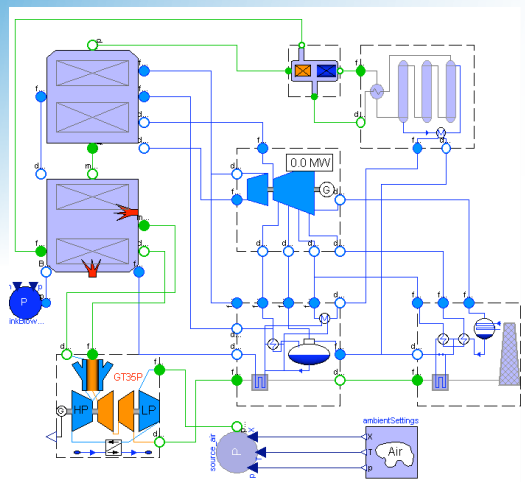
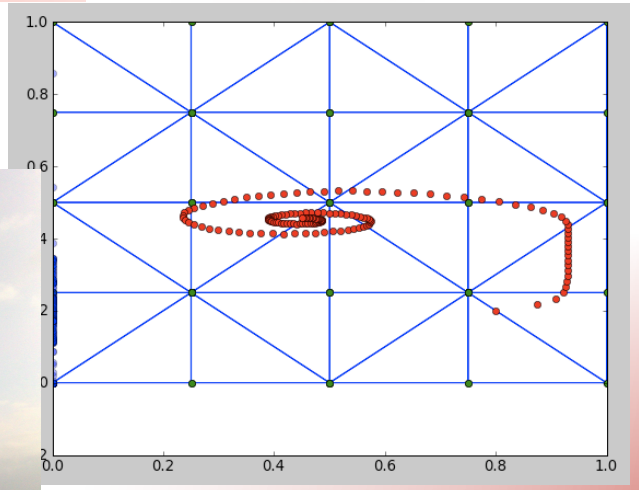
Optimization of Modelica Models with

JModelica.org and Optimica





LUND UNIVERSITY

Modelon

Outline

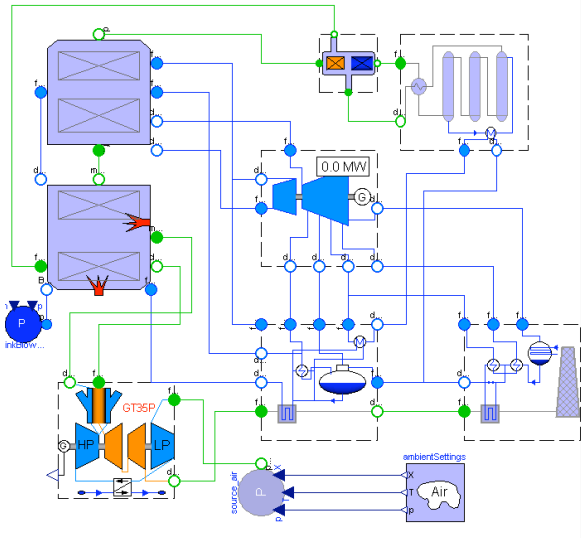
- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

Outline

- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

What is Modelica?

- A language for modeling of complex heterogeneous physical systems
- Open language
  - Managed an developed by non-profit organization Modelica Association
  - Several tools, commercial and free
    - MapleSim
    - Dymola
    - Simulation X
    - OpenModelica
    - Jmodelica.org
- Extensive free standard library
  - Electrical, mechanical, thermodynamics, fluid



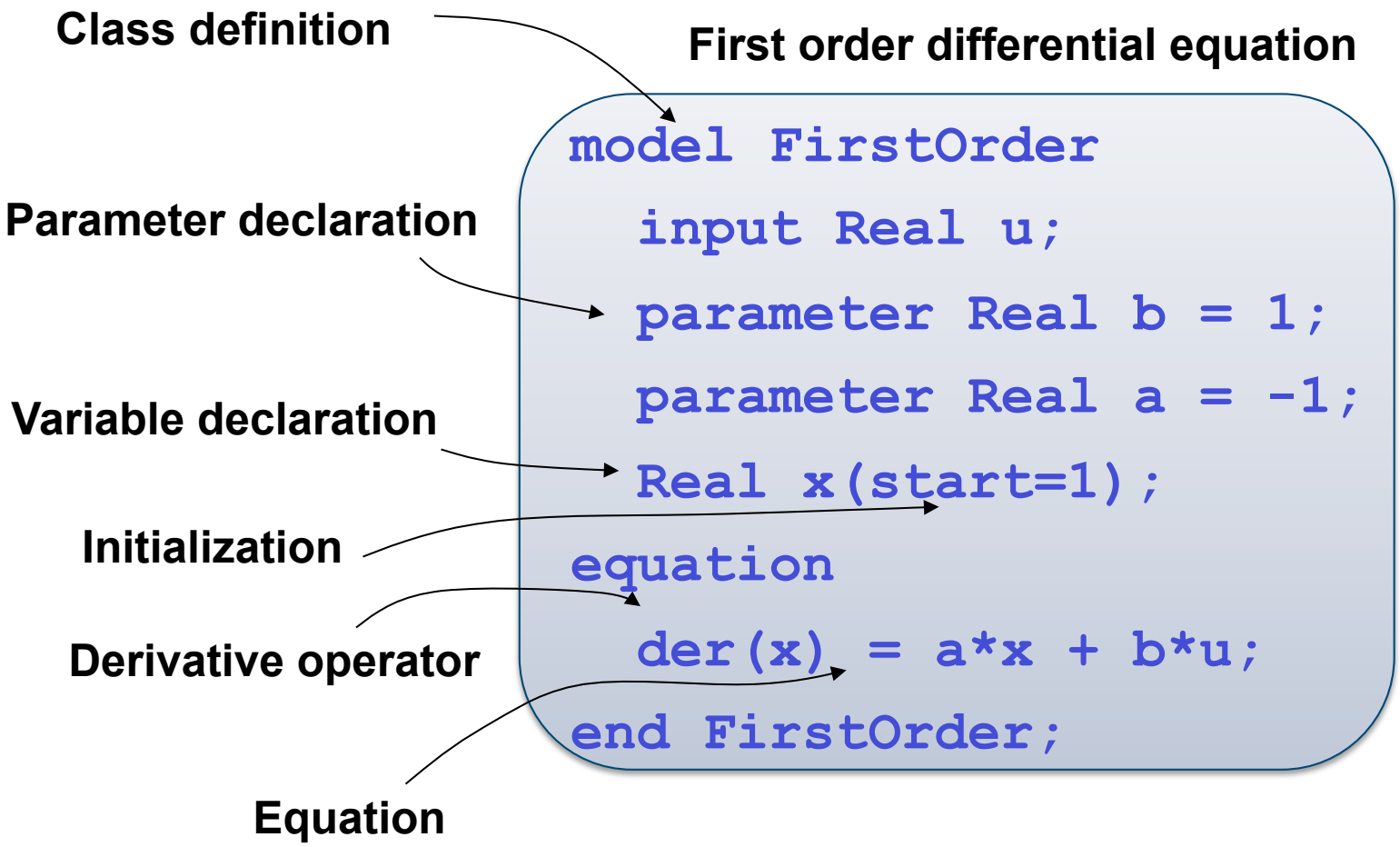
Modelica history

- Evolved from continuous simulation community
  - Simnon
  - Omola/Omsim
  - Bond graphs
- Wide range of applications from start
  - Electronics
  - Mechanics
  - Thermodynamics
- Language development
  - Modelica specification 1.0 in 1997
  - Modelica specification 3.1 in 2009 (3.2 scheduled for March 2010)
- Actively developed by tool vendors and practitioners
  - 65th design meeting in Lund February 2010

Key features of Modelica

- Declarative equation-based modeling
  - Text book style equations
- Multi-domain modeling
  - Heterogeneous modeling
- Object oriented modeling
  - Inheritance and generics
- Software component model
  - Instances and (acausal) connections
- Model libraries
- Function support
- Hybrid Differential Algebraic Equation (DAE) formalism
- Large models (>10.000 equations)

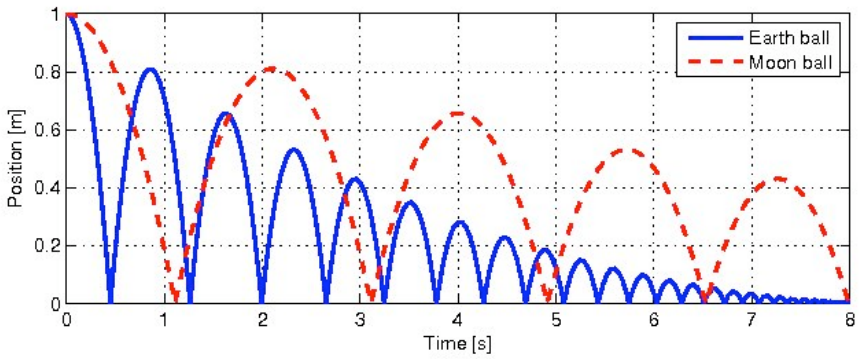
A simple Modelica Model



Hybrid modeling

```
class BouncingBall //A model of a bouncing ball
  parameter Real g = 9.81; //Acceleration due to gravity
  parameter Real e = 0.9; //Elasticity coefficient
  Real pos(start=1); //Position of the ball
  Real vel(start=0); //Velocity of the ball
equation
  der(pos) = vel;   // Newtons second law
  der(vel) = -g;
  when pos <=0 then
    reinit(vel,-e*pre(vel));
  end when;
end BouncingBall;
```

```
class BBex
  BouncingBall eBall;
  BouncingBall mBall(g=1.62);
end BBex;
```

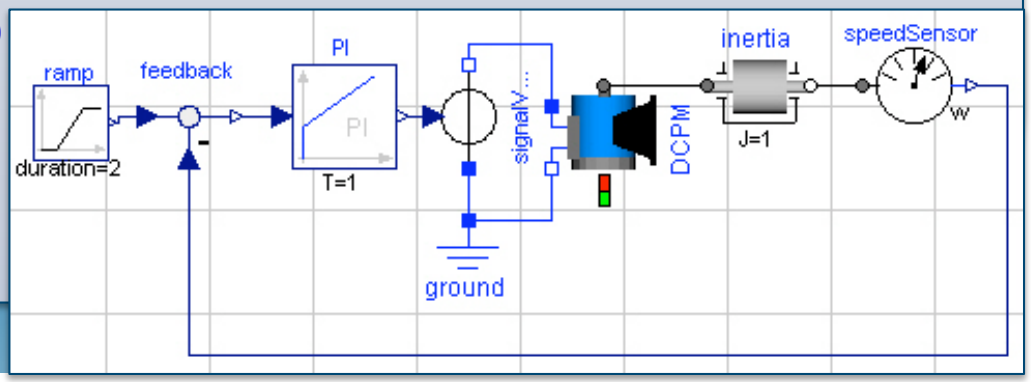


Functions and algorithms

```
function bubbleSort
  input Real [:] unordElem;
  output Real [size(unordElem, 1)] ordElem;
  protected
    Real tempVal;
    Boolean isOver = false;
  algorithm
    ordElem := unordElem;
    while not isOver loop
      isOver := true;
      for i in 1:size(ordElem, 1)-1 loop
       if ordElem[i] > ordElem[i+1]
       then
          tempVal      := ordElem[i];
          ordElem[i]   := ordElem[i+1];
          ordElem[i+1] := tempVal;
          isOver := false;
       end if;
      end for;
    end while;
end bubbleSort;
```

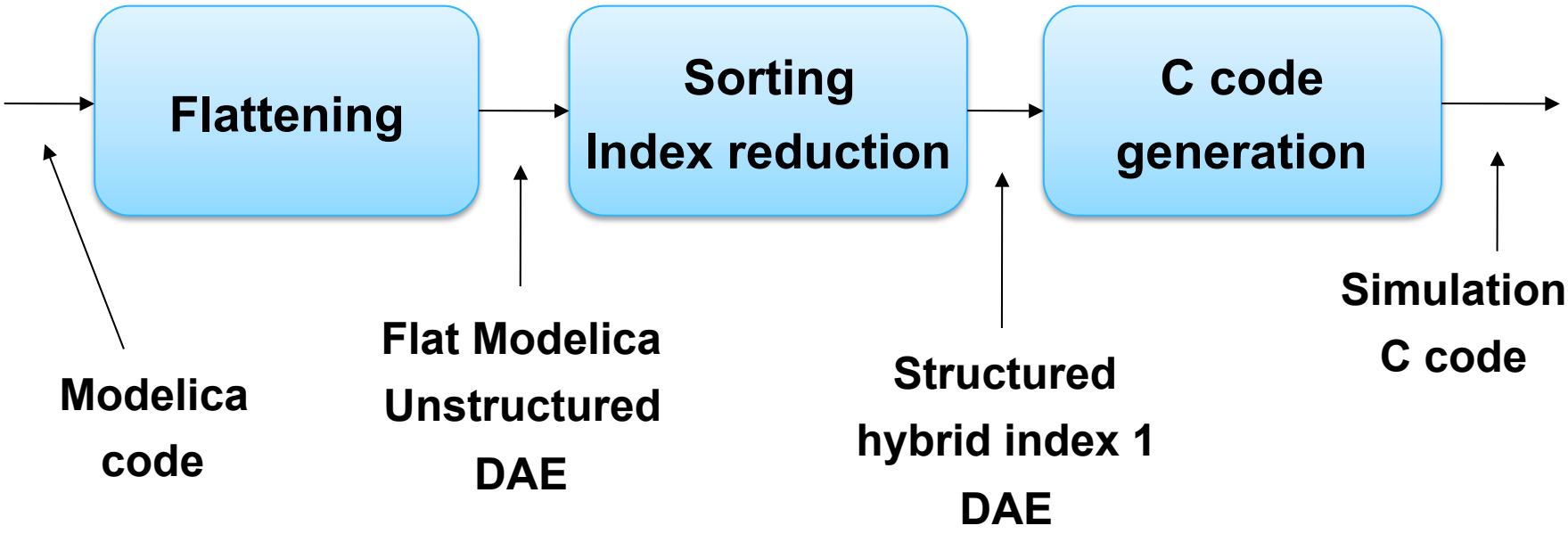
Graphical and textual modeling

```
model MotorControl
  Modelica.Mechanics.Rotational.Inertia inertia;
  Modelica.Mechanics.Rotational.Sensors.SpeedSensor speedSensor;
  Modelica.Electrical.Machines.BasicMachines.DCMachines.DC_PermanentMagnet DCPM;
  Modelica.Electrical.Analog.Basic.Ground ground;
  Modelica.Electrical.Analog.Sources.SignalVoltage signalVoltage;
  Modelica.Blocks.Math.Feedback feedback;
  Modelica.Blocks.Sources.Ramp ramp(height=100, startTime=1);
  Modelica.Blocks.Continuous.PI PI(k=-2);
equation
  connect(inertia.flange_b, speedSensor.flange_a);
  connect(DCPM.flange_a, inertia.flange_a);
  connect(speedSensor.w, feedback.u2);
  connect(ramp.y, feedback.u1);
  connect(signalVoltage.n, DCPM.pin_ap);
  connect(signalVoltage.p, ground.p);
  connect(ground.p, DCPM.pin_an);
  connect(feedback.y, PI.u);
  connect(PI.y, signalVoltage.v);
end MotorControl;
```



Translation of Modelica models

- Generation of a mathematical model description from Modelica code



Outline

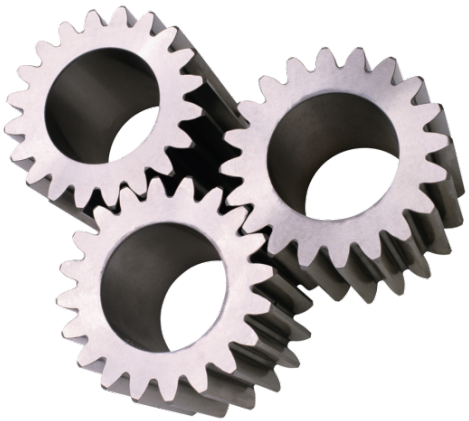
- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

Optimization and Modelica

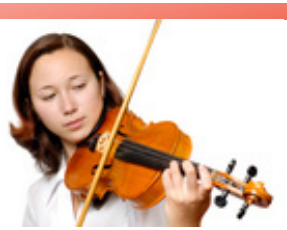
- Modelica increasingly used in industry
  - Expert knowledge
  - Capital investments
- Mainly simulation... ...but new areas emerge
  - Model reduction
  - Parameter identification
  - Dynamic optimization
  - Model predictive control
- Usages reported so far
  - Cope with simulation-oriented interfaces
  - Treat model essentially as a black box
- Fast algorithms explores model structure
  - Real-time optimization

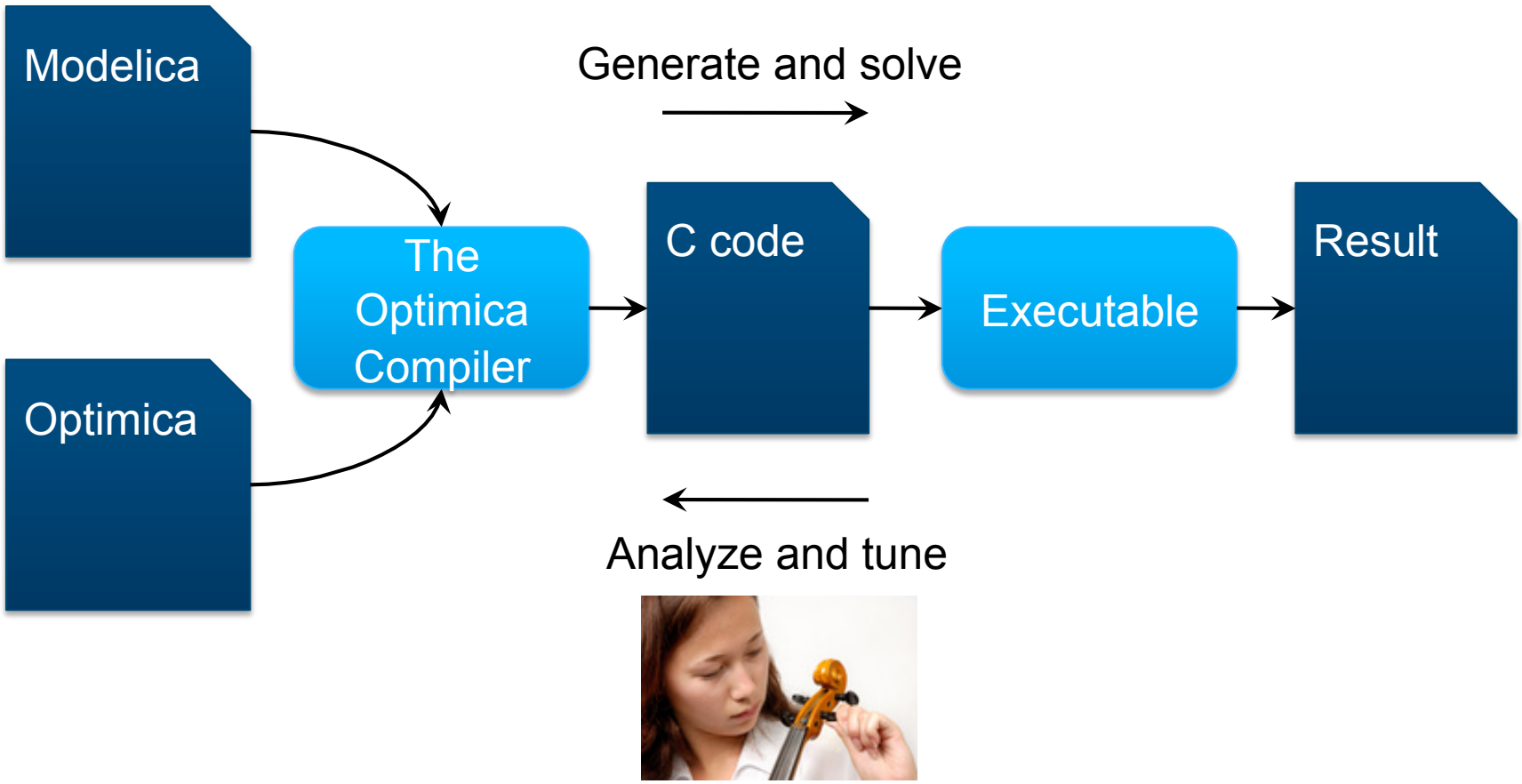
Dynamic optimization

- Many algorithms
  - Applicability highly model-dependent (ODE, DAE, PDE, hybrid)
    - Calculus of variations
    - Single/Multiple shooting
    - Simultaneous methods
    - Simulation-based methods (GA, simulated annealing)
- Analogy with different simulation algorithms
  - Heavy burden to used numerical algorithms
  - Fortran, C, (AMPL)
- Engineering need for high-level descriptions
  - Shift focus from *encoding*
  - to *formulation* of optimization problem



Typical workcycle





Dynamic optimization

$$\min_{u(t),\,p} \Psi(\bar{z}, p)$$

subject to the dynamic system

$$F(\dot{x}(t), x(t), y(t), u(t), p, t) = 0, \quad t \in [t_0, t_f]$$

and the constraints

$$c_{ineq}(x(t), y(t), u(t), p) \leqslant 0, \quad t \in [t_0, t_f]$$
$$c_{eq}(x(t), y(t), u(t), p) = 0, \quad t \in [t_0, t_f]$$
$$c^p_{ineq}(\bar{z}, p) \leqslant 0$$
$$c^p_{eq}(\bar{z}, p) = 0$$

where

$$\bar{z} = [x(t_1), \ldots, x(t_{N_p}), y(t_1), \ldots, y(t_{N_p}), u(t_1), \ldots, u(t_{N_p})]^T, \quad t_i \in [t_0, t_f]$$

Optimization with Modelica

- Strong support for modeling of dynamic systems
- Missing elements
  - Cost function
  - Constraints
  - What to optimize
  - Initial guesses
- Optimica
  - Small extension of Modelica
  - Enable high-level formulation of optimization problems

An example

$$\min_{u(t)} \int_{t_0}^{t_f} 1\, dt$$

subject to the dynamic constraint

$$\dot{x}_1(t) = (1 - x_2(t)^2)x_1(t) - x_2(t) + u(t), \quad x_1(0) = 0$$
$$\dot{x}_2(t) = x_1(t), \quad x_2(0) = 1$$

and

$$x_1(t_f) = 0$$
$$x_2(t_f) = 0$$
$$-1 \leqslant u(t) \leqslant 1$$

A Modelica model

```
model VDP
  Real x1(start=0);
  Real x2(start=1);
  input Real u;
equation
  der(x1) = (1-x2^2)*x1 - x2 + u;
  der(x2) = x1;
end VDP;
```

Modelon

An Optimica model

```
optimization VDP_Opt(objective=cost(finalTime),
      startTime=0,
      finalTime(free=true, initialGuess=1))
  VDP vdp(u(free=true,initialGuess=0.0));
    Real cost (start=0);
equation
    der(cost) = 1;
constraint
    vdp.x1(finalTime) = 0;
    vdp.x2(finalTime) = 0;
    vdp.u >= -1; vdp.u <= 1;
end VDP_Opt;
```

Outline

- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

The JModelica.org open source project

What?

JModelica.org is an extensible Modelica-based open source platform for optimization, simulation and analysis of complex dynamic systems.

Our mission:

To offer a community-based, free, open source, accessible, user and application oriented Modelica environment for optimization and simulation of complex dynamic systems, built on well-recognized technology and supporting major platforms

Origin:

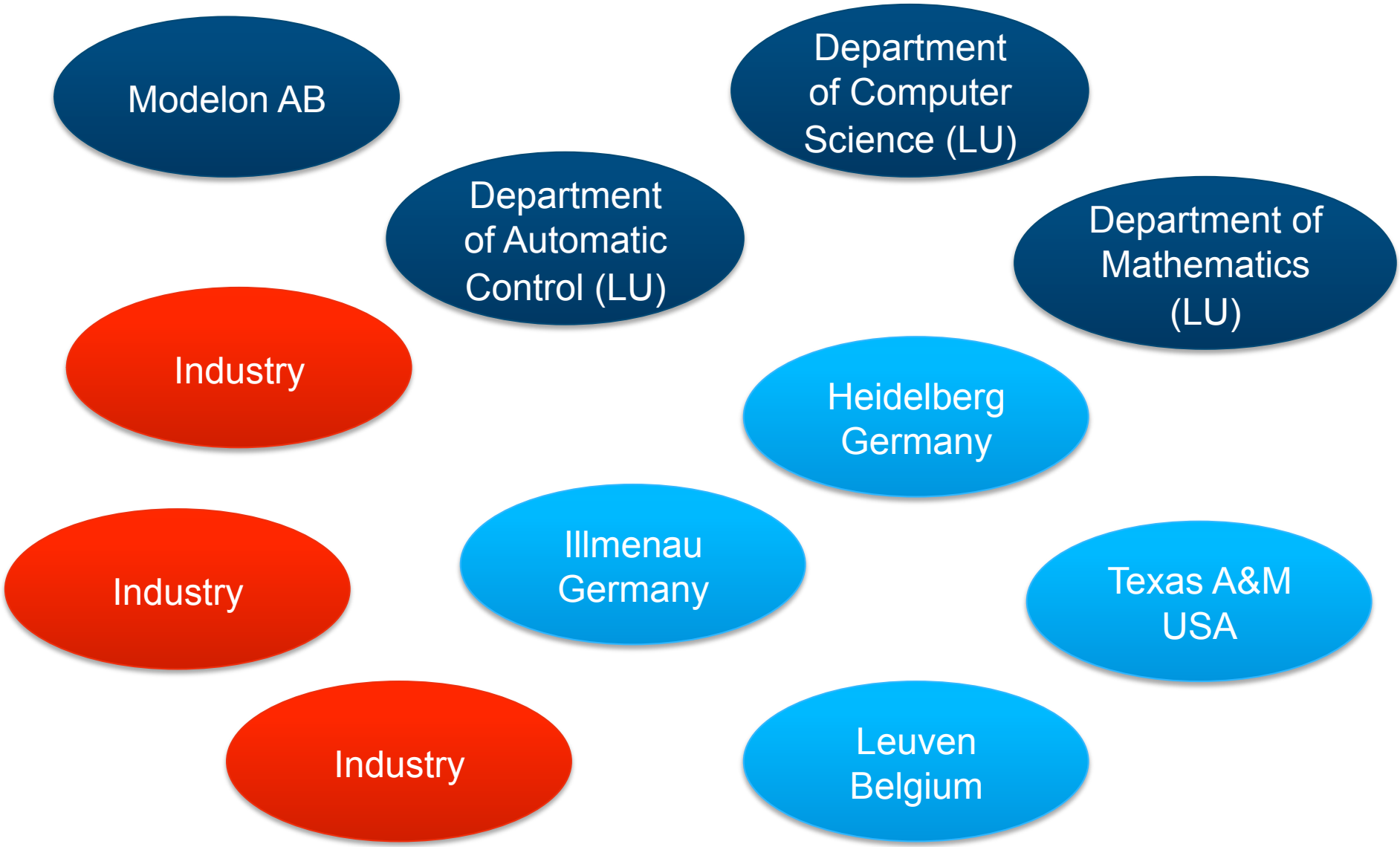
JModelica.org is the result of research at the Department of Automatic control, Lund University, and is now maintained and developed by Modelon AB in cooperation with academia.

Modelon

JModelica.org is open source

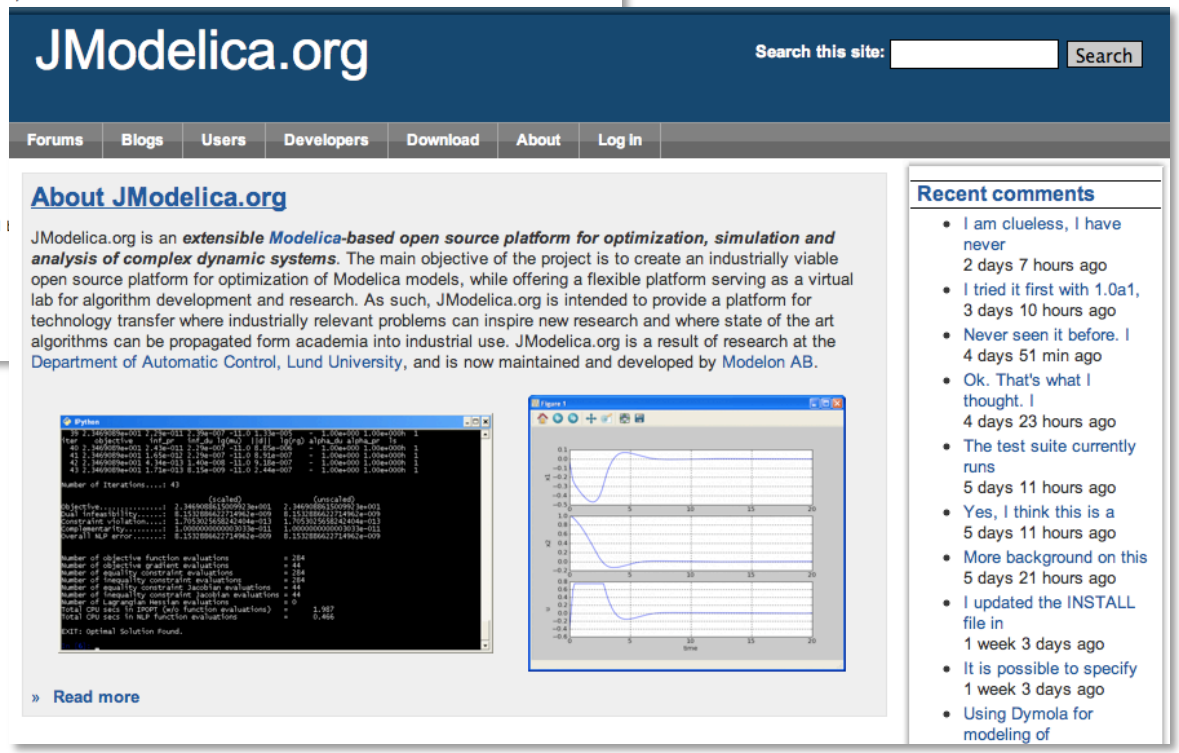
- Source code is freely available
  - Open Source Initiative approved licenses
  - GPL (CPL)
- Infrastructure supporting a community
  - Transparency of development
  - Interactive web site
  - User forums

Ecosystem


Modelon AB
Department of Computer Science (LU)
Department of Automatic Control (LU)
Department of Mathematics (LU)
Industry
Heidelberg Germany
Industry
Ilmenau Germany
Texas A&M USA
Industry
Leuven Belgium

Community

- Transparency
- Track activity
- Releases

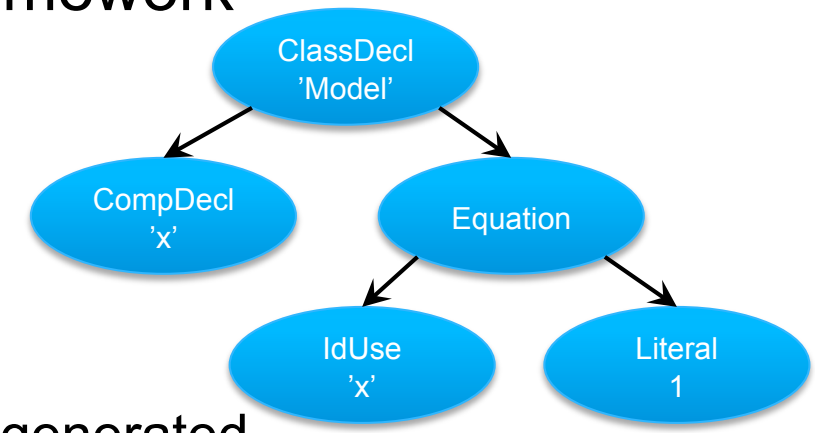


- Documentation
- News
- Forums

Technologies

- Modelica
- JastAdd meta-compiler tool
  - Modelica/Optimica compiler front-ends pure Java
  - Easily embedded jar-files
- Python
  - Scientific computing environment
  - Scripting and visualization
  - Custom application development
- XML
  - Model meta data
- Eclipse
  - Modelica and Optimica IDE
  - Refactoring

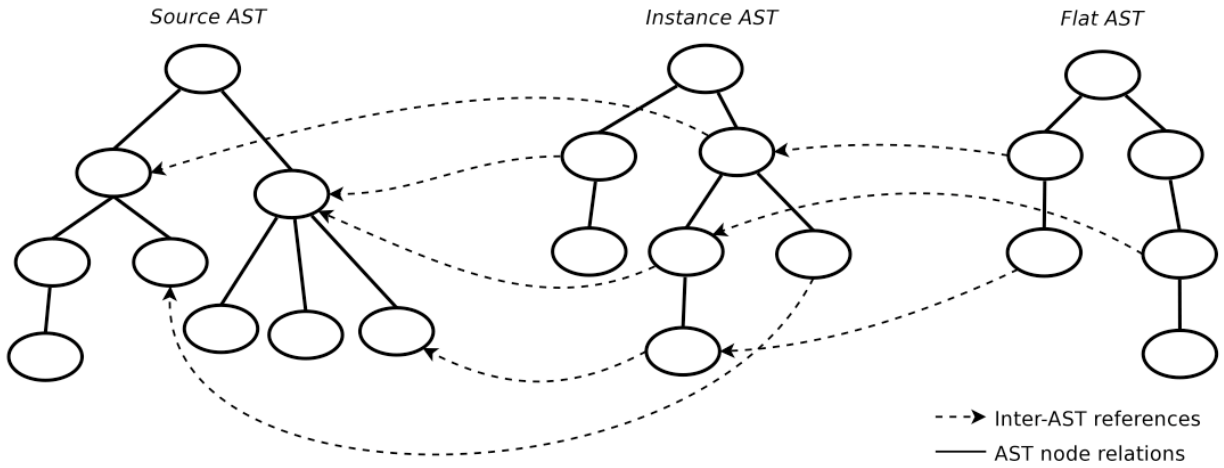
JastAdd



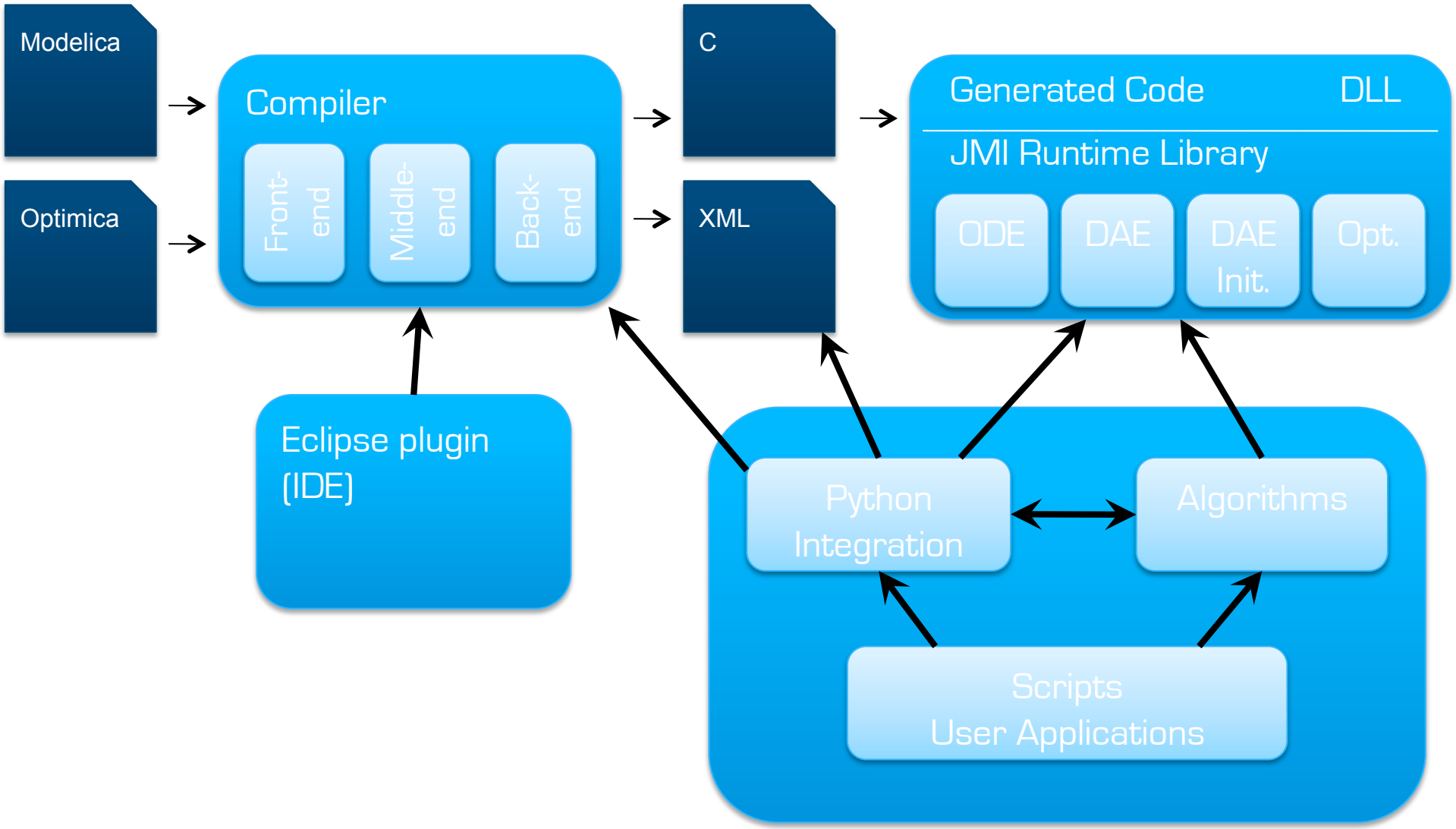
- A compiler meta-compilation framework
- Extension of Java
- Object-orientation
- Static aspect-orientation
- Abstract syntax specification
  - AST class hierarchy automatically generated
- Reference attributed grammars
- Declarative programming
  - Attributes define properties of AST nodes
  - Equations define values
  - Reference attributes to connect nodes in AST
- Generates Java code
  - Portable and easy to embed jar files

Abstract Syntax Trees

- Abstract syntax trees represented by Java objects
- AST traversal API available
- Query language elements
- Three ASTs
  - Source AST results from parsing
  - Instance AST represents a model instance
  - Flat AST represents flattened model

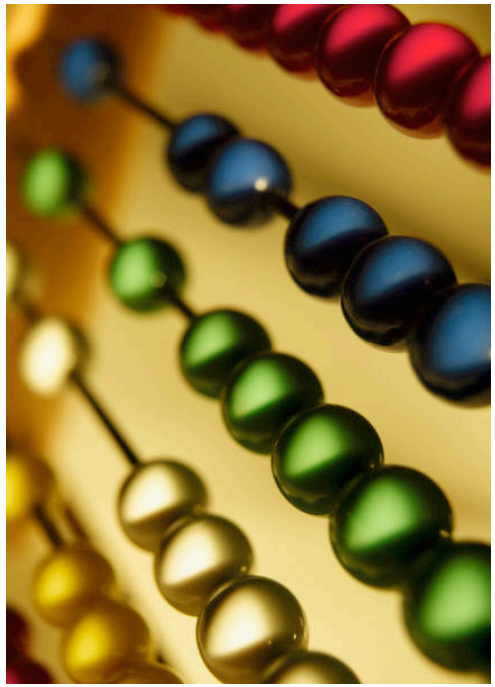


JModelica.org architecture



Algorithms in JModelica.org

- Collocation optimization algorithm
  - Large but sparse non-linear program
  - Fast
  - Numerical solver Ipopt
- Multiple-shooting algorithm
  - Simulation-based
  - Sensitivity equations
  - SUNDIALS DAE integrator



Outline

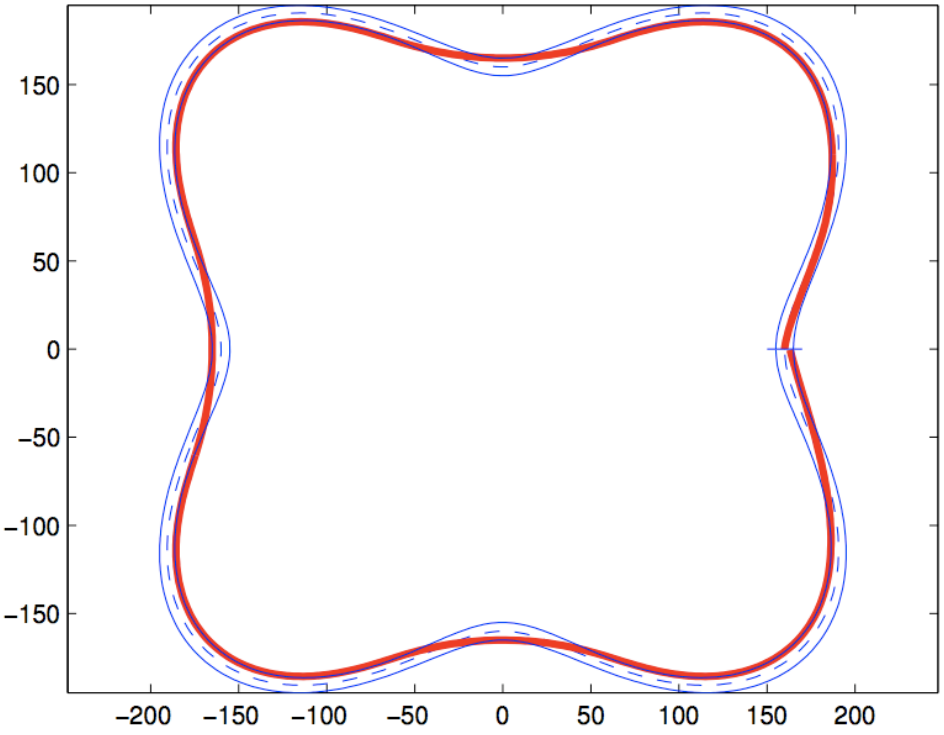
- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

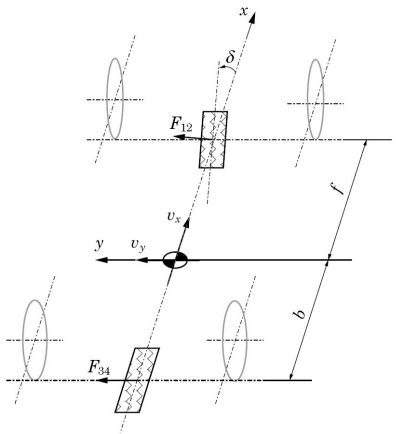
Grade change optimization

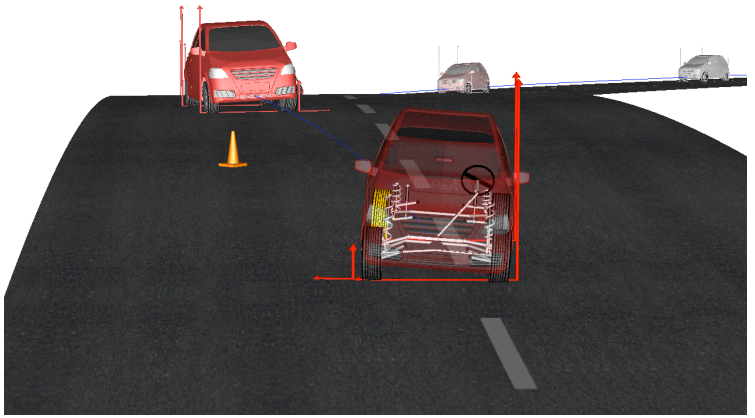
- Collaboration with Plastics manufacturer Borealis
- Polyethylene production
- Tree reactors in series
  - Pre-loop
  - Loop
  - Gas phase
- Decision support
  - Flexible production
  - Raw material prices vary
  - Minimize off-spec
- JModelica.org for optimization
- PhD student project: Per-Ola Larsson

Optimal racing

- Minimize lao time
- Simplified car model
- Find acceleration and steering angle



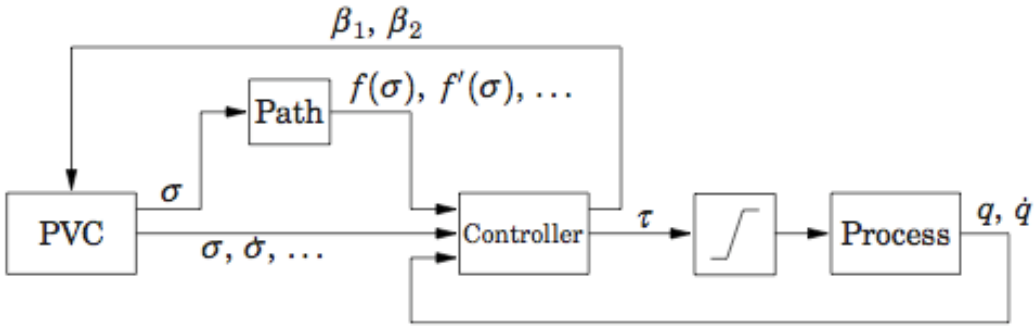




Master's thesis by Henrik Danielsson

Time optimal robot control

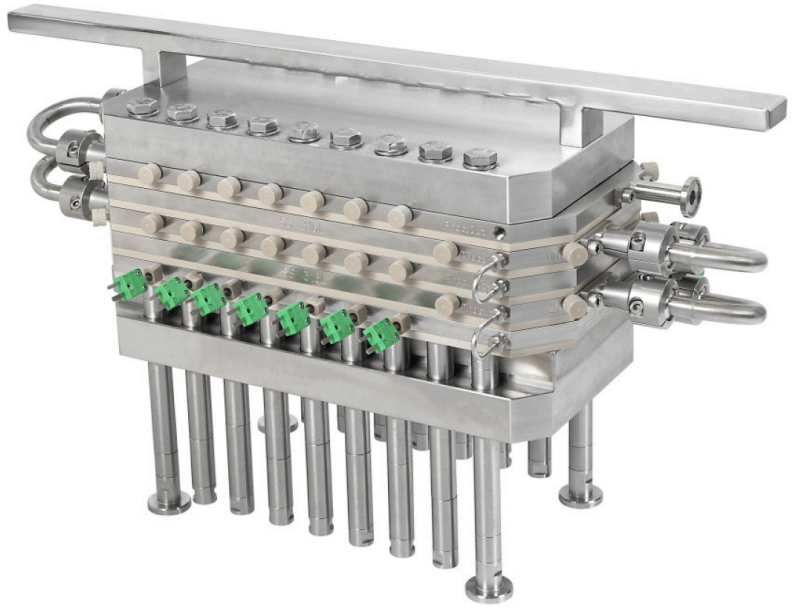
- Track specified paths in 3D
- Record path
- Generate splines
- Optimize
- Implement



Master's thesis by Marin Hast

Optimal Start-up of a Plate Reactor

- Start-up of a plate reactor with exothermic reaction
- Tubular reactor inside a heath-exchanger
- Multiple injection points (hot spots)
- Generate open loop start-up trajectories using dynamic optimization
- Large scale system
  - >100 equations and variables
- Multiple objectives
  - Bounded temperature profiles
  - Maximum conversion
  - Minimum start-up times
  - Robust to parameter variations



Outline

- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

What's next?

- Improved Modelica compliance
  - Vectors
  - Functions
- Simulation
  - SUNDIALS
  - Hybrid systems
- Continued work on Python interface
  - Feedback appreciated
- Extended XML export
- Improved AST interfaces
- Continued work on optimization algorithms
- Eclipse plug-in
- Application-driven development

XML export

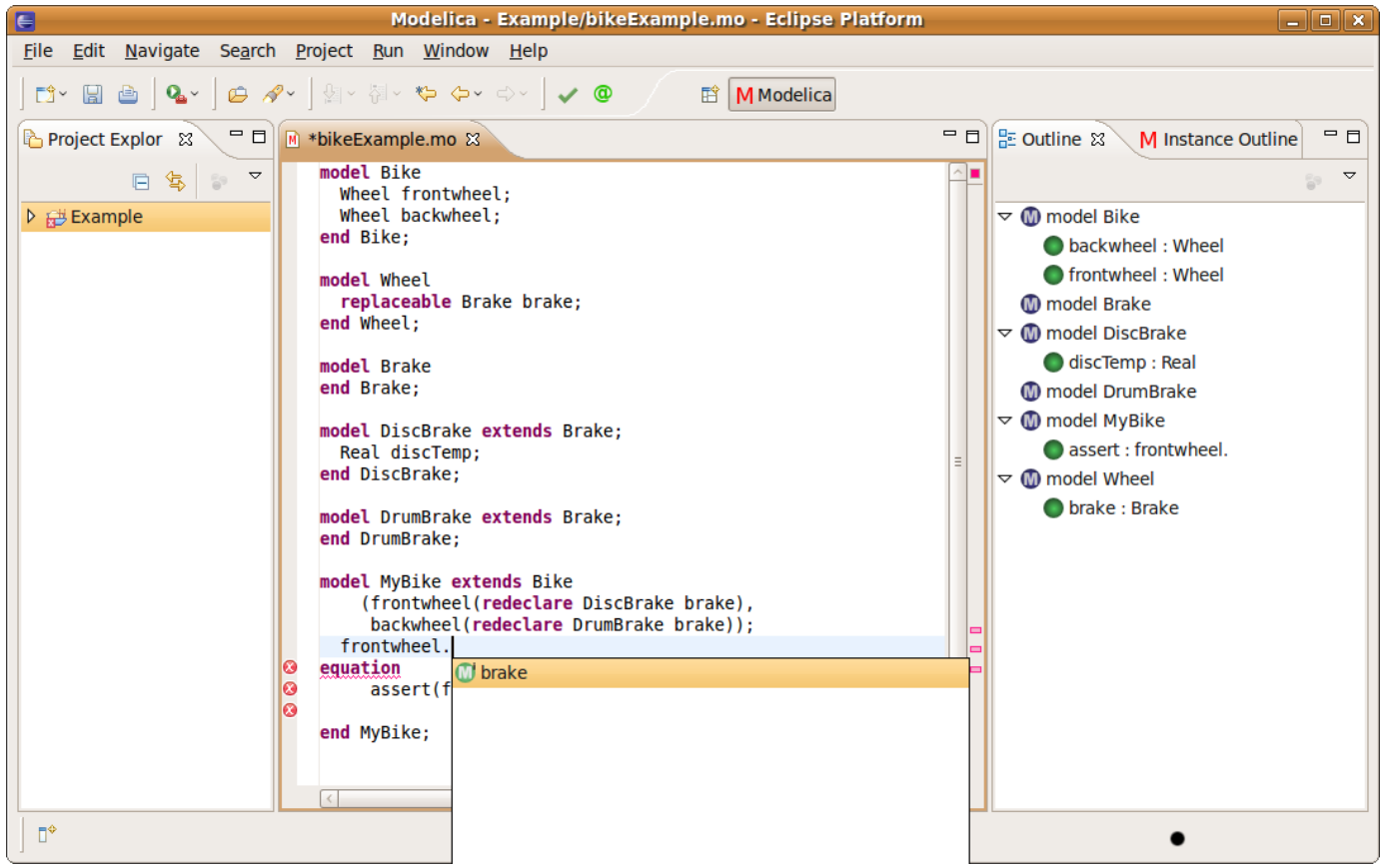
- Currently supports XML export of model meta data
  - Variable names
  - Attributes
  - Parameter values
  - Equations
  - Optimica (cost functions and constraints)
- Planned extensions
  - Functions
  - XML import
  - Integration with ACADO Toolkit (optimization package)
- Increased flexibility
  - Decouple front-ends and back-ends
  - Standardized transformation language (XSLT)

Interfaces to ASTs

- Currently API for flat model
  - Variables of primitive types
  - Equations
- Planned API for source AST
  - Add elements (classes, components…) interactively
  - Query and modify annotations
- Planned API for instance AST
  - Query model instances
  - Analyze modification environments

Modelica/Optimica Eclipse plugin

- Class and instance outlines
- Name completion á la JDT
- Refactoring
- Graphical ed.



JModelica.org in research

- Continued work on Optimica language extension
  - Execution of Model predictive controllers
- PIC-LU (Process Industrial Center at Lund University)
  - Grade change optimization at Borealis
  - Poly-ethane process
  - Robust optimization
- Parallelization of optimization algorithms
  - Explore the power of multi and many core
  - Decomposition schemes in interior point mehtods
- Safe refactoring
  - Exploit research related to JastAdd
  - Adapt and extend framework developed for Java

Join the community

- Use in industrial applications
- Education
- Interfacing your algorithms
- Research
- Develop