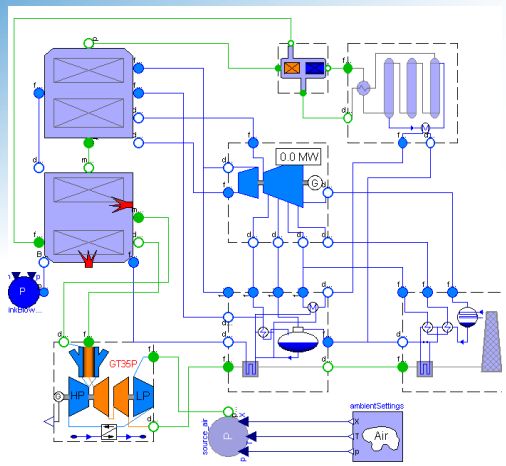
Modeling and Optimization with Optimica and JModelica.org – Languages and Tools for Solving Large-Scale Dynamic Optimization problems



LUND UNIVERSITY

Modelon

Outline

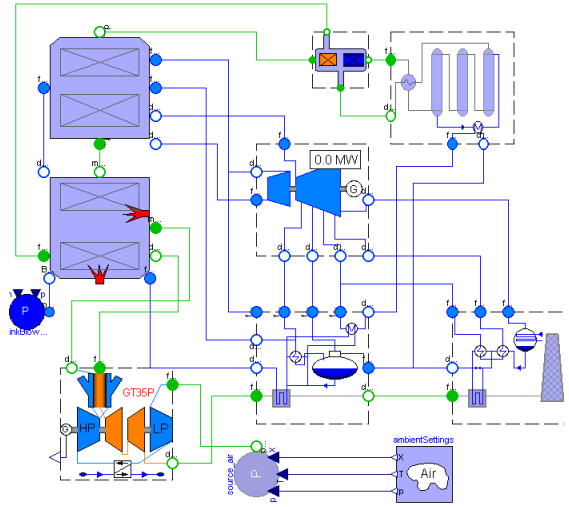
- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What’s next?

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



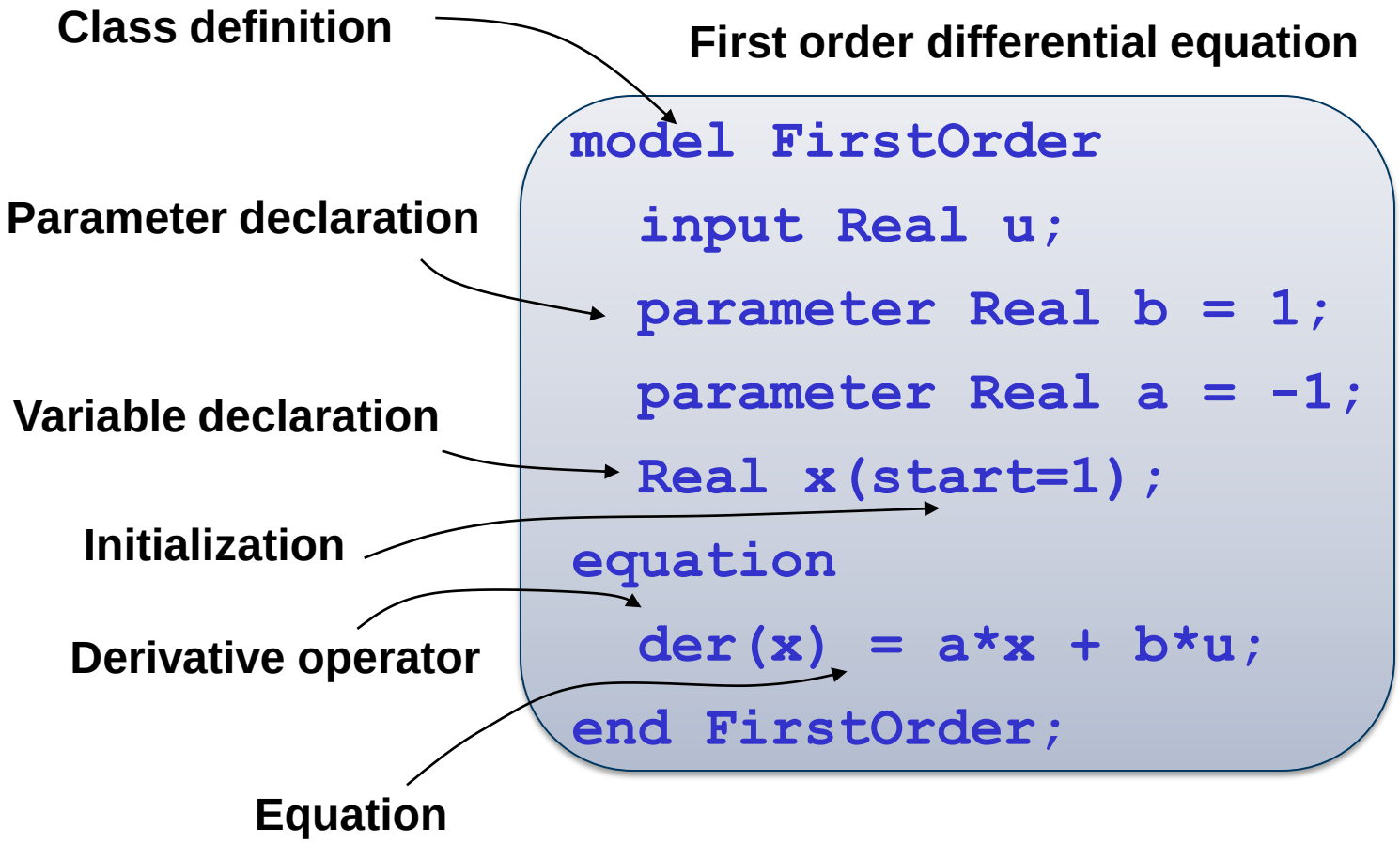
Modelica history

- 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 3.2 in 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)

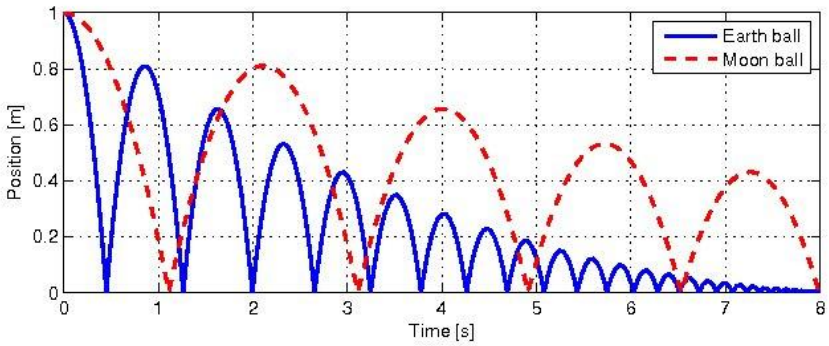
A simple Modelica Model



Hybrid modeling

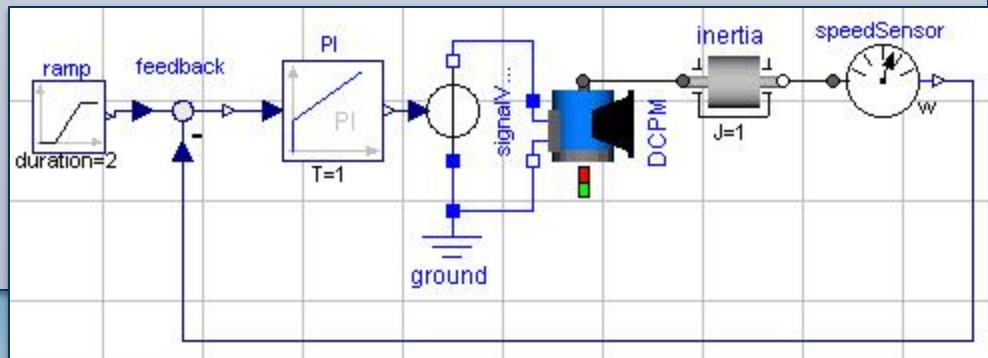
```
class BouncingBall //A model of a bouncing ball
  parameter Real g = 9.81; //Acceleration due to gravity
  parameter Real e = 0.9; //Elasticity coefficient
  Real pos(start=1); //Position of the ball
  Real vel(start=0); //Velocity of the ball
equation
  der(pos) = vel;   // Newtons second law
 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



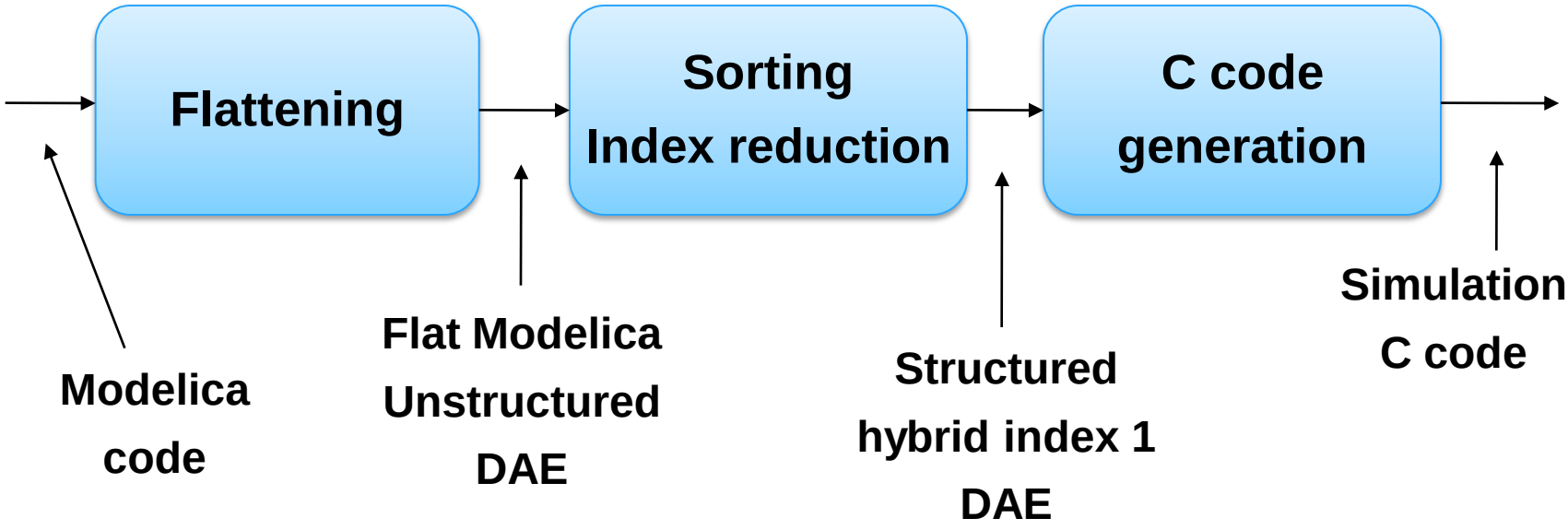
Graphical and textual modeling

```
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
    - Direct single/multiple shooting
    - Direct collocation 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

Generate and solve

Modelica

Optimica

The Optimica Compiler

C code

Executable

Result

Analyze and tune

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

Optimica – 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;
```

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

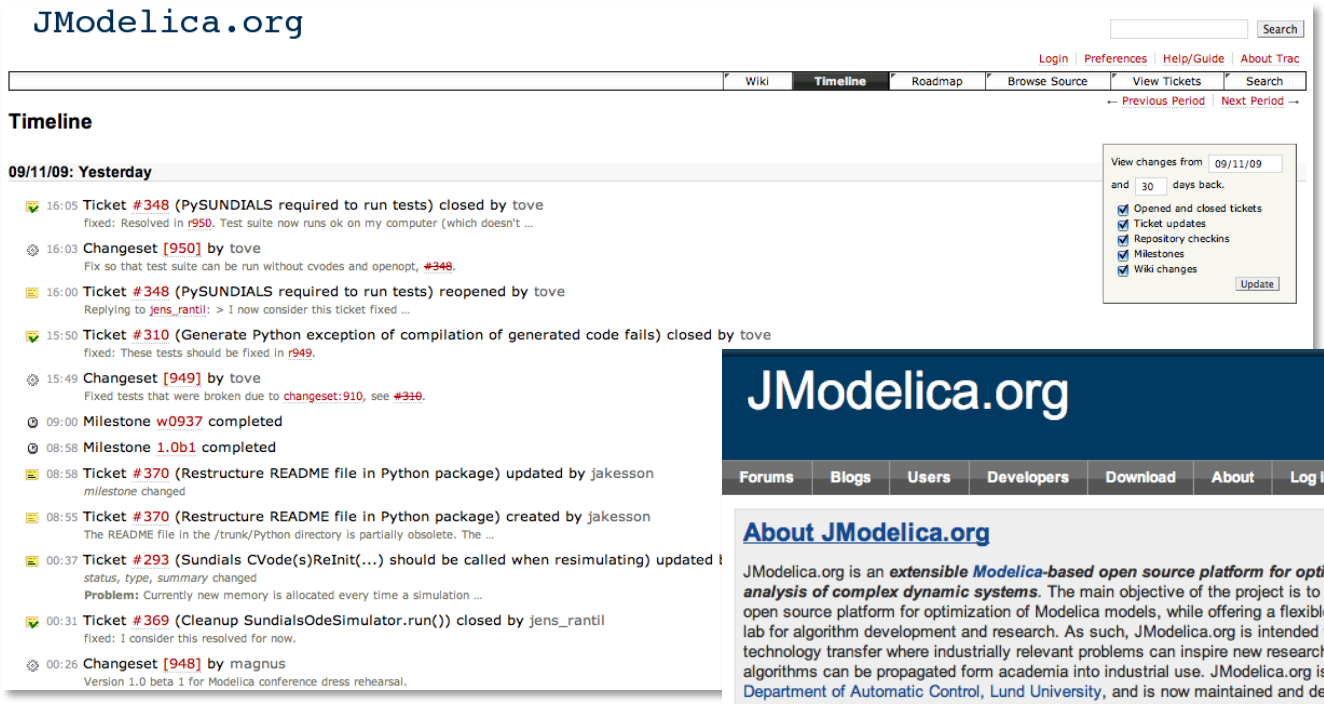
Origin:

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.

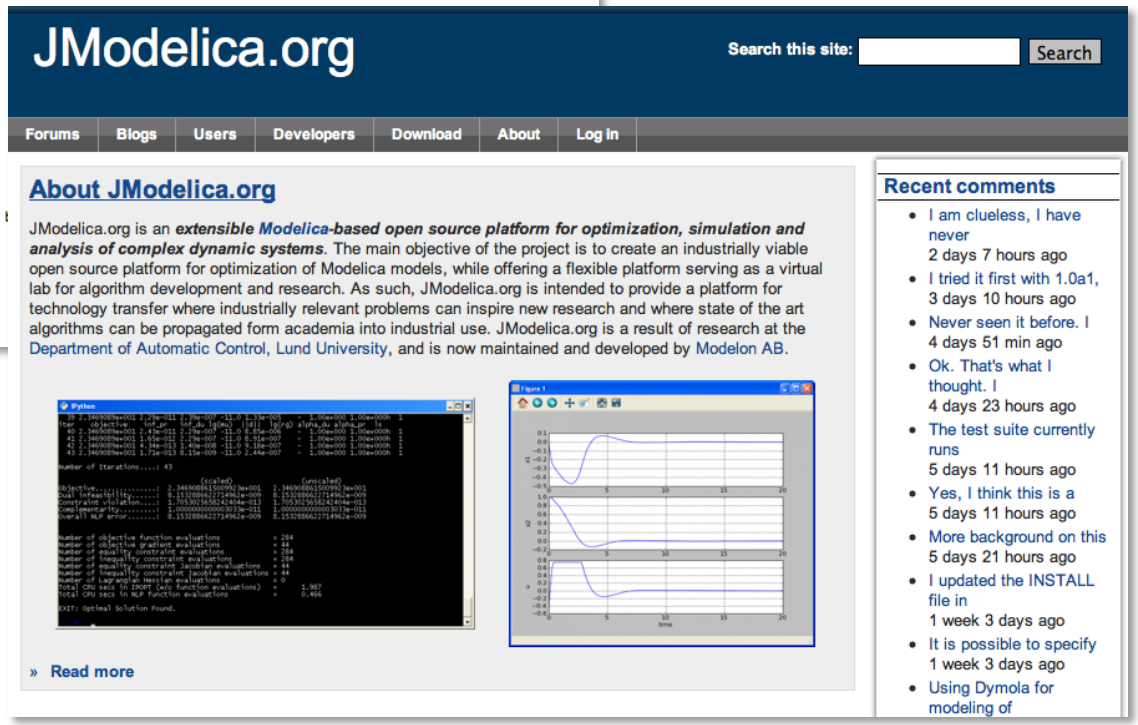
JModelica.org is open source

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

Community



- Transparency
- Track activity
- Releases

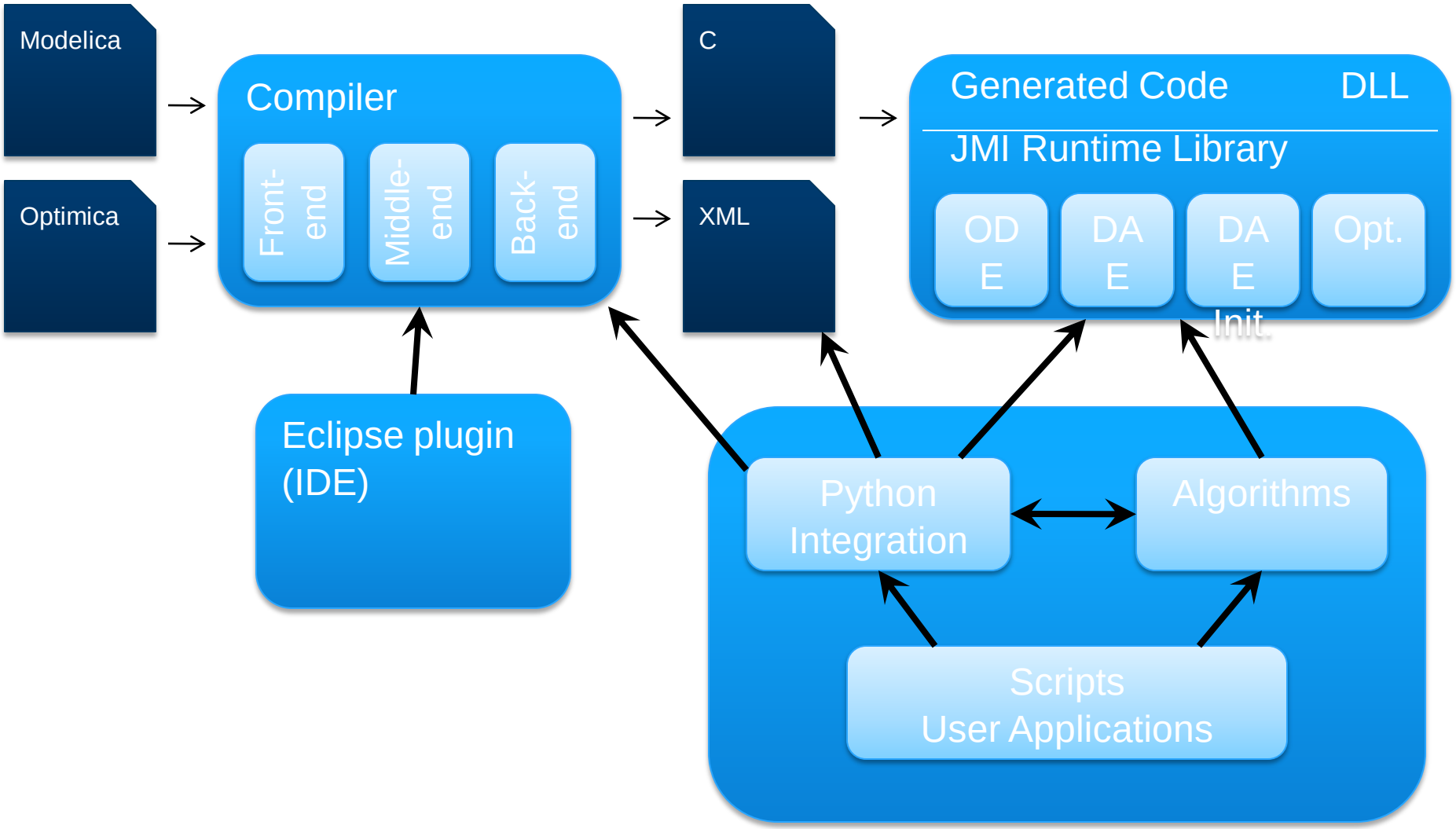


- Documentation
- News
- Forums

Technologies

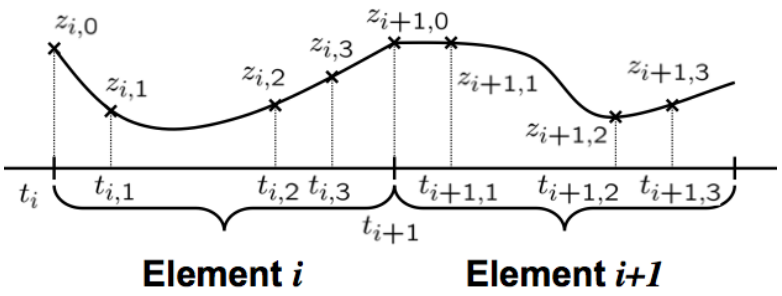
- Modelica
- JastAdd meta-compiler tool
  - Modelica/Optimica compiler front-ends pure Java
  - Easily embedded jar-files
- Python
  - Scientific computing environment
  - Scripting and visualization
  - Custom application development
- XML
  - Model meta data and equations
- Eclipse
  - Modelica and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

JModelica.org architecture



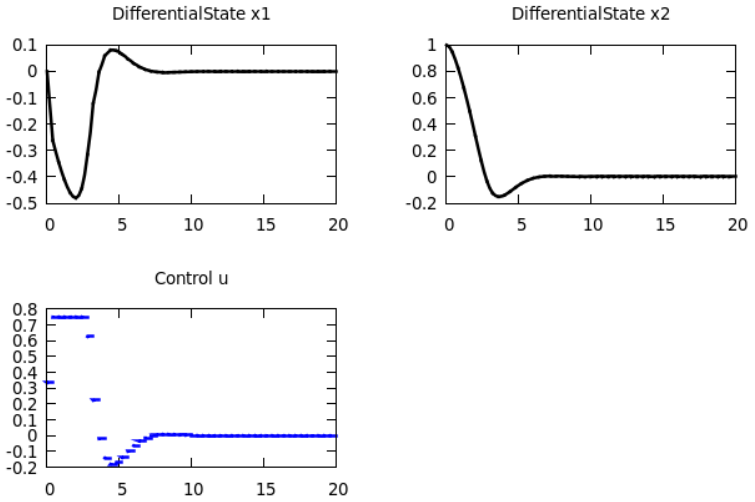
Interface I: C model execution API–JMI

- C functions for evaluation of:
  - DAE residual
  - DAE initialization system
  - Event indicator functions (hybrid systems)
  - Cost function
  - Constraints
- Integrated AD package: CppAD
  - High-accuracy Jacobians
  - Sparsity patterns
- Direct collocation algorithm
  - Approximate controls and states by piecewise polynomials
  - Large but sparse non-linear program
  - NLP solver IPOPT



Interface II: XML export

- Standardized format for DAE model exchange
- Neutral w.r.t. model usage
  - Simulation, optimization, LFTs, model reduction
- (Not yet quite) neutral w.r.t. modeling language
- Model meta data
  - Variable names
  - Parameter values
- Model equations
  - DAE
  - Initialization system
  - Functions and algorithms
  - Optimization
- XML import to ACADO



MSc project by Roberto Parrotto in collaboration with Joel Andersson, KU Leuven

Outline

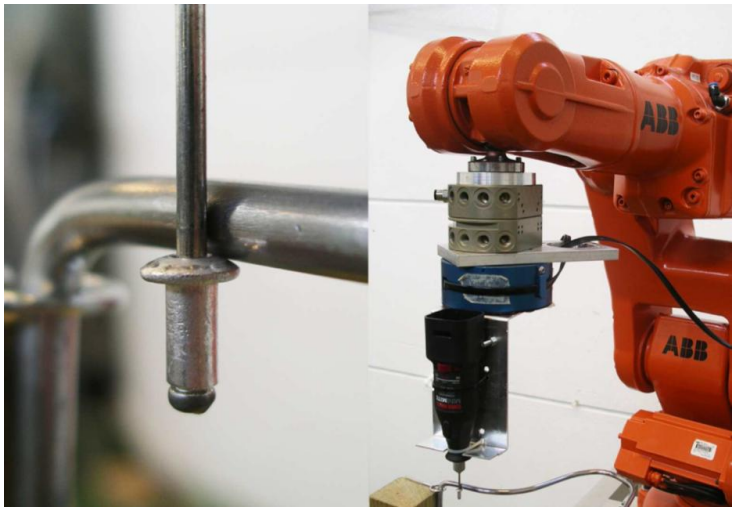
- Modelica
- Dynamic optimization and Optimica
- The JModelica.org open source project
- Applications
- What's next?

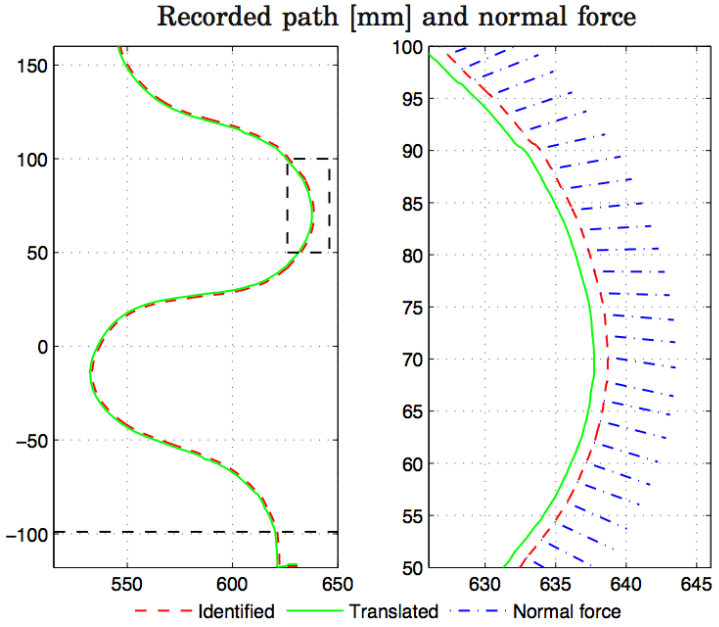
Grade change optimization

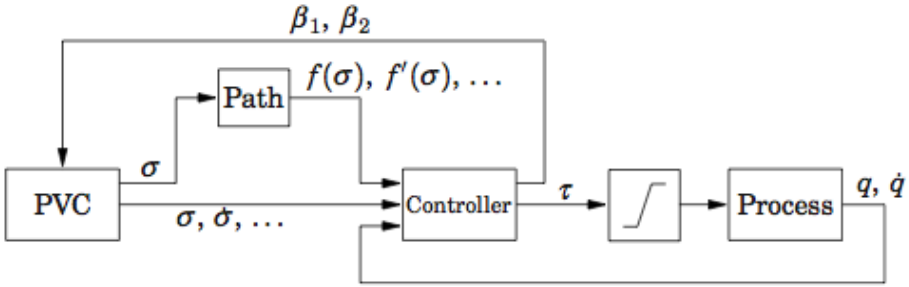
- Collaboration with plastics manufacturer Borealis
- Polyethylene production
- Tree reactors in series
  - Pre-loop
  - Loop
  - Gas phase
- Decision support
  - Flexible production
  - Raw material prices vary
  - Minimize off-spec
- JModelica.org for optimization
  - Grade changes and parameter estimation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

Time optimal robot control

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







Master's theses by Marin Hast, Björn Olofsson and Henrik Nilss

Outline

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

What's next?

- Improved Modelica compliance
  - Generics
  - Fluid modeling support
- Simulation
  - SUNDIALS
  - Hybrid systems
- Continued work on Python interface
  - Feedback appreciated
- Algorithm integration
- Application-driven development

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 (Texas A&M)
  - Explore the power of multi and many core
  - Decomposition schemes for interior point methods
- Safe refactoring
  - Modelica source code transformations
  - 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