Equation-Based Modeling and Simulation in Miking

Oscar Eriksson

Royal Institute of Technology KTH

2023-11-23

Outline

Equation-Based Modeling and Equation-Based Object-Oriented (EOO) Modeling Languges (MLs) by Example

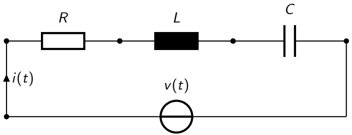
M-EOO Compiler Overview

Equation-Based Modeling and Simulation

A small RLC-circuit example

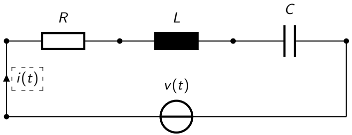
Equation-Based Modeling and Simulation

A small RLC-circuit example



Equation-Based Modeling and Simulation

A small RLC-circuit example

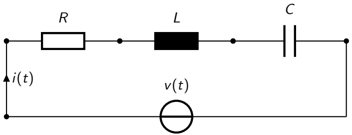


Example simulation trace

$i(t)$ over some time interval

Equation-Based Modeling and Simulation

A small RLC-circuit example

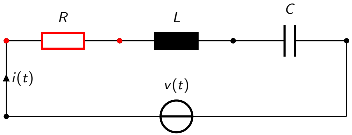


Component Equations

Example simulation trace

$i(t)$ over some time interval

Connection Equations



Component Equations

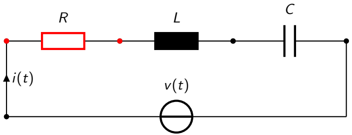
Example simulation trace

$i(t)$ over some time interval

Connection Equations

Equation-Based Modeling and Simulation

A small RLC-circuit example



Component Equations

$$u_R(t) = R \cdot i_R(t)$$

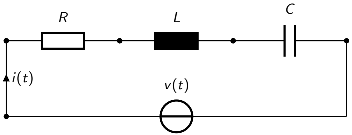
Example simulation trace

$i(t)$ over some time interval

Connection Equations

Equation-Based Modeling and Simulation

A small RLC-circuit example



Example simulation trace

$i(t)$ over some time interval

Component Equations

$$u_R(t) = R \cdot i_R(t)$$

$$u_L(t) = L \cdot \frac{d}{dt} i_L(t)$$

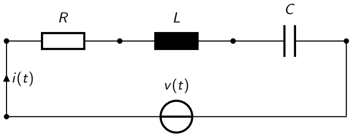
$$\frac{d}{dt} u_C(t) = C \cdot i_C(t)$$

$$u_V(t) = v(t)$$

Connection Equations

Equation-Based Modeling and Simulation

A small RLC-circuit example



Example simulation trace

$i(t)$ over some time interval

Component Equations

$$u_R(t) = R \cdot i_R(t)$$

$$u_L(t) = L \cdot \frac{d}{dt} i_L(t)$$

$$\frac{d}{dt} u_C(t) = C \cdot i_C(t)$$

$$u_V(t) = v(t)$$

Connection Equations

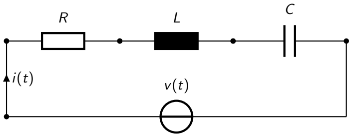
$$i_R(t) = i_L(t)$$

$$i_L(t) = i_C(t)$$

$$u_v(t) = u_R(t) + u_L(r) + u_C(t)$$

Equation-Based Modeling and Simulation

A small RLC-circuit example



Example simulation trace

$i(t)$ over some time interval

EOO ML

Component Equations

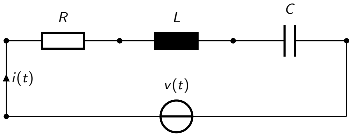
$$u_R(t) = R \cdot i_R(t)$$
$$u_L(t) = L \cdot \frac{d}{dt} i_L(t)$$
$$\frac{d}{dt} u_C(t) = C \cdot i_C(t)$$
$$u_V(t) = v(t)$$

Connection Equations

$$i_R(t) = i_L(t)$$
$$i_L(t) = i_C(t)$$
$$u_v(t) = u_R(t) + u_L(r) + u_C(t)$$

Equation-Based Modeling and Simulation

A small RLC-circuit example



Example simulation trace

$i(t)$ over some time interval

EOO ML

- Component equations in libraries

Component Equations

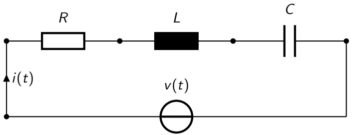
$$u_R(t) = R \cdot i_R(t)$$
$$u_L(t) = L \cdot \frac{d}{dt} i_L(t)$$
$$\frac{d}{dt} u_C(t) = C \cdot i_C(t)$$
$$u_V(t) = v(t)$$

Connection Equations

$$i_R(t) = i_L(t)$$
$$i_L(t) = i_C(t)$$
$$u_v(t) = u_R(t) + u_L(r) + u_C(t)$$

Equation-Based Modeling and Simulation

A small RLC-circuit example



Example simulation trace

$i(t)$ over some time interval

EOO ML

- Component equations in libraries
- Compiler **finds** connection equation

Component Equations

$$u_R(t) = R \cdot i_R(t)$$

$$u_L(t) = L \cdot \frac{d}{dt} i_L(t)$$

$$\frac{d}{dt} u_C(t) = C \cdot i_C(t)$$

$$u_V(t) = v(t)$$

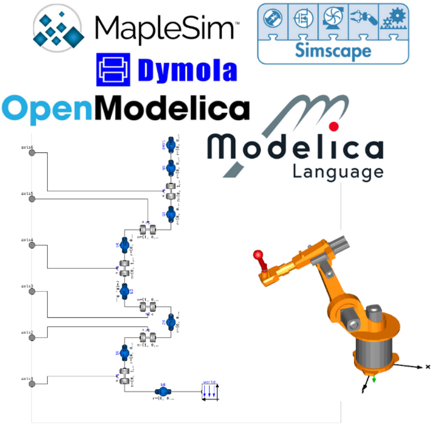
Connection Equations

$$i_R(t) = i_L(t)$$

$$i_L(t) = i_C(t)$$

$$u_v(t) = u_R(t) + u_L(r) + u_C(t)$$

EOO MLs in General



*From the Modelica standard library

- Established modeling paradigm
- E.g., modeling of vehicles, power plants, and aircraft

M-EOO, a functional EOO ML

- M-EOO is a *DSL* for *EOO modeling and simulation*
- *Statically typed* functional style EOO ML
- Implemented in the *Miking framework*
- Early stage of development
- Small standard library for analog circuits and 1D mechanics

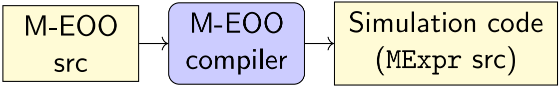
M-EOO, a functional EOO ML

- M-EOO is a *DSL* for *EOO modeling and simulation*
- *Statically typed* functional style EOO ML
- Implemented in the *Miking framework*
- Early stage of development
- Small standard library for analog circuits and 1D mechanics

M-EOO
src

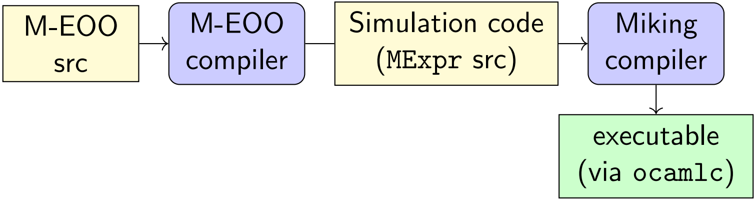
M-EOO, a functional EOO ML

- M-EOO is a *DSL* for *EOO modeling and simulation*
- *Statically typed* functional style EOO ML
- Implemented in the *Miking framework*
- Early stage of development
- Small standard library for analog circuits and 1D mechanics

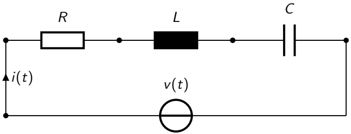


M-EOO, a functional EOO ML

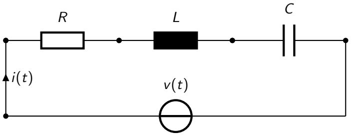
- M-EOO is a *DSL* for *EOO modeling and simulation*
- *Statically typed* functional style EOO ML
- Implemented in the *Miking framework*
- Early stage of development
- Small standard library for analog circuits and 1D mechanics



The RLC Circuit in M-EOO



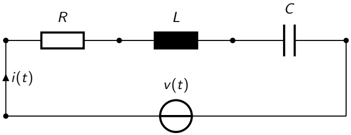
The RLC Circuit in M-EOO



RLC Circuit Model

```
main model
  def R = 0.2 end
  def L = 1.0 end
  def C = 1.0 end
  node n1,n2,n3,n4,n5
  var t,i:Real
equation
  t' = 1.0;
  Resistor(R,n1,n2);
  Inductor(L,n2,n3);
  Capacitor(C,n3,n4);
  VoltageSource(sigmoid(t,7.0),n5,n4);
  CurrentSensor(i,n5,n1)
output
  (t,i)
end
```

The RLC Circuit in M-EOO



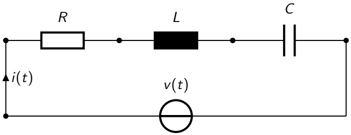
RLC Circuit Model

```
main model
  def R = 0.2 end
  def L = 1.0 end
  def C = 1.0 end
  node n1,n2,n3,n4,n5
  var t,i:Real
equation
  t' = 1.0;
  Resistor(R,n1,n2);
  Inductor(L,n2,n3);
  Capacitor(C,n3,n4);
  VoltageSource(sigmoid(t,7.0),n5,n4);
  CurrentSensor(i,n5,n1)
output
  (t,i)
end
```

Inductor Model

```
model Inductor:Real*Node*Node->Model
model Inductor(L,cathode,anode) =
var u,i:Real
equation
u = L*i';
connect cathode to anode in
Electrical with u across i through
end
```

The RLC Circuit in M-EOO



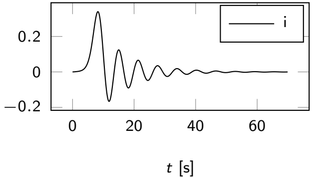
RLC Circuit Model

```
main model
  def R = 0.2 end
  def L = 1.0 end
  def C = 1.0 end
  node n1,n2,n3,n4,n5
  var t,i:Real
equation
  t' = 1.0;
  Resistor(R,n1,n2);
  Inductor(L,n2,n3);
  Capacitor(C,n3,n4);
  VoltageSource(sigmoid(t,7.0),n5,n4);
  CurrentSensor(i,n5,n1)
output
  (t,i)
end
```

Inductor Model

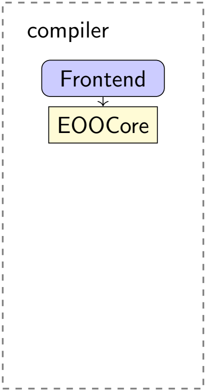
```
model Inductor:Real*Node*Node->Model
model Inductor(L,cathode,anode) =
var u,i:Real
equation
u = L*i';
connect cathode to anode in
Electrical with u across i through
end
```

Simulation Trace for i



The M-EOO Compiler Pipeline

The M-EOO Compiler Pipeline



←*parsing, desugaring, and typechecking*

←*extended subset of pure MExpr*

EOOCore

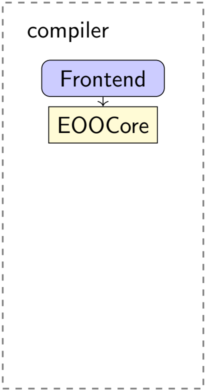
M-EOO model

```
main model
  def R = 0.2 end
  def L = 1.0 end
  def C = 1.0 end
  node n1,n2,n3,n4,n5
  var t,i:Real
equation
  t' = 1.0;
  Resistor(R,n1,n2);
  Inductor(L,n2,n3);
  Capacitor(C,n3,n4);
  VoltageSource(sigmoid(t,7.0),n5,n4);
  CurrentSensor(i,n5,n1)
output
  (t,i)
end
```

EOOCore program

```
let #var"R" = 0.2 in
let #var"L" = 1. in
let #var"C" = 1. in
let n5 = gensym {} in
let n4 = gensym {} in
let n3 = gensym {} in
let n2 = gensym {} in
let n1 = gensym {} in
let i: Float = gendynvarf "i" in
let t: Float = gendynvarf "t" in
(let eqn: [Equation] =
    concat
      [ eqnf (dotf 1 t) 1. ]
    (concat
      (#var"Resistor"
         (#var"R", n1, n2))
    -- ... more components ...
    (concat (#var"VoltageSource"
                 (sigmoid (t, 7.),
                  n5, n4))
      (#var"CurrentSensor"
          (i, n5, n1))))))
 in eqn, (t, i))
```

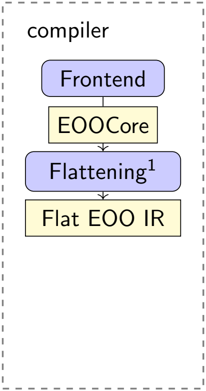
The M-EOO Compiler Pipeline



←*parsing, desugaring, and typechecking*

←*extended subset of pure MExpr*

The M-EOO Compiler Pipeline



←*parsing, desugaring, and typechecking*

←*extended subset of pure MExpr*

←*by partial evaluation*

←*equations and graphs*

[1] D. Broman, J. Siek. 2012. Modelyze: a Gradually Typed Host Language for Embedding Equation-Based Modeling Languages. Tech. Report

Flattening

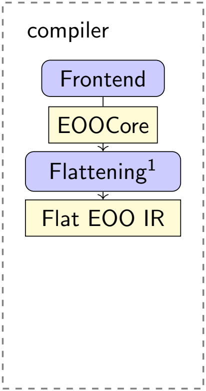
EOOCore program

```
let #var"R" = 0.2 in
let #var"L" = 1. in
let #var"C" = 1. in
let n5 = gensym {} in
let n4 = gensym {} in
let n3 = gensym {} in
let n2 = gensym {} in
let n1 = gensym {} in
let i: Float = gendynvarf "i" in
let t: Float = gendynvarf "t" in
(let eqn: [Equation] =
    concat
      [ eqnf (dotf 1 t) 1. ]
     (concat
       (#var"Resistor"
          (#var"R", n1, n2))
     -- ... more components ...
     (concat (#var"VoltageSource"
                 (sigmoid (t, 7.),
                  n5, n4))
       (#var"CurrentSensor"
           (i, n5, n1))))))
 in eqn, (t, i))
```

Flat EOO IR

```
-- ...
eqns:
  subf (dvar 1 uC)
       (mulf 1. (dvar 0 iC));
  subf (dvar 0 uL)
       (mulf 1. (dvar 1 iL));
  subf (dvar 0 uR)
       (mulf 0.2 (dvar 0 iR));
  subf (dvar 1 t) 1.

out:
  (dvar 0 t, dvar 0 i)

graphs:
-- ... Graph encoding connections
```

The M-EOO Compiler Pipeline



← *parsing, desugaring, and typechecking*

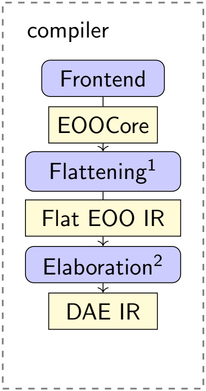
← *extended subset of pure MExpr*

← *by partial evaluation*

← *equations and graphs*

[1] D. Broman, J. Siek. 2012. Modelyze: a Gradually Typed Host Language for Embedding Equation-Based Modeling Languages. Tech. Report

The M-EOO Compiler Pipeline



← *parsing, desugaring, and typechecking*

← *extended subset of pure MExpr*

← *by partial evaluation*

← *equations and graphs*

← *adds connection equations*

← *Differential-Algebraic Equations*

---

[1] D. Broman, J. Siek. 2012. Modelyze: a Gradually Typed Host Language for Embedding Equation-Based Modeling Languages. Tech. Report

[2] E.g., J. McPhee. 1996. On the use of linear graph theory in multibody system dynamics. Nonlinear Dynamics

Elaboration

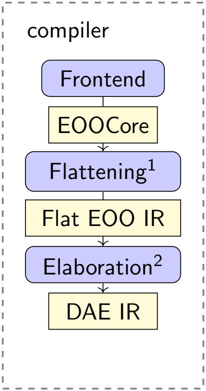
Flat EOO IR

```
-- ...
eqns:
  subf (dvar 1 uC)
       (mulf 1. (dvar 0 iC));
  subf (dvar 0 uL)
       (mulf 1. (dvar 1 iL));
  subf (dvar 0 uR)
       (mulf 0.2 (dvar 0 iR));
  subf (dvar 1 t) 1.

out:
  (dvar 0 t, dvar 0 i)

graphs:
-- ... Graph encoding connections
```

DAE IR

```
-- ...
eqns:
  subf
    (dvar 1 uC)
    (mulf 1. (dvar 0 iC));
  subf
    (dvar 0 uL)
    (mulf 1. (dvar 1 iL));
  subf
    (dvar 0 uR)
    (mulf 0.2 (dvar 0 iR));
  subf
    (dvar 1 t) 1.;
  subf
    (dvar 0 iR)
    (addf 0. (dvar 0 i));
  subf
    (dvar 0 iL)
    (addf 0. (dvar 0 i));
  subf
    (dvar 0 iC)
    (addf 0. (dvar 0 i));
-- ... Additional equation
out:
  (dvar 0 t, dvar 0 i)
```

The M-EOO Compiler Pipeline



←*parsing, desugaring, and typechecking*

←*extended subset of pure MExpr*

←*by partial evaluation*

←*equations and graphs*

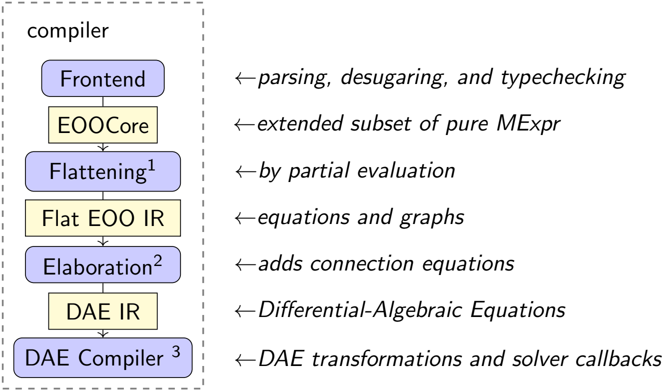
←*adds connection equations*

←*Differential-Algebraic Equations*

[1] D. Broman, J. Siek. 2012. Modelyze: a Gradually Typed Host Language for Embedding Equation-Based Modeling Languages. Tech. Report

[2] E.g., J. McPhee. 1996. On the use of linear graph theory in multibody system dynamics. Nonlinear Dynamics

The M-EOO Compiler Pipeline



[1] D. Broman, J. Siek. 2012. Modelyze: a Gradually Typed Host Language for Embedding Equation-Based Modeling Languages. Tech. Report

[2] E.g., J. McPhee. 1996. On the use of linear graph theory in multibody system dynamics. Nonlinear Dynamics

[3] O. Eriksson, V. Palmkvist, and D. Broman. 2023. Partial Evaluation of Automatic Differentiation for Differential-Algebraic Equations Solvers. (GPCE 2023)

Summary

I have presented an overview the EOO DSL M-EOO and its compiler

Prototype Implementation

`https://github.com/miking-lang/miking-dae` on the branch `eoo`