify unambiguously

- Robust methods for initialization problems (no human intervention)
- Meaningful diagnostic feedback for engineers and domain experts

- Embed domain-specific knowledge into model (and meta-model) libraries

Homotopy-based initialization

- Define a simplified problem which is easier to solve

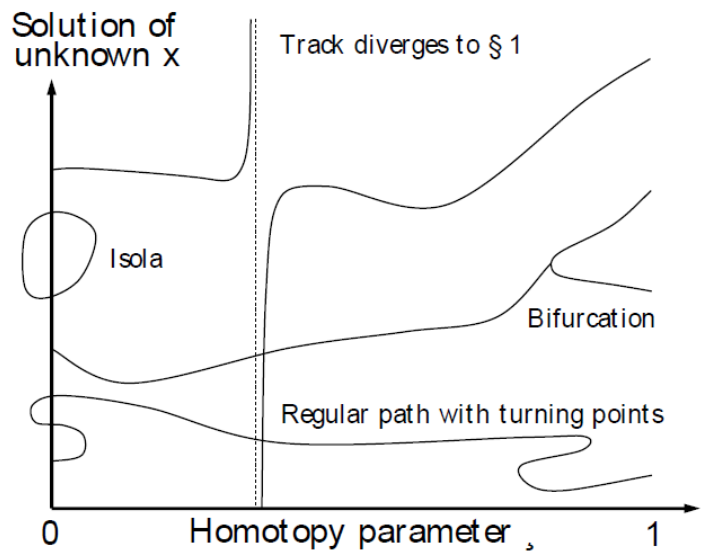
$$F_s(x) = 0$$

- Continuously transform into the actual problem

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

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

- Beware of singularities!

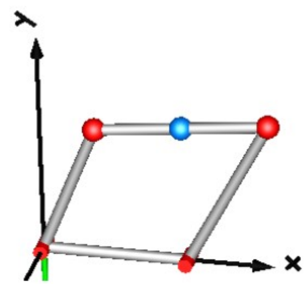


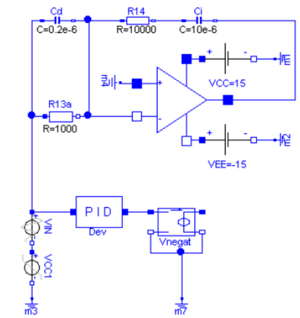
Homotopy-based initialization – First results

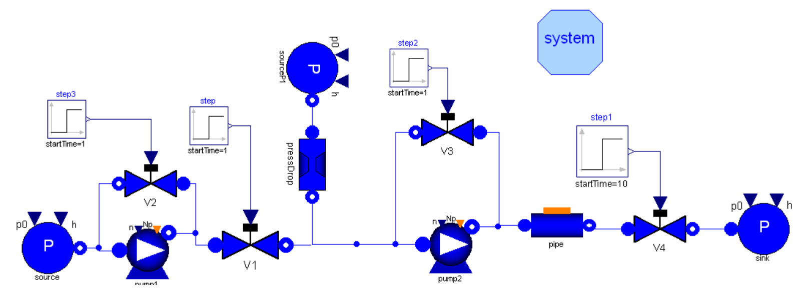
- Multibody systems with multiple configurations
- Analog electronic circuits
- Hydraulic networks
- Calibration of air-conditiong systems
- Large power plants

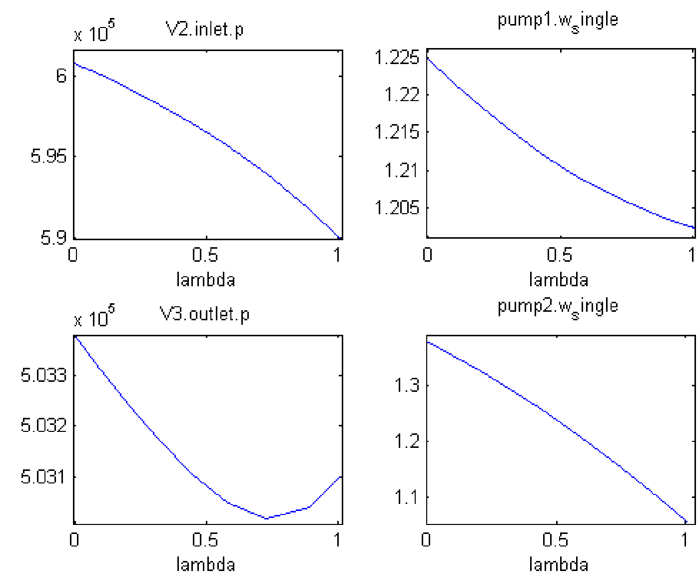
- Two papers at the next Modelica Conference

- One possible way to embed expert knowledge to help the end user



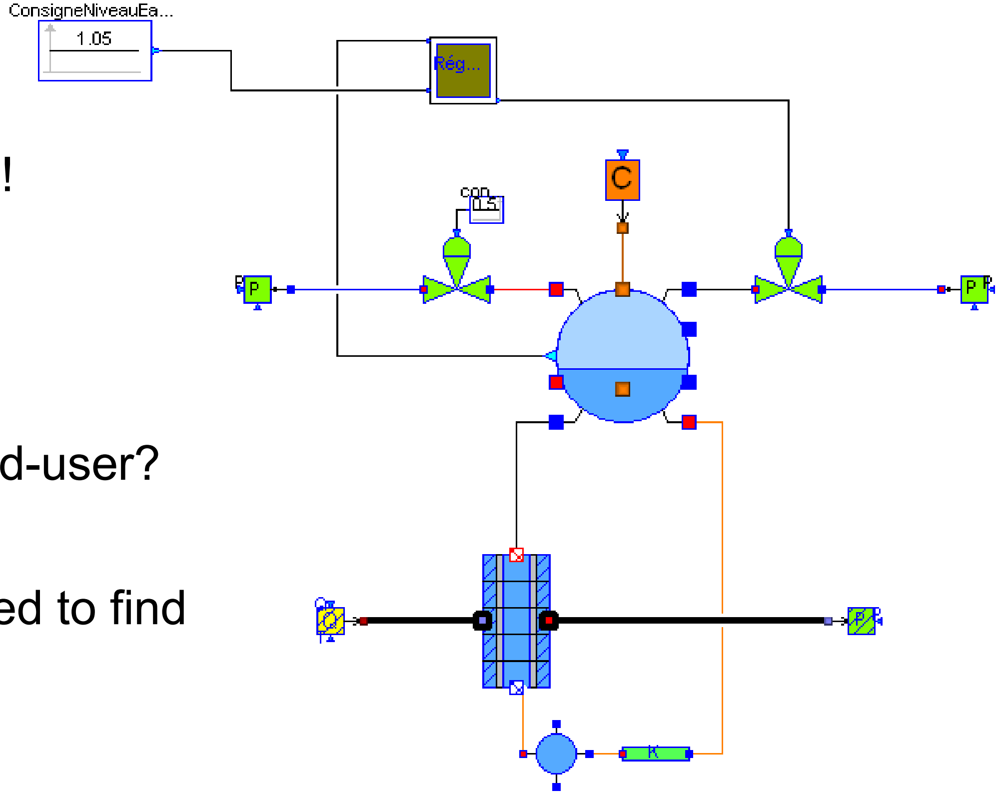






Improved support for troubleshooting

- The strange case of the steam generator
- "Wrong" parameters
  → no steady-state solution!
- Dynamic analysis
- Meaningful hints for the end-user?
- Substantial research needed to find general solutions!



Improved support for troubleshooting – cont'd

- Troubleshooting singular initialization problems
- The case of steady-state initialization of closed-loop circuits

$$\frac{dM_1}{dt} = w_1 - w_2 \qquad \frac{dM_1}{dt} = 0$$

$$\frac{dM_2}{dt} = w_2 - w_1 \qquad \frac{dM_2}{dt} = 0$$

- How to give meaningful feedback to end-users in this case?

Off-line and on-line optimal control

$$\min_{x,u,z,p} \int_0^T l(t, x(t), \dot{x}(t), z(t), u(t), p)\, dt + E(T, x(T), z(T), p)$$

subject to

$$f(t, x(t), \dot{x}(t), z(t), u(t), p) = 0 \quad t \in [0, T]$$

$$h(t, x(t), \dot{x}(t), z(t), u(t), p) \leq 0 \quad t \in [0, T]$$

$$x(0) = x_0$$

$$u_{\min} \leq u(t) \leq u_{\max} \quad t \in [0, T]$$

$$p_{\min} \leq p \leq p_{\max}$$

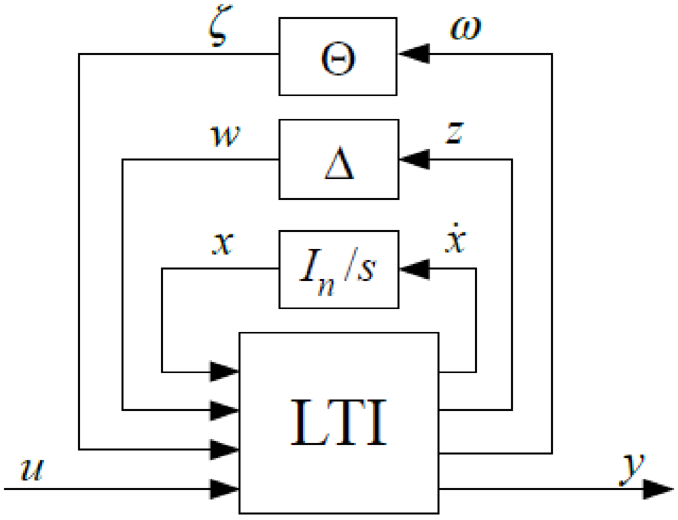
- System model described in Modelica
- Additional settings described in Optimica (small extension of Modelica)
- Jmodelica tool: optimization back-ends
  - IPOPT (direct collocation method)
  - ACADO (multiple-shooting)
- Off-line optimization
- On-line optimization (MPC)

Transformation of models into control-oriented formalism

- Beyond simulation: model transformation
- Control design methods require models with special structure as input
  - linear state-space
  - nonlinear state-space
  - piecewise linear or affine
  - linear-parameter-varying (LPV)
  - linear fractional representation (LFR)
- Modelica front-end helps closing the theory-practice gap

Modelica → LFR

- Prototype implementation of Modelica → LFR transformation
  - developed @ Politecnico
  - uses OMC+ Maxima + LFR toolbox (ONERA/DLR)



- Application
  - Robust control
  - Gain-scheduling control
  - Gray-box identification

Nonlinear Model Order Reduction

- Simulation oriented models usually too complex for direct use in model-based control system
- Model reduction techniques required

- Define refererence scenario(s)
- Define maximum acceptable error
- Rank terms in equations according to sensitivity on output error
- Eliminate / approximate them progressively

- Used in behavioural modelling of digital IC's (Analog Insydes)
- Prototype tool Univ. Duisburg (→ fast real-time performance)

Accurate / general simulation models → Fast / smooth / low-order control-oriented models

Conclusions

- OOM is a blessing for the control engineer (esp. w.r.t. block-oriented)
- Initialization is currently Achille's heel
- Models mainly used for simulation\

- More reliable, automatic, and user-friendly specification and solution of initialization problems
- More end-user relevant feeback in case of trouble
- Model Order Reduction
- Model transformation into control-oriented formalism
- Direct support for optimal control and MPC

Thank you for you kind attention!