Modeling, Identification and Control, Vol. 37, No. 4, 2016, pp. 225–238, ISSN 1890–1328

Industrial Evaluation of Integrated Performance Analysis and Equation Model Debugging for Equation-Based Models

Å. Kinnander[1] M. Sjölund[2] A. Pop[2]

[1]*Siemens Turbo Machinery AB, Finspång, Sweden. E-mail:* ake.kinnander@outlook.com

[2]*Department of Computer and Information Science, Linköping University, Linköping, Sweden.*
E-mail: {martin.sjolund,adrian.pop}@liu.se

Abstract

The ease of use and the high abstraction level of equation-based object-oriented (EOO) languages such as Modelica has the drawback that performance problems and modeling errors are often hard to find. To address this problem, we have earlier developed advanced performance analysis and equation model debugging support in the OpenModelica tool. The aim of the work reported in this paper is to perform an independent investigation and evaluation of this equation model performance analysis and debugging methods and tool support on industrial models.

The results turned out to be mainly positive. The integrated debugger and performance analyzer locates several kinds of errors such as division by zero, chattering, etc., and greatly facilitates finding the equations that take most of the execution time during simulation.

It remains to further evaluate the performance profiler and debugger on even larger industrial models.

Keywords: profiler, debugging, Modelica, industrial, OpenModelica

1 Introduction

The development of today's complex products requires integrated environments and equation-based object-oriented declarative (EOO) languages such as Modelica (Modelica Association, 2014; Fritzson, 2015) for modeling and simulation.

The increased ease of use, the high abstraction, and the expressivity of such languages are very attractive properties. However, the drawback of this high-level approach is that understanding the causes of unexpected behavior, slow performance, and numerical errors of certain simulation models is very difficult, in particular for users who are not experts in simulation methods.

Therefore Pop et al. (2014) have recently developed an advanced equation model debugger for the Modelica language, as part of the OpenModelica (Open Source Modelica Consortium, 2016) tool. This is quite different from debuggers of conventional algorithmic programming language debuggers (Stallman et al., 2014; Nethercote and Seward, 2007; Zeller, 2009). Pop and Fritzson (2005) developed a debugger for the algorithmic subset of Modelica and Bunus (2004) developed a debugger that analyzes the causes of over-constrained or under-constrained systems of equations. The new debugger is also based on the recent development of the advanced bootstrapped OpenModelica compiler (Sjölund et al., 2014).

The applications used for evaluation perform simulation of combined cycle power plants. This involves the dynamics of water cycling from water to steam and back while streaming in different flow regimes through pipes, valves and volumes, affecting the heat transfer from the flue gases. To handle these rather complicated phenomena including boiling and condensation in and on tubes, accurate dynamic models often require high computation power, efficient programming as well as a good balance between accuracy and computational speed in the aspect of simulation purposes. The performance analyzer, also called profiler, which is a tool that informs where in user equations CPU power is spent and gives thereby possibility to evaluate different mathematical methods and make deliberated trading between accuracy and computational speed. As described in Sjölund (2015), the techniques used when profiling Modelica equation-based models are quite different from profiling of general programs (Graham et al., 1983). Some earlier more limited approaches to profiling Modelica models are presented by Huhn et al. (2011) and Schulze et al. (2010).

The integrated equation model debugger has been evaluated by the designers and performs well on both small and big models. However, an independent evaluation of the integrated performance analyzer and debugger by industrial users on industrial problems was still missing. Such an evaluation is the main topic of this paper. We have earlier made a preliminary industrial evaluation only of the debugging functionality (Kinnander et al., 2016). This paper presents an evaluation of the integrated performance analysis and debugging methods and tool, including a slightly updated version of the debugging evaluation results presented by Kinnander et al. (2016).

The rest of the paper is structured as follows: first the errors to be investigated and models to be evaluated are briefly presented. Section 2 introduces the debugger tests in more detail. Section 3 presents debugging of errors in the logarithmic temperature calculation whereas Section 4 presents debugging of errors due to bad initial values. Section 5 presents the performance analyzer and its use. Finally, Section 6 presents conclusions.

1.1 Errors to be Investigated

In order to investigate different types of errors that could be expected to occur, a small and simple evaporator model is used. This has been fetched from a larger model used for transient analysis of combined power plants by Siemens Industrial Turbomachinery AB in Finspång, Sweden. The following errors are to be investigated:

1. Division by zero
2. Errors in the average logarithmic temperature difference used for heat transfer calculation:
    a) Inlet temperature difference =0
    b) Inlet temperature difference=outlet temperature difference.
3. Boiling in the evaporator that causes halt of simulation progress by much too small time steps (stiffness)
4. Various test of bad initial values, with variation of pressure, temperatures, flows and masses in the different parts of the process.

The model selected is a simplified model of an error free model, hence the above test will be deliberately inserted and the debugging tool will only be examined by its outputs, while a sharper application for a real model development where errors are unknown and the debugger support for identifying them will be more apparent, will be carried out later. The reason for this is the limited time available for testing, and that a sharp application will only provide stochastic errors and could thereby not be planned in time.

1.2 Models for the Debugger and Performance Profiler Evaluation

The evaporator test model shown in Figure 1 will be used for the investigation, containing an instance (`Evap`) of the `Evaporator` model shown in the middle of the connection diagram.

It consists of an evaporator model that has flue gases as heating source and water as coolant, producing dry steam to the steam sink. The steam production is decided by the heat from the flue gases, the enthalpy (temperature and pressure) of the water source (`FWpump`), and the steam extraction to the steam sink (`SteamSink`) that in turn is tracking the evaporators drum pressure with a negative bias of 0.1 to 1 bar. The model has 1110 equations.

The evaporator model is designed according to Figure 2.

The evaporator model has a drum model (`Drum`), a heat exchanger (`Hex`) and a level controller (`DLC`) that controls level by a control valve (`FW_CV`). The level controller is no actual water level controller in length units (m), instead it controls the amount of water (kg) in the drum. The thermal capacity of the drum metals is represented of a heat capacity model (`DrumWallHeatMass`) but the insulation towards surroundings are assumed to be perfect, i.e. no heat losses to the outside of the drum. This model has 809 equations.

The drum model is the volume model from the Fluid library manipulated to only let steam exit, i.e., always perfect separation. The heat exchanger is according to Figure 3.

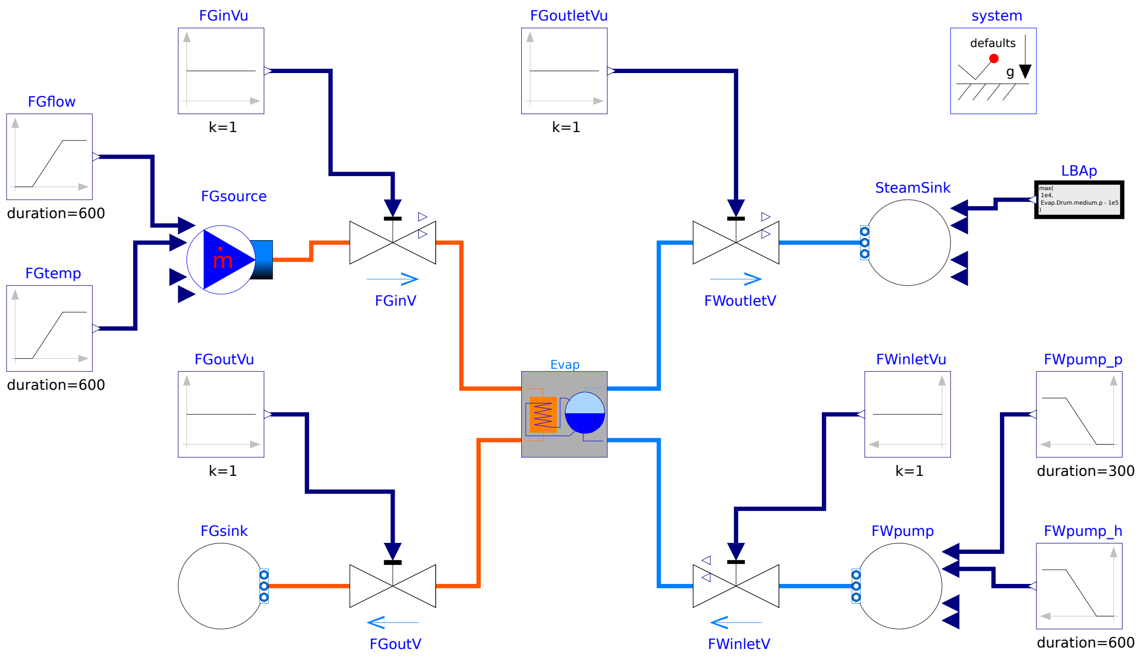


Figure 1: Evaporator test model

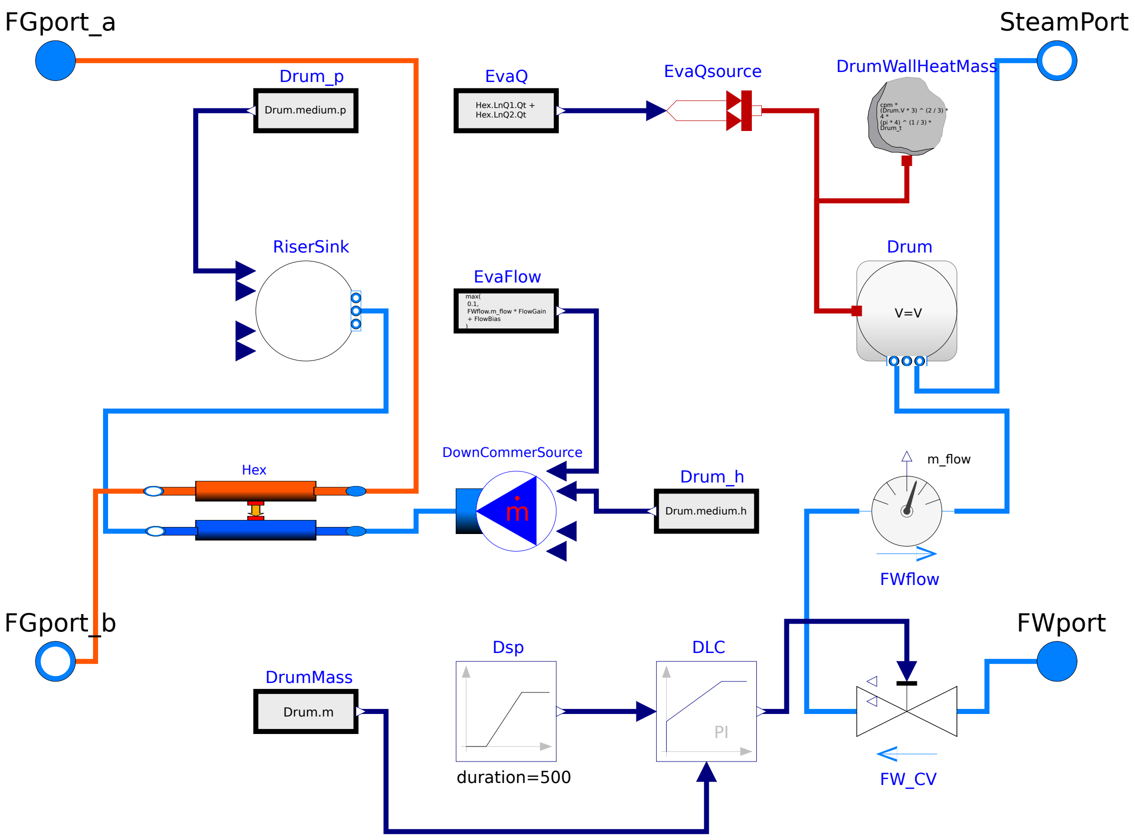


Figure 2: Evaporator model

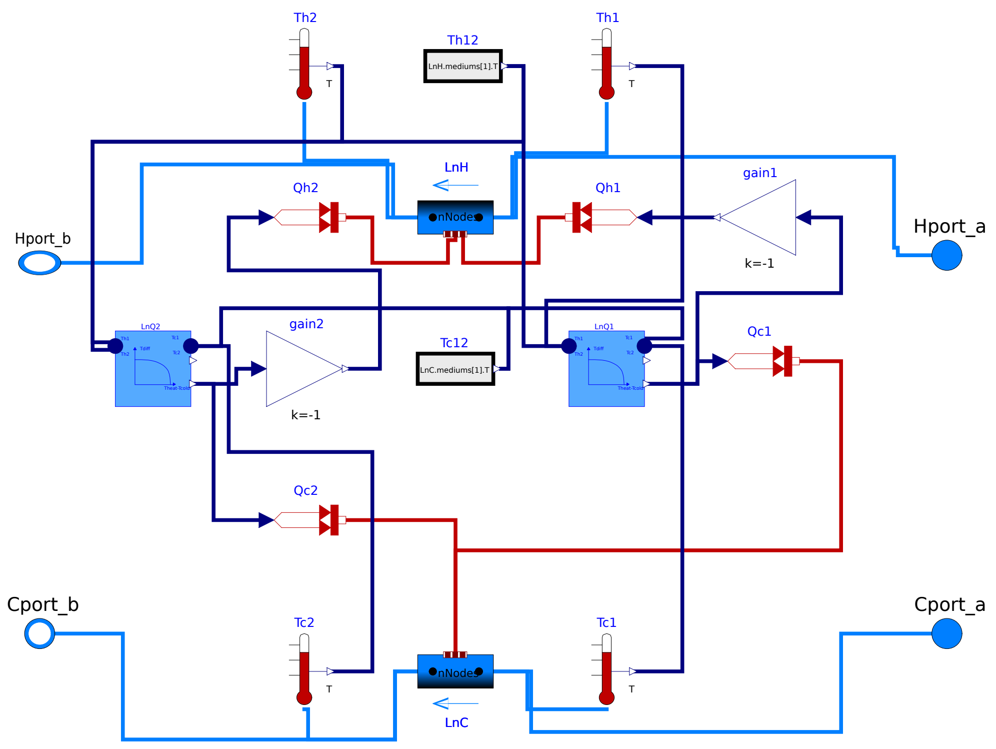


Figure 3: Heat exchanger

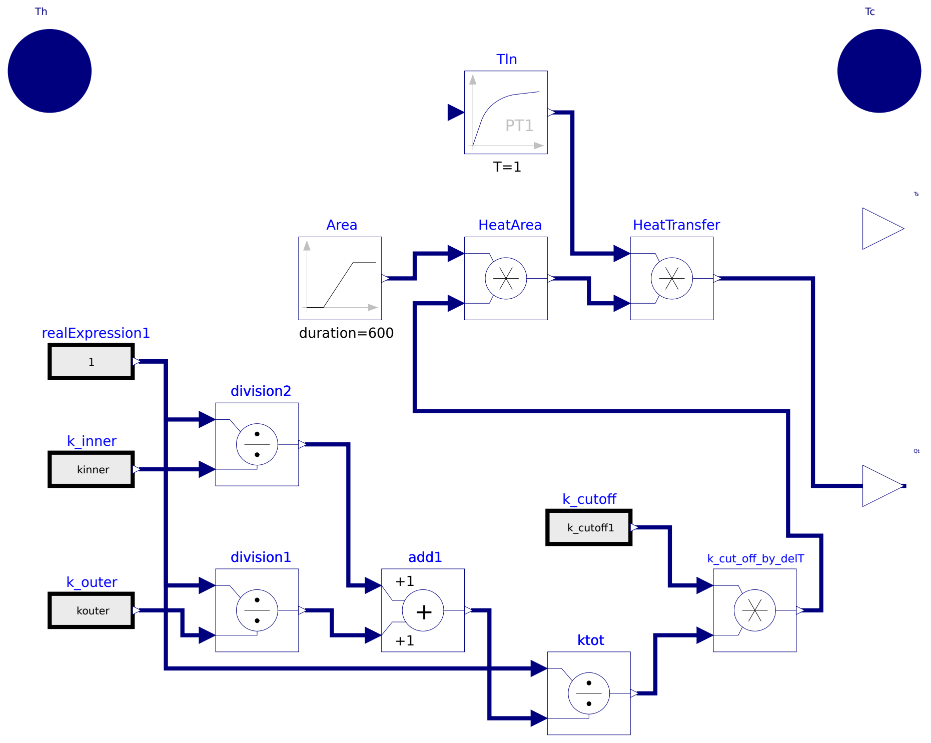


Figure 4: Heat transfer calculation model

The heat exchanger has a pipe model (`LnH`) that corresponds to the flue gas channel and a pipe model (`LnC`) that corresponds to the pipe bundle carrying the water to be heated. The heat exchanger has parallel flows. In this simplified model the overall heat transfer coefficient ($J/m^2/K$) is a constant, i.e., it takes not configuration or medium properties into account. The driving temperature difference is calculated for respectively inlet and outlet sections of the pipe bundles (i.e. they are configured with 2 nodes each) by the models `LnQ1` and `LnQ2`. The temperatures are measured besides inlets and outlets for both pipe models also in the middle (`Tc12` and `Th12`). This model has 580 equations.

Finally, the heat calculation models `LnQ1` and `LnQ2` are according to Figure 4. All other models are from the standard Modelica library.

The model has a low pass filter with an input connected in the text layer where the logarithmic temperature difference from the connectors `Th` and `Tc` which both are connecting both inlet and outlet temperatures respectively medium side, is calculated. The output of the model is the calculated heat transfer (W). The overall heat transfer coefficient `ktot` is calculated from parameters representing the heat transfer coefficient from flue gas to metal, `k_outer`, and from metal to water, `k_inner`. In the present full size boiler model those parameters are replaced with connectors that provide more accurate values, based on medium and flow properties calculated in separate models. The heat transfer area is a ramp with selectable duration and height values, to be adopted to what is needed from initializing aspect (duration of suitable size respectively to the real heat transfer area).

This model contributes with 39 equations of the total of 1110 equations.

2 Debugger Tests

2.1 Activation of Debugger

The debugger is activated by setting the flag « Launch transformational debugger ». After a successful simulation the output windows are containing the following information (Figure 5).

The simulation output window contains assertion violation messages that are false, because the enthalpy flow H (W) has too narrow range in the Standard Modelica library. It ought to be at least 10 times as big.

This violation has no influence on the simulation result (might there be an unnecessary delay?). The window shows with a green bar that 100 % of simulation is done and the blue text that it has been successful. The transformational debugger window shows all variables in the variable browsers window and all equations in the equation browser window, as found in the simulation code. All other frames in the debugger are empty.

2.2 Division by Zero by Parameter Setting

The test is done by setting parameter `k_inner` to zero. The simulation output window displays the following messages (Figure 6).

The simulation output window gives the required information that simulation crashed at initialization due to an assertion that avoids division by zero and this is caused by `k_inner=0`.The debugger window looks as before but after clicking debug more in the simulation output window it looks as in (Figure 6).

The equation browser marks initial equation with index 102: `Evap.Hex.LnQ1.add1.u1 := DIVISION(1.0, Evap.Hex.LnQ1.kinner)`, which is the same information as in simulation output window. The frame denoted `Defines` gives the variable that becomes undefined by the zero division. The frame denoted `Depends` give the variable name `Evap.Hex.LnQ1.k_inner`. For this error the debugger gives all information necessary.

2.3 Division by Zero by Time Function

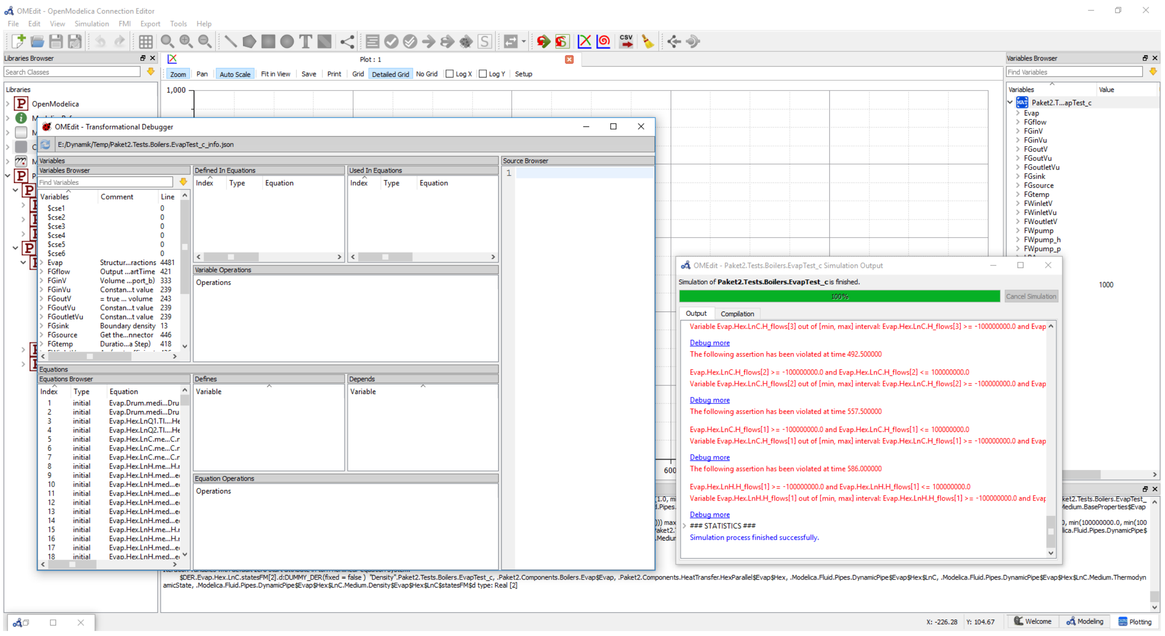
The `k_inner` variable is replaced by a time function that ramps it down to zero in 100 seconds. This results in a never ending simulation.

The solver manages to pass the 100 s time point where `k_inner` is zero and a division by zero occurs. No plots are available but the ramp proceeds to negative values for `k_inner`. The solver has skipped the exact 100 s time point, but then continued into other problems, due to the negative `k_inner` value. On the passage it has however produced two messages about zero division at time 100 when they occurred. In the case of the user being unaware of the division problem, the large amount of output in the simulation output window hides those messages.

For a ramped denominator passing zero, the debugger is not optimal in case the solver manages to pass the critical point and that consecutive errors then hide the information from the user. A solution would be the option to let the user decide if division by zero should be accepted or not, i.e., the solver should then interrupt and save when any denominator having a passage of zero.

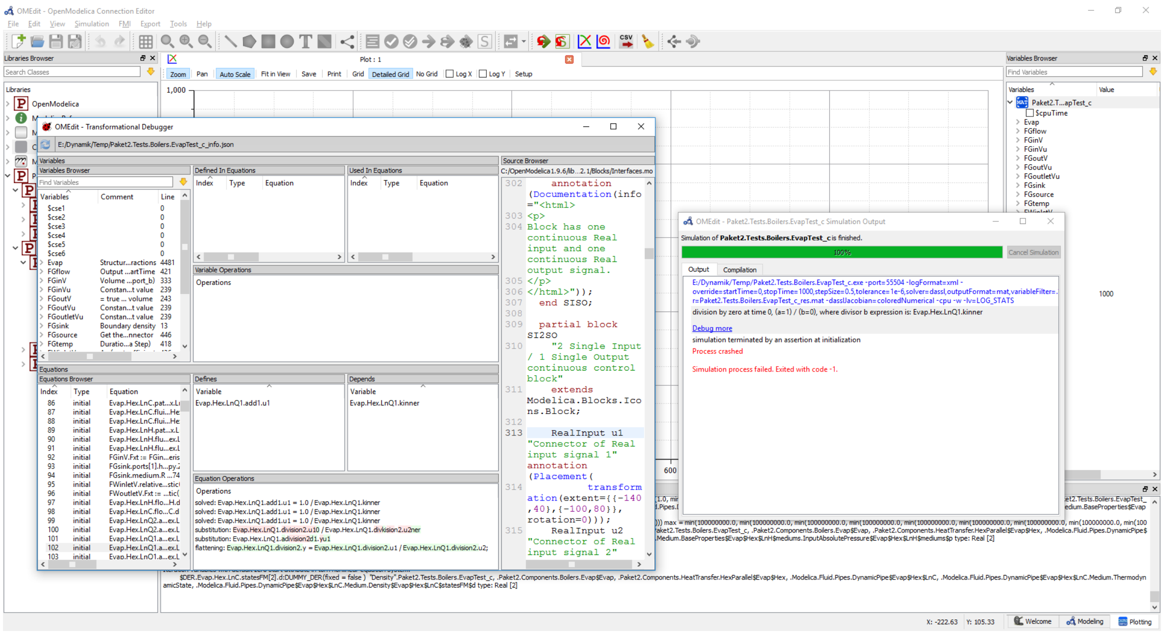
2.4 Division by Zero due to Mismatch in Parameter Settings

By deselection of the heat transfer used in the `LnC` pipe in the Hex model (Figure 3) the test model still checks OK, but now with only 1106 equations instead of 1110.



```
Simulation of Paket2.Tests.Boilers.EvapTest_c is finished.
100%

Variable Evap.Hex.LnC.H_flows[3] out of [min, max] interval: Evap.Hex.LnC.H_flows[3] >= -100000000.0 and Evap
Debug more
The following assertion has been violated at time 492.500000

Evap.Hex.LnC.H_flows[2] >= -100000000.0 and Evap.Hex.LnC.H_flows[2] <= 100000000.0
Variable Evap.Hex.LnC.H_flows[2] out of [min, max] interval: Evap.Hex.LnC.H_flows[2] >= -100000000.0 and Evap
Debug more
The following assertion has been violated at time 557.500000

Evap.Hex.LnC.H_flows[1] >= -100000000.0 and Evap.Hex.LnC.H_flows[1] <= 100000000.0
Variable Evap.Hex.LnC.H_flows[1] out of [min, max] interval: Evap.Hex.LnC.H_flows[1] >= -100000000.0 and Evap
Debug more
The following assertion has been violated at time 586.000000

Evap.Hex.LnH.H_flows[1] >= -100000000.0 and Evap.Hex.LnH.H_flows[1] <= 100000000.0
Variable Evap.Hex.LnH.H_flows[1] out of [min, max] interval: Evap.Hex.LnH.H_flows[1] >= -100000000.0 and Evap
Debug more
### STATISTICS ###
Simulation process finished successfully.
```

Figure 5: Information from OpenModelica with debugger activated at successful simulation

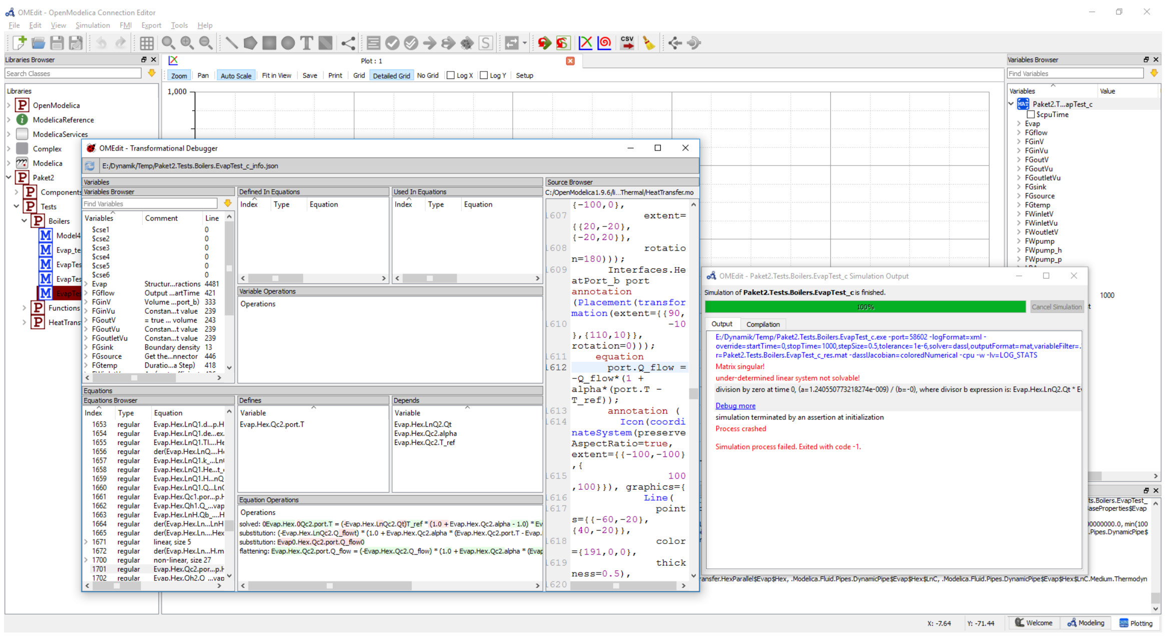


```
Defines
Variable
Evap.Hex.LnQ1.add1.u1

Depends
Variable
Evap.Hex.LnQ1.kinner

Operations
solved: Evap.Hex.LnQ1.add1.u1 = 1.0 / Evap.Hex.LnQ1.kinner
solved: Evap.Hex.LnQ1.add1.u1 = 1.0 / Evap.Hex.LnQ1.kinner
solved: Evap.Hex.LnQ1.add1.u1 = 1.0 / Evap.Hex.LnQ1.kinner
substitution: Evap.Hex.LnQ1.division2.u10 / Evap.Hex.LnQ1.divkision2.u2ner
substitution: Evap.Hex.LnQ1.adivision2d1.yu1
flattening: Evap.Hex.LnQ1.division2.y = Evap.Hex.LnQ1.division2.u1 / Evap.Hex.LnQ1.division2.u2;
```

```
    annotation (Documentation(info="<html>
<p>
Block has one continuous Real input and one continuous Real output signal.
</p>
</html>"));
  end SISO;

  partial block SI2SO
    "2 Single Input / 1 Single Output continuous control block"
    extends Modelica.Blocks.Icons.Block;

    RealInput u1 "Connector of Real input signal 1" annotation (Placement(
          transformation(extent={{-140,40},{-100,80}}, rotation=0)));
    RealInput u2 "Connector of Real input signal 2"
```

```
Simulation of Paket2.Tests.Boilers.EvapTest_c is finished.
100%

E:/Dynamik/Temp/Paket2.Tests.Boilers.EvapTest_c.exe -port=55504 -logFormat=xml -
override=startTime=0,stopTime=1000,stepSize=0.5,tolerance=1e-6,solver=dassl,outputFormat=mat,variableFilter=.
r=Paket2.Tests.Boilers.EvapTest_c_res.mat -dasslJacobian=coloredNumerical -cpu -w -lv=LOG_STATS
division by zero at time 0, (a=1) / (b=0), where divisor b expression is: Evap.Hex.LnQ1.kinner
Debug more
simulation terminated by an assertion at initialization
Process crashed

Simulation process failed. Exited with code -1.
```

Figure 6: Debugger activated after division by zero by parameter setting.



```
Simulation of Paket2.Tests.Boilers.EvapTest_c is finished.
E:/Dynamik/Temp/Paket2.Tests.Boilers.EvapTest_c.exe -port=58602 -logFormat=xml -
override=startTime=0,stopTime=1000,stepSize=0.5,tolerance=1e-6,solver=dassl,outputFormat=mat,variableFilter=.
r=Paket2.Tests.Boilers.EvapTest_c_res.mat -dasslJacobian=coloredNumerical -cpu -w -lv=LOG_STATS
Matrix singular!
under-determined linear system not solvable!
division by zero at time 0, (a=1.240550773218274e-009) / (b=-0), where divisor b expression is: Evap.Hex.LnQ2.Qt * E
Debug more
simulation terminated by an assertion at initialization
Process crashed

Simulation process failed. Exited with code -1.
```

```
Defines: Evap.Hex.Qc2.port.T
Depends: Evap.Hex.LnQ2.Qt
         Evap.Hex.Qc2.alpha
         Evap.Hex.Qc2.T_ref
Equation Operations
solved: 0Evap.Hex.0Qc2.port.T = (-Evap.Hex.LnQc2.Qt)T_ref * (1.0 + Evap.Hex.Qc2.alpha - 1.0) * Ev
substitution: (-Evap.Hex.LnQc2.Q_flowt) * (1.0 + Evap.Hex.Qc2.alpha * (Evap.Hex.Qc2.port.T - Evap.
substitution: Evap0.Hex.Qc2.port.Q_flow0
flattening: Evap.Hex.Qc2.port.Q_flow = (-Evap.Hex.Qc2.Q_flow) * (1.0 + Evap.Hex.Qc2.alpha * (Evap
```

Figure 7: Outputs after no use of heat transfer specified but corresponding heat transfer connectors any way connected

Simulation gives the following simulation output window and transformational debugger window (Figure 7).

The simulation crashes at initialization due to division by zero where the denominator is involving the variables `Qt` and `alpha`. A check of the model reveals that `alpha` is the heat transfer coefficient to surroundings and is deliberately zero. That is, this model is impossible to run although it checks OK. The debugger points to the equation numbered 1258. Marking that equation points to a source code line 1612 where the Modelica standard library (MSL) component `PrescribedHeatFlow` (PHF) equals the heat ports heat flow (`Q_Flow`) to the connector input `Q_Flow` (the prescribed flow). This equation contains also a dependence on `alpha`, but with `alpha`=0 the equation should be OK: `port.Q_Flow=-Q_Flow`. The PHF model calculates its internal temperature in order to be able to have the prescribed heat flow by this equation (index 1258). This temperature should be the same as the connected pipes temperature as there is no thermal resistance in the connection itself so why is it calculated in this way? One could not be sure that any user would understand that this equation is unexpected and caused by a wrong parameter setting in the pipe model.

It would have been better if there would be a consistency check between connectors and their use in the model. One way would be not to show the connector unless it is activated, or give an error message if it is anyway connected. Could the variables browser shed some light over the problem? The variables browser is presented in (Figure 8) after the port temperature has been clicked.

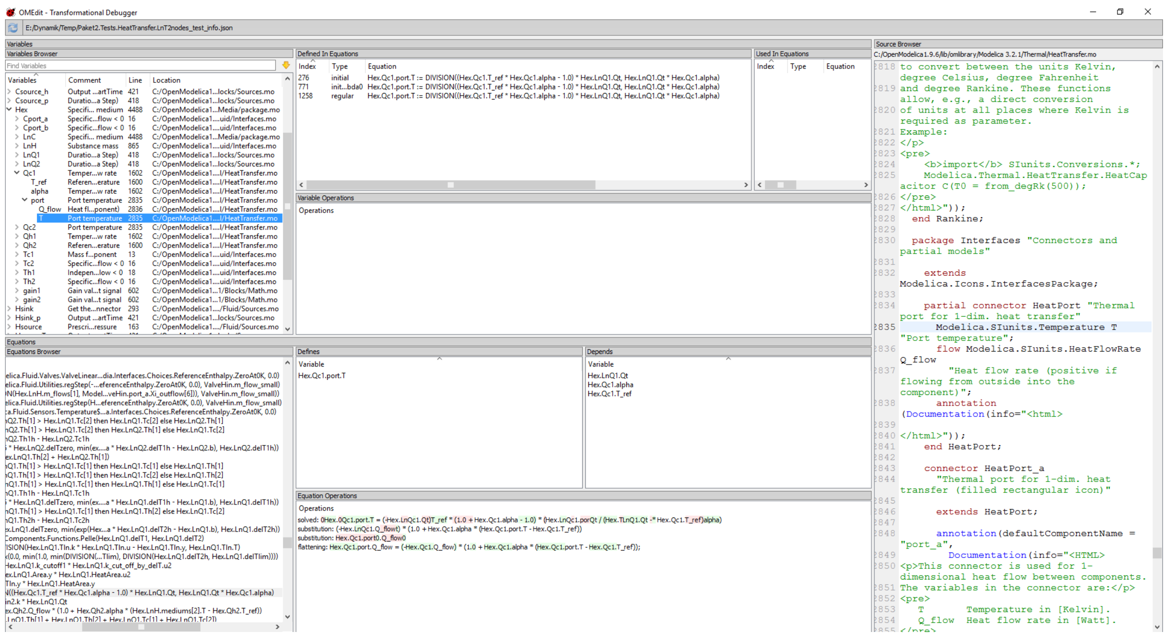
It gives the same information augmented with the initial equations. A question is why the debugger is not pointing to the initial equations indices 276 and 771, as the simulation output window tells that the error appeared at initialization.

3 Errors in the Logarithmic Temperature Difference Calculation

Errors in the logarithmic temperature difference calculation should be treated the same way as the division by zero. Interesting is however, if the solver also for this type of errors manage to pass the critical point as their time duration could be expected to be very short. Basically this investigation is more an investigation of the solver and not the debugger but the debugger will be activated and therefore also this investigation is a part of the paper.

3.1 Temperature Differences Passing Zero

This test is achieved by removing the numerical fences that prevent zero crossing. Unfortunately, it turns out that there are no crossings that passes `delT=0` for the case simulated, and the test needs further work to be



E:/Dynamik/Temp/Paket2.Tests.HeatTransfer.LnT2nodes_test_info.json

Variables	Comment	Line	Location
Csource_h	Output ...artTime	421	C:/OpenModelica1...locks/Sources.mo
Csource_p	Duratio...a Step)	418	C:/OpenModelica1...locks/Sources.mo
Hex	Specifi... medium	4488	C:/OpenModelica1...Media/package.mo
Cport_a	Specific...flow < 0	16	C:/OpenModelica1....uid/Interfaces.mo
Cport_b	Specific...flow < 0	16	C:/OpenModelica1....uid/Interfaces.mo
LnC	Specifi... medium	4488	C:/OpenModelica1...Media/package.mo
LnH	Substance mass	865	C:/OpenModelica1....uid/Interfaces.mo
LnQ1	Duratio...a Step)	418	C:/OpenModelica1...locks/Sources.mo
LnQ2	Duratio...a Step)	418	C:/OpenModelica1...locks/Sources.mo
Qc1	Temper...w rate	1602	C:/OpenModelica1....l/HeatTransfer.mo
T_ref	Referen...erature	1600	C:/OpenModelica1....l/HeatTransfer.mo
alpha	Temper...w rate	1602	C:/OpenModelica1....l/HeatTransfer.mo
port	Port temperature	2835	C:/OpenModelica1....l/HeatTransfer.mo
Q_flow	Heat fl...ponent)	2836	C:/OpenModelica1....l/HeatTransfer.mo
T	Port temperature	2835	C:/OpenModelica1....l/HeatTransfer.mo
Qc2	Port temperature	2835	C:/OpenModelica1....l/HeatTransfer.mo
Qh1	Temper...w rate	1602	C:/OpenModelica1....l/HeatTransfer.mo
Qh2	Referen...erature	1600	C:/OpenModelica1....l/HeatTransfer.mo
Tc1	Mass f...ponent	13	C:/OpenModelica1....uid/Interfaces.mo
Tc2	Specific...flow < 0	16	C:/OpenModelica1....uid/Interfaces.mo
Th1	Indepen...low < 0	18	C:/OpenModelica1....uid/Interfaces.mo
Th2	Specific...flow < 0	16	C:/OpenModelica1....uid/Interfaces.mo
gain1	Gain val...t signal	602	C:/OpenModelica1...1/Blocks/Math.mo
gain2	Gain val...t signal	602	C:/OpenModelica1...1/Blocks/Math.mo
Hsink	Get the...nnector	293	C:/OpenModelica1..../Fluid/Sources.mo
Hsink_p	Output ...artTime	421	C:/OpenModelica1...locks/Sources.mo
Hsource	Prescri...ressure	163	C:/OpenModelica1..../Fluid/Sources.mo

Defined In Equations

Index	Type	Equation
276	initial	Hex.Qc1.port.T := DIVISION((Hex.Qc1.T_ref * Hex.Qc1.alpha - 1.0) * Hex.LnQ1.Qt, Hex.LnQ1.Qt * Hex.Qc1.alpha)
771	init...bda0	Hex.Qc1.port.T := DIVISION((Hex.Qc1.T_ref * Hex.Qc1.alpha - 1.0) * Hex.LnQ1.Qt, Hex.LnQ1.Qt * Hex.Qc1.alpha)
1258	regular	Hex.Qc1.port.T := DIVISION((Hex.Qc1.T_ref * Hex.Qc1.alpha - 1.0) * Hex.LnQ1.Qt, Hex.LnQ1.Qt * Hex.Qc1.alpha)

Used In Equations

Index	Type	Equation

Variable Operations

Operations

Equations Browser

```
elica.Fluid.Valves.ValveLinear...dia.Interfaces.Choices.ReferenceEnthalpy.ZeroAt0K, 0.0)
elica.Fluid.Utilities.regStep(-...eferenceEnthalpy.ZeroAt0K, 0.0), ValveHin.m_flow_small)
N(Hex.LnH.m_flows[1], Model...veHin.port_a.Xi_outflow[6])), ValveHin.m_flow_small))
elica.Fluid.Utilities.regStep(H...eferenceEnthalpy.ZeroAt0K, 0.0), ValveHin.m_flow_small)
a.Fluid.Sensors.Temperature$...a.Interfaces.Choices.ReferenceEnthalpy.ZeroAt0K, 0.0)
nQ2.Th[1] > Hex.LnQ1.Tc[2] then Hex.LnQ1.Tc[2] else Hex.LnQ2.Th[1]
nQ2.Th[1] > Hex.LnQ1.Tc[2] then Hex.LnQ2.Th[1] else Hex.LnQ1.Tc[2]
nQ2.Th1h - Hex.LnQ2.Tc1h
 * Hex.LnQ2.delTzero, min(ex....a * Hex.LnQ2.delT1h - Hex.LnQ2.b), Hex.LnQ2.delT1h))
ex.LnQ1.Th[2] + Hex.LnQ2.Th[1])
nQ1.Th[1] > Hex.LnQ1.Tc[1] then Hex.LnQ1.Tc[1] else Hex.LnQ1.Th[1]
nQ1.Th[1] > Hex.LnQ1.Tc[1] then Hex.LnQ1.Tc[2] else Hex.LnQ1.Th[2]
nQ1.Th[1] > Hex.LnQ1.Tc[1] then Hex.LnQ1.Th[1] else Hex.LnQ1.Tc[1]
nQ1.Th1h - Hex.LnQ1.Tc1h
 * Hex.LnQ1.delTzero, min(ex....a * Hex.LnQ1.delT1h - Hex.LnQ1.b), Hex.LnQ1.delT1h))
nQ1.Th[1] > Hex.LnQ1.Tc[1] then Hex.LnQ1.Th[2] else Hex.LnQ1.Tc[2]
nQ1.Th2h - Hex.LnQ1.Tc2h
ex.LnQ1.delTzero, min(exp(Hex...a * Hex.LnQ1.delT2h - Hex.LnQ1.b), Hex.LnQ1.delT2h))
Components.Functions.Pelle(Hex.LnQ1.delT1, Hex.LnQ1.delT2)
ISION(Hex.LnQ1.TIn.k * Hex.LnQ1.TIn.u - Hex.LnQ1.TIn.y, Hex.LnQ1.TIn.T)
(0.0, min(1.0, min(DIVISION(...Tlim), DIVISION(Hex.LnQ1.delT2h, Hex.LnQ1.delTlim))))
Hex.LnQ1.k_cutoff1 * Hex.LnQ1.k_cut_off_by_delT.u2
ex.LnQ1.Area.y * Hex.LnQ1.HeatArea.u2
TIn.y * Hex.LnQ1.HeatArea.y
N((Hex.Qc1.T_ref * Hex.Qc1.alpha - 1.0) * Hex.LnQ1.Qt, Hex.LnQ1.Qt * Hex.Qc1.alpha)
in2.k * Hex.LnQ1.Qt
ex.Qh2.Q_flow * (1.0 + Hex.Qh2.alpha * (Hex.LnH.mediums[2].T - Hex.Qh2.T_ref))
.LnQ1.Th[1] + Hex.LnQ1.Th[2] + Hex.LnQ1.Tc[1] + Hex.LnQ1.Tc[2])
```

Defines

Variable
Hex.Qc1.port.T

Depends

Variable
Hex.LnQ1.Qt
Hex.Qc1.alpha
Hex.Qc1.T_ref

Equation Operations

```
Operations
solved: 0Hex.0Qc1.port.T = (-Hex.LnQc1.Qt)T_ref * (1.0 + Hex.Qc1.alpha - 1.0) * (Hex.LnQc1.porQt / (Hex.TLnQ1.Qt -* Hex.Qc1.T_ref)alpha)
substitution: (-Hex.LnQc1.Q_flowt) * (1.0 + Hex.Qc1.alpha * (Hex.Qc1.port.T - Hex.Qc1.T_ref))
substitution: Hex.Qc1.port0.Q_flow0
flattening: Hex.Qc1.port.Q_flow = (-Hex.Qc1.Q_flow) * (1.0 + Hex.Qc1.alpha * (Hex.Qc1.port.T - Hex.Qc1.T_ref));
```

Source Browser

C:/OpenModelica1.9.6/lib/omlibrary/Modelica 3.2.1/Thermal/HeatTransfer.mo

```
to convert between the units Kelvin,
degree Celsius, degree Fahrenheit
and degree Rankine. These functions
allow, e.g., a direct conversion
of units at all places where Kelvin is
required as parameter.
Example:
</p>
<pre>
    <b>import</b> SIunits.Conversions.*;
    Modelica.Thermal.HeatTransfer.HeatCap
acitor C(T0 = from_degRk(500));
</pre>
</html>"));
  end Rankine;

  package Interfaces "Connectors and
partial models"

    extends
Modelica.Icons.InterfacesPackage;

    partial connector HeatPort "Thermal
port for 1-dim. heat transfer"
      Modelica.SIunits.Temperature T
"Port temperature";
      flow Modelica.SIunits.HeatFlowRate
Q_flow
        "Heat flow rate (positive if
flowing from outside into the
component)";
      annotation
(Documentation(info="<html>

</html>"));
    end HeatPort;

    connector HeatPort_a
      "Thermal port for 1-dim. heat
transfer (filled rectangular icon)"

      extends HeatPort;

      annotation(defaultComponentName =
"port_a",
        Documentation(info="<HTML>
<p>This connector is used for 1-
dimensional heat flow between components.
The variables in the connector are:</p>
<pre>
   T       Temperature in [Kelvin].
   Q_flow  Heat flow rate in [Watt].
```

Figure 8: Information from the debugger variables browser

carried out, and therefore postponed and not published in this paper. This is unexpected and errors might have occurred when moving the model from another tool to OpenModelica.

3.2 Temperature Differences at Outlet and Inlet Passing Equal Values

This is happening without any numerical problems, i.e., the solver skips the critical time where they are equal or happens to avoid it without any actions.

4 Bad Initial Values or Bad Simulation Boundaries

The debugger support, if any, is to be investigated for this type of problems where simulation runs into numerical problems.

4.1 Too high Backpressure

By increasing the back pressure from the steam pipes to exceed drum pressure, and thus preventing steam flow out of the drum the simulation terminates at 277.7 s. The result file is written, i.e. it is possible to plot. The plotting reveals that the simulation crash is probably due to the drum getting filled with water. The transformational debugger window points at the drum. The simulation output window recommends to log non-linear systems (NLS). Doing this gives a not responding OpenModelica. Restart gives a runtime error. A restart and simulation again without LOG_NLS activated gives the same result. The plotting of the Drum parameter mass shows that drum gets filled as it has no outlet (Figure 9).

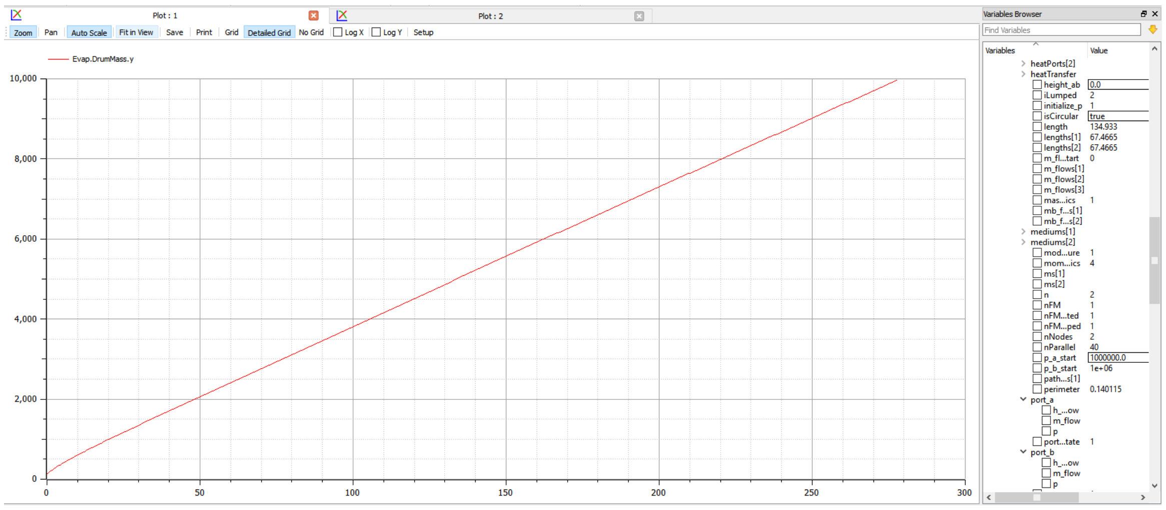
The debugger test failed here on an OpenModelica problem with handling LOG_NLS. However, at this error the result file was generated and provides useful information for debugging. On the other hand, LOG_NLS is not a part of the debugger. The debugger information for this type of failure is not sufficient to remedy the problem directly, although it points at the drum as a probable cause. Eventually the recommended logging of NLS could have given the direct cause of crash. From the OpenModelica user point of view, the plotting after crash is very valuable, and it reveals that the drum gets filled from the steam pipe model[1], which calls for corrective actions regardless of the what caused the actual solver crash.

5 Performance Analyzer Usage Evaluation

The performance analyzer (usually called profiler) analysis methods and implementation are described in more detail in (Sjölund, 2015, Chapter 5).

The OpenModelica profiler uses compiler-assisted source code instrumentation.

[1]LBA in Figure 1, named according to the Kraftwerk-Kennzeichen-System (KKS) identification system.



Variables	Value
heatPorts[2]	
heatTransfer	
height_ab	0.0
iLumped	2
initialize_p	1
isCircular	true
length	134.933
lengths[1]	67.4665
lengths[2]	67.4665
m_fl...tart	0
m_flows[1]	
m_flows[2]	
m_flows[3]	
mas...ics	1
mb_f...s[1]	
mb_f...s[2]	
mediums[1]	
mediums[2]	
mod...ure	1
mom...ics	4
ms[1]	
ms[2]	
n	2
nFM	1
nFM...ted	1
nFM...ped	1
nNodes	2
nParallel	40
p_a_start	1000000.0
p_b_start	1e+06
path...s[1]	
perimeter	0.140115
port_a	
h_...ow	
m_flow	
p	
port...tate	1
port_b	
h_...ow	
m_flow	
p	

Figure 9: Plot of drum mass at blocked drum outlet

There is one call to the clock before executing the equation block or function call and one call to the clock after execution of the block. Associated with each call is a counter that keeps track on how many times this function was triggered for the given time step. Similarly, each call is associated with clock data one variable for the total time spent in the block for all time steps, and one variable for the total time spent in the block for the current time step. These calls to the clock are in the code generation as a macro set that is generated if profiling is desired this means zero overhead unless profiling is explicitly enabled.

Profiling can be enabled for all equations and function calls. With profiling enabled only for equation blocks (SCCs) and functions, the overhead cost is low compared to the cost of solving most nonlinear systems of equations, which is more suitable for real-time simulation.

The instrumentation is performed by compiling a model with additional code generated to query real-time clocks at appropriate places.

The integrated performance profiler functionality is accessible from the GUI of the transformational debugger (Figure 10).

In the foreground is the window Transformational Debugger that invoked by the selection of « Profiler All » in the simulation setup options for simulation flags. Behind that the « Simulation Output » window, giving the important information that 100% of specified simulation time is successfully reached, but also various warnings in red with an option to debug more. (The output window is better always placed in forefront, to see the progress of the simulation, or alternatively the plot window if it plots the progress continuously or at least frequently). In the background the OpenModelica OMEdit main window is visible, now displaying the plotting interface.

A closer look at the Transformational Debugger and Profiling window (Figure 11) gives the following:

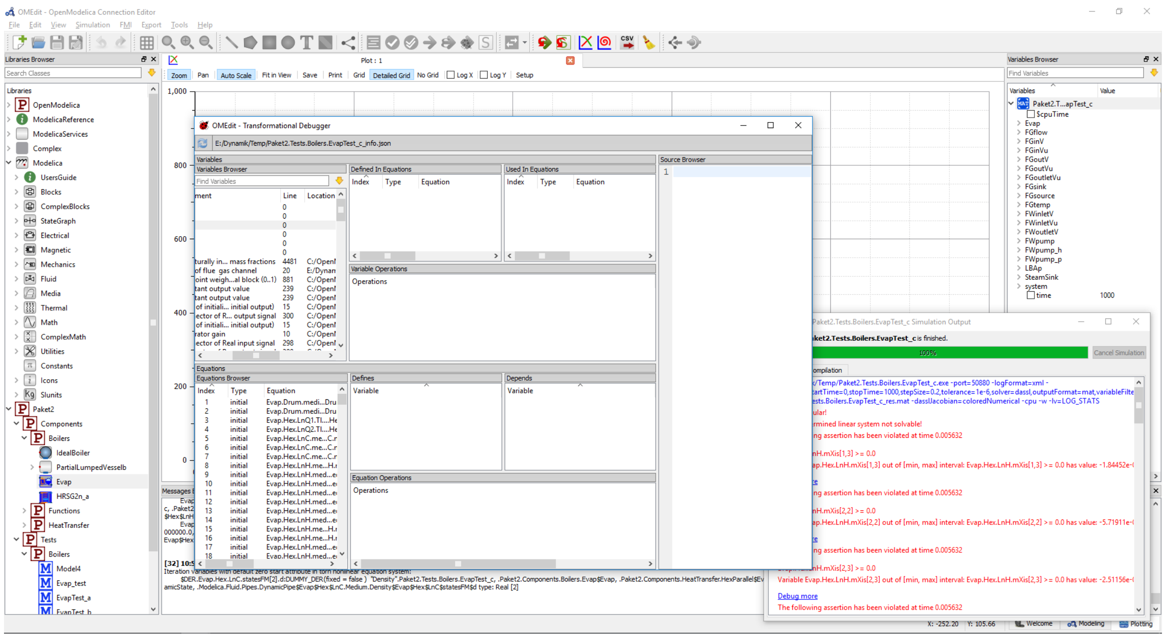
To the left the integrated transformational debugger and performance profiler displays two browser windows, one for variables and one for equations. Clicking on a variable in the variable browser window (middle left) will show the following:

- where it is defined
- where it is used
- operations made
- the source code line that defines the variable value will be highlighted (source browser to the right)

The contents of displayed lines are truncated to allow all windows above to fit into the same screen. Expanding/resizing windows and scrolling can display all hidden text. Thus, knowledge about where a variable is used becomes instantly available which substantially helps to analyze the consequences of a model change.

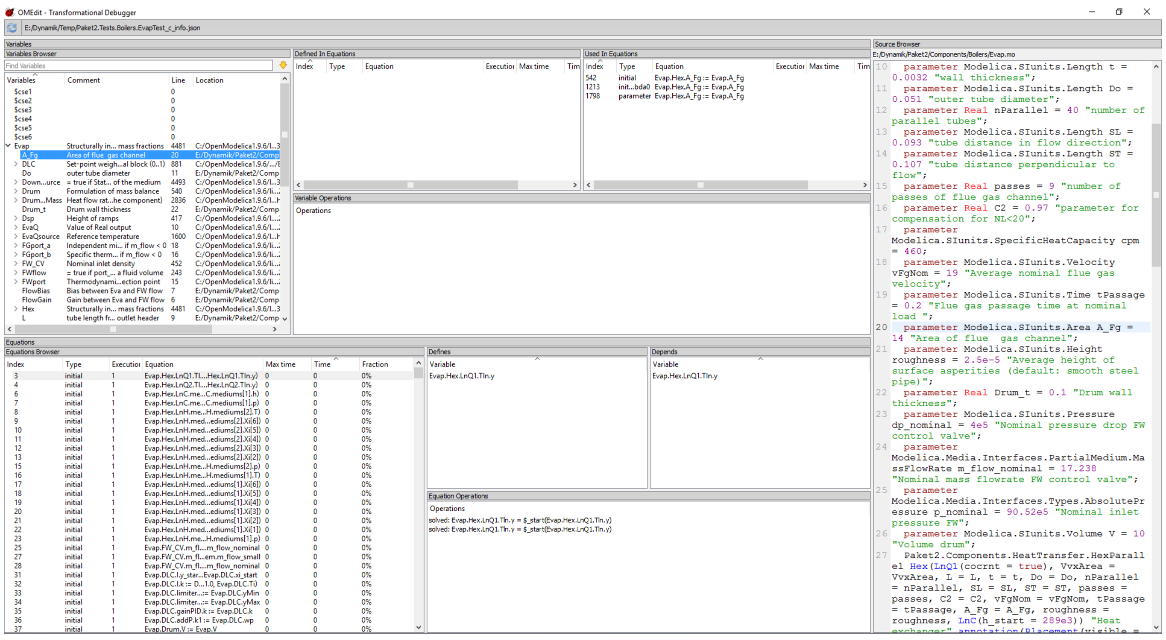
The index presented provides the link to the equations which are used in the model and presented in the equation browser. Clicking on a line in the equation browser window (lower left) gives:

- which variable the equation defines
- what variables it depends upon
- what operations that are made for the equation
- where in the source code the equation is placed



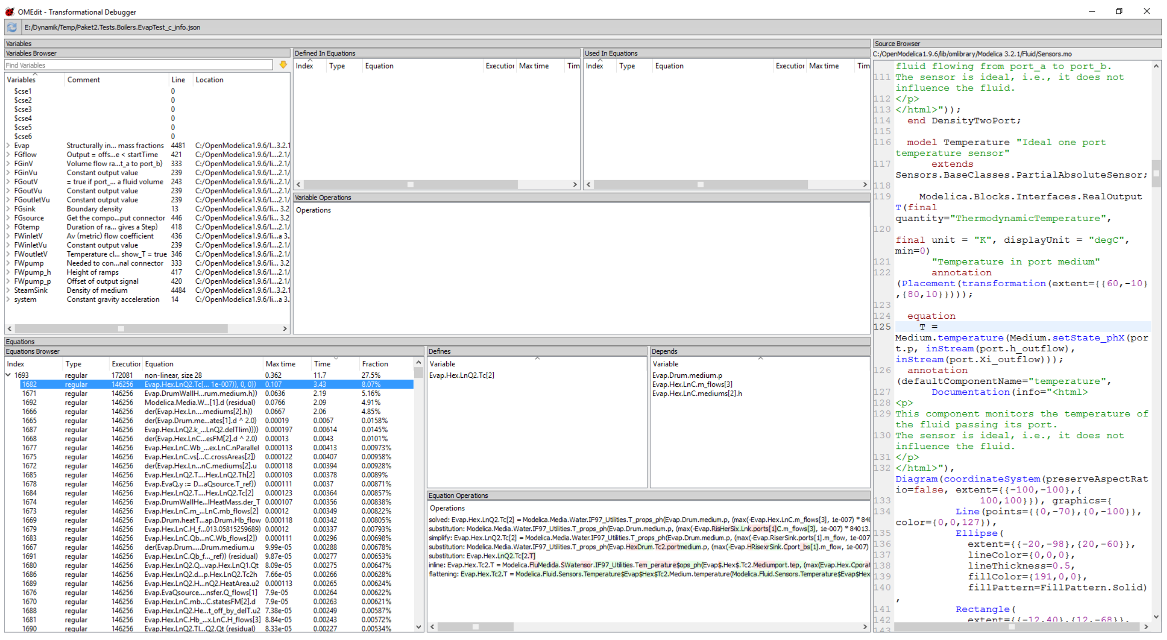
```
Paket2.Tests.Boilers.EvapTest_c Simulation Output
ket2.Tests.Boilers.EvapTest_c is finished.
100%
/Temp/Paket2.Tests.Boilers.EvapTest_c.exe -port=50880 -logFormat=xml -
tartTime=0,stopTime=1000,stepSize=0.2,tolerance=1e-6,solver=dassl,outputFormat=mat,variableFilte
ests.Boilers.EvapTest_c_res.mat -dasslJacobian=coloredNumerical -cpu -w -lv=LOG_STATS
ular!
rmined linear system not solvable!
ng assertion has been violated at time 0.005632
nH.mXis[1,3] >= 0.0
ap.Hex.LnH.mXis[1,3] out of [min, max] interval: Evap.Hex.LnH.mXis[1,3] >= 0.0 has value: -1.84452e-
ng assertion has been violated at time 0.005632
nH.mXis[2,2] >= 0.0
ap.Hex.LnH.mXis[2,2] out of [min, max] interval: Evap.Hex.LnH.mXis[2,2] >= 0.0 has value: -5.71911e-
ng assertion has been violated at time 0.005632
nH.mXis[2,3] >= 0.0
Variable Evap.Hex.LnH.mXis[2,3] out of [min, max] interval: Evap.Hex.LnH.mXis[2,3] >= 0.0 has value: -2.51156e-
Debug more
The following assertion has been violated at time 0.005632
```

Figure 10: Output on computer screen after a successful execution with profiler activated (option all selected)



```
Source Browser
E:/Dynamik/Paket2/Components/Boilers/Evap.mo
  parameter Modelica.SIunits.Length t = 0.0032 "wall thickness";
  parameter Modelica.SIunits.Length Do = 0.051 "outer tube diameter";
  parameter Real nParallel = 40 "number of parallel tubes";
  parameter Modelica.SIunits.Length SL = 0.093 "tube distance in flow direction";
  parameter Modelica.SIunits.Length ST = 0.107 "tube distance perpendicular to flow";
  parameter Real passes = 9 "number of passes of flue gas channel";
  parameter Real C2 = 0.97 "parameter for compensation for NL<20";
  parameter Modelica.SIunits.SpecificHeatCapacity cpm = 460;
  parameter Modelica.SIunits.Velocity vFgNom = 19 "Average nominal flue gas velocity";
  parameter Modelica.SIunits.Time tPassage = 0.2 "Flue gas passage time at nominal load ";
  parameter Modelica.SIunits.Area A_Fg = 14 "Area of flue  gas channel";
  parameter Modelica.SIunits.Height roughness = 2.5e-5 "Average height of surface asperities (default: smooth steel pipe)";
  parameter Real Drum_t = 0.1 "Drum wall thickness";
  parameter Modelica.SIunits.Pressure dp_nominal = 4e5 "Nominal pressure drop FW control valve";
  parameter Modelica.Media.Interfaces.PartialMedium.MassFlowRate m_flow_nominal = 17.238 "Nominal mass flowrate FW control valve";
  parameter Modelica.Media.Interfaces.Types.AbsolutePressure p_nominal = 90.52e5 "Nominal inlet pressure FW";
  parameter Modelica.SIunits.Volume V = 10 "Volume drum";
   Paket2.Components.HeatTransfer.HexParallel Hex(LnQ1(cocrnt = true), VvxArea = VvxArea, L = L, t = t, Do = Do, nParallel = nParallel, SL = SL, ST = ST, passes = passes, C2 = C2, vFgNom = vFgNom, tPassage = tPassage, A_Fg = A_Fg, roughness = roughness, LnC(h_start = 289e3)) "Heat
```

Figure 11: Transformational debugger and profiling window



Variables	Comment	Line	Location
$cse1		0	
$cse2		0	
$cse3		0	
$cse4		0	
$cse5		0	
$cse6		0	
Evap	Structurally in... mass fractions	4481	C:/OpenModelica1.9.6/l...3.2.1
FGflow	Output = offs...e < startTime	421	C:/OpenModelica1.9.6/l....2.1/
FGinV	Volume flow ra...t_a to port_b)	333	C:/OpenModelica1.9.6/li...2.1/
FGinVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FGoutV	= true if port_... a fluid volume	243	C:/OpenModelica1.9.6/li...2.1/
FGoutVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FGoutletVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FGsink	Boundary density	13	C:/OpenModelica1.9.6/li... 3.2
FGsource	Get the compo...put connector	446	C:/OpenModelica1.9.6/li... 3.2
FGtemp	Duration of ra... gives a Step)	418	C:/OpenModelica1.9.6/l....2.1/
FWinletV	Av (metric) flow coefficient	436	C:/OpenModelica1.9.6/li...a 3.
FWinletVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FWoutletV	Temperature cl... show_T = true	346	C:/OpenModelica1.9.6/li...2.1/
FWpump	Needed to con...nal connector	333	C:/OpenModelica1.9.6/li... 3.2
FWpump_h	Height of ramps	417	C:/OpenModelica1.9.6/l....2.1/
FWpump_p	Offset of output signal	420	C:/OpenModelica1.9.6/l....2.1/
SteamSink	Density of medium	4484	C:/OpenModelica1.9.6/l...3.2.1
system	Constant gravity acceleration	14	C:/OpenModelica1.9.6/li...a 3.

Index	Type	Executior	Equation	Max time	Time	Fraction
1693	regular	172081	non-linear, size 28	0.362	11.7	27.5%
1682	regular	146256	Evap.Hex.LnQ2.Tc[... 1e-007)), 0, 0))	0.107	3.43	8.07%
1671	regular	146256	Evap.DrumWallH...rum.medium.h))	0.0636	2.19	5.16%
1692	regular	146256	Modelica.Media.W...[1].d (residual)	0.0766	2.09	4.91%
1666	regular	146256	der(Evap.Hex.Ln....mediums[2].h))	0.0667	2.06	4.85%
1665	regular	146256	der(Evap.Drum.me...ates[1].d ^ 2.0)	0.00019	0.0067	0.0158%
1687	regular	146256	Evap.Hex.LnQ2.k_...LnQ2.delTlim))))	0.000197	0.00614	0.0145%
1668	regular	146256	der(Evap.Hex.LnC...esFM[2].d ^ 2.0)	0.00013	0.0043	0.0101%
1677	regular	146256	Evap.Hex.LnC.Wb_...ex.LnC.nParallel	0.000113	0.00413	0.00973%
1675	regular	146256	Evap.Hex.LnC.vs[...C.crossAreas[2])	0.000122	0.00407	0.00958%
1672	regular	146256	der(Evap.Hex.Ln...nC.mediums[2].u	0.000118	0.00394	0.00928%
1685	regular	146256	Evap.Hex.LnQ2.T....Hex.LnQ2.Th[2]	0.000103	0.00378	0.0089%
1678	regular	146256	Evap.EvaQ.y := D...aQsource.T_ref))	0.000111	0.0037	0.00871%
1684	regular	146256	Evap.Hex.LnQ2.T....Hex.LnQ2.Tc[2]	0.000123	0.00364	0.00857%
1674	regular	146256	Evap.DrumWallHe...lHeatMass.der_T	0.000107	0.00356	0.00838%
1673	regular	146256	Evap.Hex.LnC.m_...LnC.mb_flows[2]	0.00012	0.00349	0.00822%
1669	regular	146256	Evap.Drum.heatT...ap.Drum.Hb_flow	0.000118	0.00342	0.00805%
1679	regular	146256	Evap.Hex.LnC.H_f...013.05815259689)	0.00012	0.00337	0.00793%
1683	regular	146256	Evap.Hex.LnC.Qb...nC.Wb_flows[2])	0.000111	0.00296	0.00698%
1667	regular	146256	der(Evap.Drum.....Drum.medium.u	9.99e-05	0.00288	0.00678%
1691	regular	146256	Evap.Hex.LnC.Qb_f..._ref)) (residual)	9.87e-05	0.00277	0.00653%
1680	regular	146256	Evap.Hex.LnQ2.Q...vap.Hex.LnQ1.Qt	8.09e-05	0.00275	0.00647%
1686	regular	146256	Evap.Hex.LnQ2.d...p.Hex.LnQ2.Tc2h	7.66e-05	0.00266	0.00628%
1689	regular	146256	Evap.Hex.LnQ2.H...nQ2.HeatArea.u2	0.000113	0.00265	0.00624%
1676	regular	146256	Evap.EvaQsource....nsfer.Q_flows[1]	7.9e-05	0.00264	0.00622%
1670	regular	146256	Evap.Hex.LnC.mb...C.statesFM[2].d	7.9e-05	0.00263	0.00621%
1688	regular	146256	Evap.Hex.LnQ2.He...t_off_by_delT.u2	7.38e-05	0.00249	0.00587%
1681	regular	146256	Evap.Hex.LnC.Hb_...x.LnC.H_flows[3]	8.84e-05	0.00243	0.00572%
1690	regular	146256	Evap.Hex.LnQ2.Tl...Q2.Qt (residual)	8.33e-05	0.00227	0.00534%

Defines: Variable
Evap.Hex.LnQ2.Tc[2]

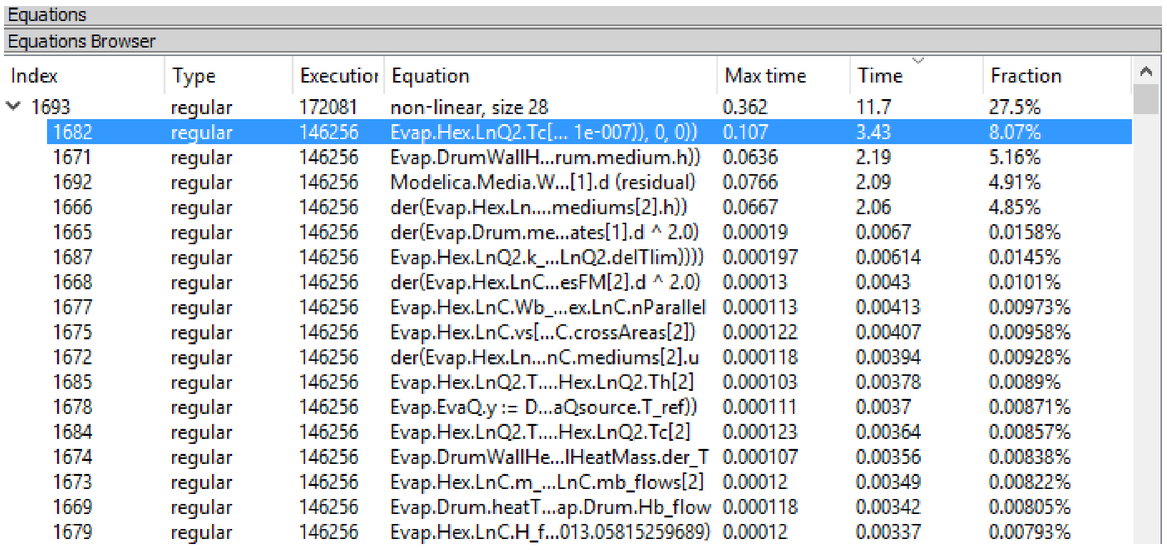
Depends: Variable
Evap.Drum.medium.p
Evap.Hex.LnC.m_flows[3]
Evap.Hex.LnC.mediums[2].h

```
Operations
solved: Evap.Hex.LnQ2.Tc[2] = Modelica.Media.Water.IF97_Utilities.T_props_ph(Evap.Drum.medium.p, (max(-Evap.Hex.LnC.m_flows[3], 1e-007) * 84
substitution: Modelica.Media.Water.IF97_Utilities.T_props_ph(Evap.Drum.medium.p, (max(-Evap.RisHerSix.Lnk.ports[1]C.m_flows[3], 1e-007) * 84013.
simplify: Evap.Hex.LnQ2.Tc[2] = Modelica.Media.Water.IF97_Utilities.T_props_ph(Evap.Drum.medium.p, (max(-Evap.RiserSink.ports[1].m_flow, 1e-007
substitution: Modelica.Media.Water.IF97_Utilities.T_props_ph(Evap.HexDrum.Tc2.portmedium.p, (max(-Evap.HRisexrSink.Cport_bs[1].m_flow, 1e-007)
substitution: Evap.Hex.LnQ2.Tc[2.T]
inline: Evap.Hex.Tc2.T = Modelica.FluMedida.SWatensor.IF97_Utilities.Tem_perature$ops_ph(Evap$.Hex$.Tc2.Mediumport.tep, (max(Evap.Hex.Cporat
flattening: Evap.Hex.Tc2.T = Modelica.Fluid.Sensors.Temperature$Evap$Hex$Tc2.Medium.temperature(Modelica.Fluid.Sensors.Temperature$Evap$Hex
```

C:/OpenModelica1.9.6/lib/omlibrary/Modelica 3.2.1/Fluid/Sensors.mo

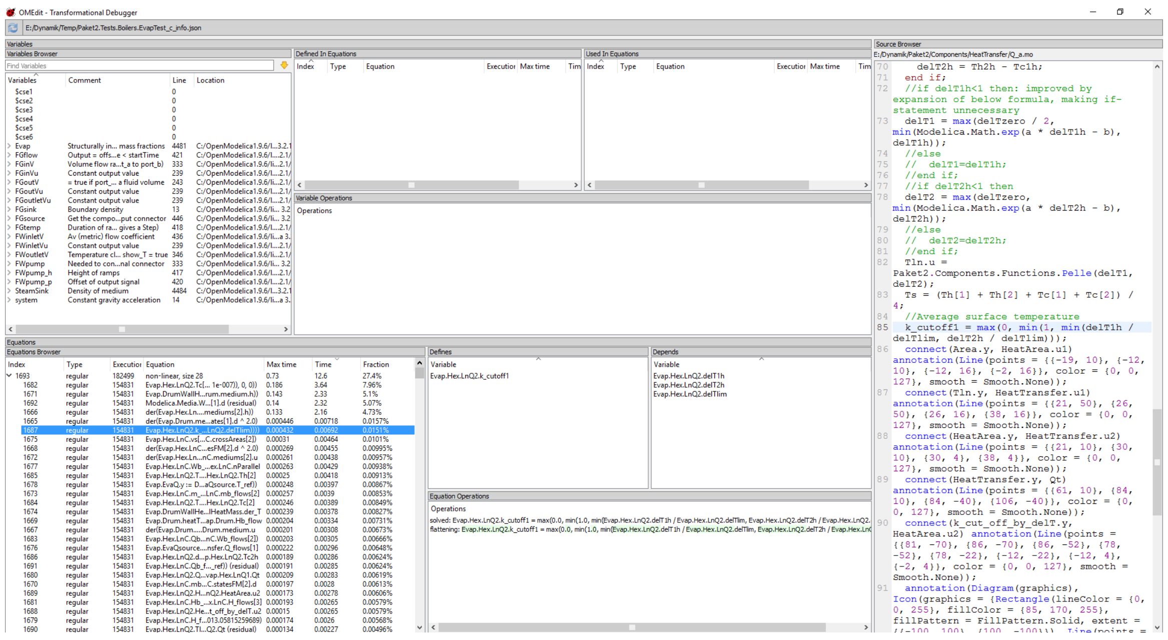
```
fluid flowing from port_a to port_b.
The sensor is ideal, i.e., it does not
influence the fluid.
</p>
</html>"));
  end DensityTwoPort;

  model Temperature "Ideal one port
temperature sensor"
      extends
Sensors.BaseClasses.PartialAbsoluteSensor;

    Modelica.Blocks.Interfaces.RealOutput
T(final
quantity="ThermodynamicTemperature",

final unit = "K", displayUnit = "degC",
min=0)
      "Temperature in port medium"
      annotation
(Placement(transformation(extent={{60,-10}
,{80,10}})));

  equation
    T =
Medium.temperature(Medium.setState_phX(por
t.p, inStream(port.h_outflow),
inStream(port.Xi_outflow)));
  annotation
(defaultComponentName="temperature",
      Documentation(info="<html>
<p>
This component monitors the temperature of
the fluid passing its port.
The sensor is ideal, i.e., it does not
influence the fluid.
</p>
</html>"),
Diagram(coordinateSystem(preserveAspectRat
io=false, extent={{-100,-100},{
              100,100}}), graphics={
          Line(points={{0,-70},{0,-100}},
color={0,0,127}),
          Ellipse(
            extent={{-20,-98},{20,-60}},
            lineColor={0,0,0},
            lineThickness=0.5,
            fillColor={191,0,0},
            fillPattern=FillPattern.Solid)
,
          Rectangle(
```

Figure 12: Using the performance profiler to find the computationally heaviest equation after sorting according to used time (table lower left), see close-up in Figure 13.



Index	Type	Executior	Equation	Max time	Time	Fraction
1693	regular	172081	non-linear, size 28	0.362	11.7	27.5%
1682	regular	146256	Evap.Hex.LnQ2.Tc[... 1e-007)), 0, 0))	0.107	3.43	8.07%
1671	regular	146256	Evap.DrumWallH...rum.medium.h))	0.0636	2.19	5.16%
1692	regular	146256	Modelica.Media.W...[1].d (residual)	0.0766	2.09	4.91%
1666	regular	146256	der(Evap.Hex.Ln....mediums[2].h))	0.0667	2.06	4.85%
1665	regular	146256	der(Evap.Drum.me...ates[1].d ^ 2.0)	0.00019	0.0067	0.0158%
1687	regular	146256	Evap.Hex.LnQ2.k_...LnQ2.delTlim))))	0.000197	0.00614	0.0145%
1668	regular	146256	der(Evap.Hex.LnC...esFM[2].d ^ 2.0)	0.00013	0.0043	0.0101%
1677	regular	146256	Evap.Hex.LnC.Wb_...ex.LnC.nParallel	0.000113	0.00413	0.00973%
1675	regular	146256	Evap.Hex.LnC.vs[...C.crossAreas[2])	0.000122	0.00407	0.00958%
1672	regular	146256	der(Evap.Hex.Ln...nC.mediums[2].u	0.000118	0.00394	0.00928%
1685	regular	146256	Evap.Hex.LnQ2.T....Hex.LnQ2.Th[2]	0.000103	0.00378	0.0089%
1678	regular	146256	Evap.EvaQ.y := D...aQsource.T_ref))	0.000111	0.0037	0.00871%
1684	regular	146256	Evap.Hex.LnQ2.T....Hex.LnQ2.Tc[2]	0.000123	0.00364	0.00857%
1674	regular	146256	Evap.DrumWallHe...lHeatMass.der_T	0.000107	0.00356	0.00838%
1673	regular	146256	Evap.Hex.LnC.m_...LnC.mb_flows[2]	0.00012	0.00349	0.00822%
1669	regular	146256	Evap.Drum.heatT...ap.Drum.Hb_flow	0.000118	0.00342	0.00805%
1679	regular	146256	Evap.Hex.LnC.H_f...013.05815259689)	0.00012	0.00337	0.00793%

Figure 13: Transformational debugger and profiling window



E:/Dynamik/Temp/Paket2.Tests.Boilers.EvapTest_c_info.json

Variables	Comment	Line	Location
$cse1		0	
$cse2		0	
$cse3		0	
$cse4		0	
$cse5		0	
$cse6		0	
Evap	Structurally in... mass fractions	4481	C:/OpenModelica1.9.6/l...3.2.1
FGflow	Output = offs...e < startTime	421	C:/OpenModelica1.9.6/l....2.1/
FGinV	Volume flow ra...t_a to port_b)	333	C:/OpenModelica1.9.6/li...2.1/
FGinVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FGoutV	= true if port_... a fluid volume	243	C:/OpenModelica1.9.6/li...2.1/
FGoutVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FGoutletVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FGsink	Boundary density	13	C:/OpenModelica1.9.6/li... 3.2
FGsource	Get the compo...put connector	446	C:/OpenModelica1.9.6/li... 3.2
FGtemp	Duration of ra... gives a Step)	418	C:/OpenModelica1.9.6/l....2.1/
FWinletV	Av (metric) flow coefficient	436	C:/OpenModelica1.9.6/li...a 3.
FWinletVu	Constant output value	239	C:/OpenModelica1.9.6/l....2.1/
FWoutletV	Temperature cl... show_T = true	346	C:/OpenModelica1.9.6/li...2.1/
FWpump	Needed to con...nal connector	333	C:/OpenModelica1.9.6/li... 3.2
FWpump_h	Height of ramps	417	C:/OpenModelica1.9.6/l....2.1/
FWpump_p	Offset of output signal	420	C:/OpenModelica1.9.6/l....2.1/
SteamSink	Density of medium	4484	C:/OpenModelica1.9.6/l...3.2.1
system	Constant gravity acceleration	14	C:/OpenModelica1.9.6/li...a 3.

Index	Type	Executior	Equation	Max time	Time	Fraction
1693	regular	182499	non-linear, size 28	0.73	12.6	27.4%
1682	regular	154831	Evap.Hex.LnQ2.Tc[... 1e-007)), 0, 0))	0.186	3.64	7.96%
1671	regular	154831	Evap.DrumWallH...rum.medium.h))	0.143	2.33	5.1%
1692	regular	154831	Modelica.Media.W...[1].d (residual)	0.14	2.32	5.07%
1666	regular	154831	der(Evap.Hex.Ln....mediums[2].h))	0.133	2.16	4.73%
1665	regular	154831	der(Evap.Drum.me...ates[1].d ^ 2.0)	0.000446	0.00718	0.0157%
1687	regular	154831	Evap.Hex.LnQ2.k_...LnQ2.delTlim))))	0.000432	0.00692	0.0151%
1675	regular	154831	Evap.Hex.LnC.vs[...C.crossAreas[2])	0.00031	0.00464	0.0101%
1668	regular	154831	der(Evap.Hex.LnC...esFM[2].d ^ 2.0)	0.000269	0.00455	0.00995%
1672	regular	154831	der(Evap.Hex.Ln...nC.mediums[2].u	0.000261	0.00438	0.00957%
1677	regular	154831	Evap.Hex.LnC.Wb_...ex.LnC.nParallel	0.000263	0.00429	0.00938%
1685	regular	154831	Evap.Hex.LnQ2.T....Hex.LnQ2.Th[2]	0.00025	0.00418	0.00913%
1678	regular	154831	Evap.EvaQ.y := D...aQsource.T_ref))	0.000248	0.00397	0.00867%
1673	regular	154831	Evap.Hex.LnC.m_...LnC.mb_flows[2]	0.000257	0.0039	0.00853%
1684	regular	154831	Evap.Hex.LnQ2.T....Hex.LnQ2.Tc[2]	0.000246	0.00389	0.00849%
1674	regular	154831	Evap.DrumWallHe...lHeatMass.der_T	0.000239	0.00378	0.00827%
1669	regular	154831	Evap.Drum.heatT...ap.Drum.Hb_flow	0.000204	0.00334	0.00731%
1667	regular	154831	der(Evap.Drum.....Drum.medium.u	0.000201	0.00308	0.00673%
1683	regular	154831	Evap.Hex.LnC.Qb...nC.Wb_flows[2])	0.000203	0.00305	0.00666%
1676	regular	154831	Evap.EvaQsource....nsfer.Q_flows[1]	0.000222	0.00296	0.00648%
1686	regular	154831	Evap.Hex.LnQ2.d...p.Hex.LnQ2.Tc2h	0.000189	0.00286	0.00624%
1691	regular	154831	Evap.Hex.LnC.Qb_f..._ref)) (residual)	0.000191	0.00285	0.00624%
1680	regular	154831	Evap.Hex.LnQ2.Q...vap.Hex.LnQ1.Qt	0.000209	0.00283	0.00619%
1670	regular	154831	Evap.Hex.LnC.mb...C.statesFM[2].d	0.000197	0.0028	0.00613%
1689	regular	154831	Evap.Hex.LnQ2.H...nQ2.HeatArea.u2	0.000173	0.00278	0.00606%
1681	regular	154831	Evap.Hex.LnC.Hb_...x.LnC.H_flows[3]	0.000193	0.00265	0.00579%
1688	regular	154831	Evap.Hex.LnQ2.He...t_off_by_delT.u2	0.00015	0.00265	0.00579%
1679	regular	154831	Evap.Hex.LnC.H_f...013.05815259689)	0.000174	0.0026	0.00568%
1690	regular	154831	Evap.Hex.LnQ2.Tl...Q2.Qt (residual)	0.000134	0.00227	0.00496%

Defines: Variable
Evap.Hex.LnQ2.k_cutoff1

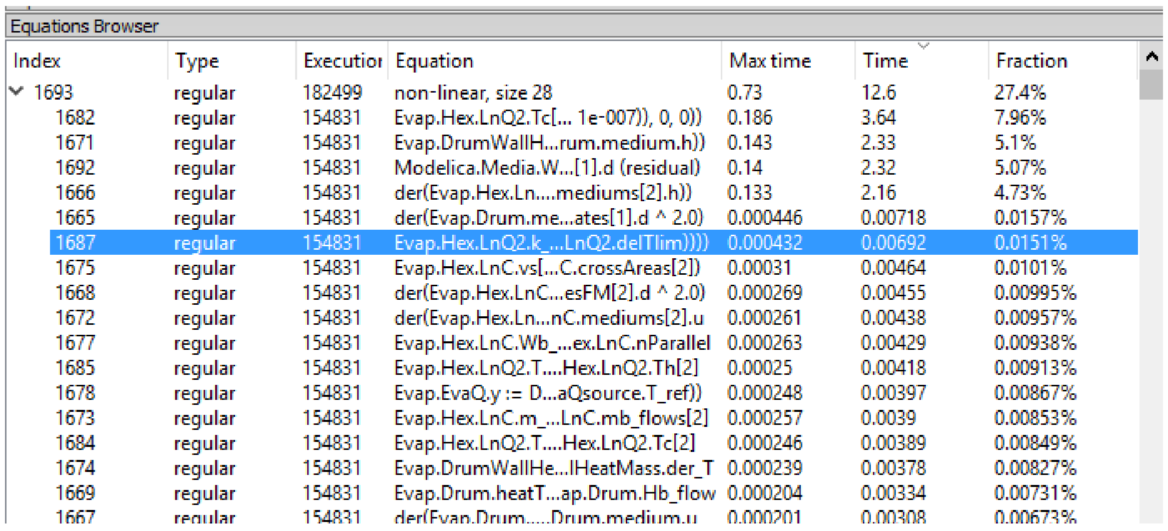
Depends: Variable
Evap.Hex.LnQ2.delT1h
Evap.Hex.LnQ2.delT2h
Evap.Hex.LnQ2.delTlim

```
solved: Evap.Hex.LnQ2.k_cutoff1 = max(0.0, min(1.0, min(Evap.Hex.LnQ2.delT1h / Evap.Hex.LnQ2.delTlim, Evap.Hex.LnQ2.delT2h / Evap.Hex.LnQ2.
flattening: Evap.Hex.LnQ2.k_cutoff1 = max(0.0, min(1.0, min(Evap.Hex.LnQ2.delT1h / Evap.Hex.LnQ2.delTlim, Evap.Hex.LnQ2.delT2h / Evap.Hex.LnQ
```

E:/Dynamik/Paket2/Components/HeatTransfer/Q_a.mo

```
    delT2h = Th2h - Tc1h;
  end if;
  //if delT1h<1 then: improved by expansion of below formula, making if-statement unnecessary
  delT1 = max(delTzero / 2, min(Modelica.Math.exp(a * delT1h - b), delT1h));
  //else
  //  delT1=delT1h;
  //end if;
  //if delT2h<1 then
  delT2 = max(delTzero, min(Modelica.Math.exp(a * delT2h - b), delT2h));
  //else
  //  delT2=delT2h;
  //end if;
  Tln.u = Paket2.Components.Functions.Pelle(delT1, delT2);
  Ts = (Th[1] + Th[2] + Tc[1] + Tc[2]) / 4;
  //Average surface temperature
  k_cutoff1 = max(0, min(1, min(delT1h / delTlim, delT2h / delTlim)));
  connect(Area.y, HeatArea.u1) annotation(Line(points = {{-19, 10}, {-12, 10}, {-12, 16}, {-2, 16}}, color = {0, 0, 127}, smooth = Smooth.None));
  connect(Tln.y, HeatTransfer.u1) annotation(Line(points = {{21, 50}, {26, 50}, {26, 16}, {38, 16}}, color = {0, 0, 127}, smooth = Smooth.None));
  connect(HeatArea.y, HeatTransfer.u2) annotation(Line(points = {{21, 10}, {30, 10}, {30, 4}, {38, 4}}, color = {0, 0, 127}, smooth = Smooth.None));
  connect(HeatTransfer.y, Qt) annotation(Line(points = {{61, 10}, {84, 10}, {84, -40}, {106, -40}}, color = {0, 0, 127}, smooth = Smooth.None));
  connect(k_cut_off_by_delT.y, HeatArea.u2) annotation(Line(points = {{81, -70}, {86, -70}, {86, -52}, {78, -52}, {78, -22}, {-12, -22}, {-12, 4}, {-2, 4}}, color = {0, 0, 127}, smooth = Smooth.None));
  annotation(Diagram(graphics), Icon(graphics = {Rectangle(lineColor = {0, 0, 255}, fillColor = {85, 170, 255}, fillPattern = FillPattern.Solid, extent =
```

Figure 14: Performance profiler output after a minor model adjustment. See also close-up in the Figure 15.



Index	Type	Executior	Equation	Max time	Time	Fraction
1693	regular	182499	non-linear, size 28	0.73	12.6	27.4%
1682	regular	154831	Evap.Hex.LnQ2.Tc[... 1e-007)), 0, 0))	0.186	3.64	7.96%
1671	regular	154831	Evap.DrumWallH...rum.medium.h))	0.143	2.33	5.1%
1692	regular	154831	Modelica.Media.W...[1].d (residual)	0.14	2.32	5.07%
1666	regular	154831	der(Evap.Hex.Ln....mediums[2].h))	0.133	2.16	4.73%
1665	regular	154831	der(Evap.Drum.me...ates[1].d ^ 2.0)	0.000446	0.00718	0.0157%
1687	regular	154831	Evap.Hex.LnQ2.k_...LnQ2.delTlim))))	0.000432	0.00692	0.0151%
1675	regular	154831	Evap.Hex.LnC.vs[...C.crossAreas[2])	0.00031	0.00464	0.0101%
1668	regular	154831	der(Evap.Hex.LnC...esFM[2].d ^ 2.0)	0.000269	0.00455	0.00995%
1672	regular	154831	der(Evap.Hex.Ln...nC.mediums[2].u	0.000261	0.00438	0.00957%
1677	regular	154831	Evap.Hex.LnC.Wb_...ex.LnC.nParallel	0.000263	0.00429	0.00938%
1685	regular	154831	Evap.Hex.LnQ2.T....Hex.LnQ2.Th[2]	0.00025	0.00418	0.00913%
1678	regular	154831	Evap.EvaQ.y := D...aQsource.T_ref))	0.000248	0.00397	0.00867%
1673	regular	154831	Evap.Hex.LnC.m_...LnC.mb_flows[2]	0.000257	0.0039	0.00853%
1684	regular	154831	Evap.Hex.LnQ2.T....Hex.LnQ2.Tc[2]	0.000246	0.00389	0.00849%
1674	regular	154831	Evap.DrumWallHe...lHeatMass.der_T	0.000239	0.00378	0.00827%
1669	regular	154831	Evap.Drum.heatT...ap.Drum.Hb_flow	0.000204	0.00334	0.00731%
1667	regular	154831	der(Evap.Drum.....Drum.medium.u	0.000201	0.00308	0.00673%

Figure 15: Close-up of performance profiler output after a minor model adjustment. The 28 equations at Index 1693 use 27.4% of the time.

The equation browser gives the link between the C-code equation solved and the source code in the Modelica source files. This browser also displays columns for execution times, number of executions and, total execution time, both as fractions of total simulation time and in seconds. Sorting by the fraction of execution time used with the largest value at the top of the table presents the equations in order according to the time consumed.

As shown in Figure 12 and Figure 13 the profiler displays that Index 1693, a strongly connected equation subsystem that defines the derivative of the enthalpy and pressure of the drum together with the derivative of the enthalpy of the outlet volume of the evaporator pipes (`LnC[2]`), is by far the equation that takes most of the CPU power, 27.5 % while next equation takes 5 % (not shown in figure above). As there are three variables defined by this equation no equation is pointed out in the Modelica file (shown in Source Browser is a previous clicked index—a clearing of that window should be considered when selected variables and equations not corresponds to a line in the source code) nor are any dependencies shown.

Index 1693 constitutes 28 other variables (size 28) and clicking on 1693 reveals indices from 1665 to 1692 (28 equations). Those are the used equations by the solver and clicking on any of them shows the corresponding source code, and what variables the equation is dependent upon and the operations made.

The conclusion from this is that the Drum is the computationally heaviest part of the model, and improvements of its equations should have the best chance of improving performance and reducing the simulation time.

Exploring the 28 equations (Figure 13) reveals only one that is provided by the user (amongst the equations using 85 % of the calculation time). The rest are equations from the Modelica Fluid library. To see the impact by that equation on the performance, one of its parameters was changed. The equation has a parameter `delTlim` that is limiting the heat transfer calculation preventing that the temperature differences become equal, thereby causing a simulation crash. Increasing `delTlim` value from 0.1 to 0.5 ℃, which deteriorates the accuracy of the heat transfer calculation, gives the result that equations with index 1693 makes a minor reduction from 27.5 % to 27.4 % of the total time (Figure 14 and Figure 15), but the simulation time (major part of the total time) is anyway reduced from 30.0 to 28.5 s (not shown).

From that change the user could conclude that changing `delTlim`, which rather drastically deteriorates the accuracy of the heat transfer, the resulting improvement on the performance for index 1693 is negligible, but the change any way influences the total time a bit more substantially.

In general our experience is that the performance analyzer/profiler is a very important tool for model performance optimization since it is very easy to see the link between a slow execution and the equations used. This information is very helpful when changing the model in order to speed up its simulation.

6 Conclusions

A basic property of the debugger is to assist in case of numerical problems and violation of assertions like numerical ranges that a certain variable is expected never to exceed. Then the debugger points out the equation causing the problem. In the work reported in this paper only small to medium-sized (1000-equation size) industrial models have been tested, which demonstrates that the debugger and performance analyzer work well and give significant assistance to the user. The debugger has also been briefly evaluated by Pop et al. (2014) on 11116-equation size models in the Modelica Standard Library such as V6Engine. To really evaluate the benefits of the debugger, but also its functionality, it should also be applied to larger industrial models.

The following conclusions were made from the tests with the transformational debugger and performance analyzer for equation models:

- The debugger works well to find zero denominators that are parameters.

- The debugger does not come into play automatically if the zero denominator is only a momentarily value, as the solver managed work around such time points in so far tested simulations. However, it catches the problem in the simulation output window, and gives a message that by clicking opens the transformational debugger window which displays the concerned equation. However, there is a risk that this is unnoticed as the solver continues and could generate a lot of consequential or other messages that could hide the zero denominator messages. It would be preferable if the simulation output window could aggregate messages of the same type into one, expandable, line, thereby giving a better overview of all the types of messages the simulation has generated.

- A zero denominator caused by structural model errors, like connection to not used connectors (this should not pass the model checking) the debugger points to the causing equation. One could not ask more of the transformational debugger, but the OpenModelica model check or model building could be made to prevent such mistakes.

- In case of numerical problems causing long execution times the debugger points to the equations that have problems, but to understand the exact problem, plots of variables could be necessary—hence the result file should always be generated, regardless if the simulation is interrupted by solver or manually. This is not the case in the tested version for all the tests.

- The performance analyzer/profiler is a very important tool for model optimization as it is possible to easily see the link between a slow execution and the equations used. Compared to the alternative method where only the CPU curve together with plots of all other variables are available for guidance in the process of finding a good spot in the model to improve, the profiler makes the process of performance optimization of models radically shorter.

Acknowledgments

This work was partially supported by the ITEA2 MODRIO project and the ITEA3 OPENCPS project via the Swedish Government (Vinnova), by the RTISIM project funded by Vinnova, and by Siemens Turbo Machinery AB.

References

Bunus, P. *Debugging techniques for Equation-Based languages*. Doctoral thesis No 873, Linköping University, Department of Computer and Information Science, 2004. URL `http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-35555`.

Fritzson, P. *Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach*. Wiley-IEEE Press, 2 edition, 2015.

Graham, S., Kessler, P., and McKusick, M. An execution profiler for modular programs. *Software: Practice and Experience*, 1983. 13(8):671–685. doi:10.1002/spe.4380130803.

Huhn, M., Sjölund, M., Chen, W., Schulze, C., and Fritzson, P. Tool support for Modelica real-time models. In C. Clauß, editor, *Proceedings of the 8th International Modelica Conference*. Linköping University Electronic Press, 2011. doi:10.3384/ecp11063537.

Kinnander, Å., Sjölund, M., and Pop, A. Industrial evaluation of an efficient equation model debugger in OpenModelica. In *Proceedings of 9th EUROSIM Congress on Modelling and Simulation*. 2016.

Modelica Association. Modelica: A unified object-oriented language for physical systems modeling, language specification version 3.3 revision 1. 2014. URL `http://www.modelica.org/`.

Nethercote, N. and Seward, J. Valgrind: a framework for heavyweight dynamic binary instrumentation. In *Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation*, PLDI '07. pages 89–100, 2007. doi:10.1145/1250734.1250746.

Open Source Modelica Consortium. Openmodelica. 2016. URL `https://openmodelica.org/`.

Pop, A. and Fritzson, P. A portable debugger for algorithmic Modelica code. In G. Schmitz, editor, *Proceedings of the 4th International Modelica Conference*. 2005.

Pop, A., Sjölund, M., Ashgar, A., Fritzson, P., and Casella, F. Integrated Debugging of Modelica Models. *Modeling, Identification and Control*, 2014. 35(2):93–107. doi:10.4173/mic.2014.2.3.

Schulze, C., Huhn, M., and Schüler, M. Profiling of Modelica real-time models. In P. Fritzson, E. Lee, F. Cellier, and D. Broman, editors, *Proceedings of the 3rd International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*. Linköping University Electronic Press, pages 23–32, 2010. URL `http://www.ep.liu.se/ecp/047/`.

Sjölund, M. *Tools and Methods for Analysis, Debugging, and Performance Improvement of Equation-Based Models*. Doctoral thesis No 1664, Linköping University, Department of Computer and Information Science, 2015. doi:10.3384/diss.diva-116346.

Sjölund, M., Fritzson, P., and Pop, A. Bootstrapping a Compiler for an Equation-Based Object-Oriented Language. *Modeling, Identification and Control*, 2014. 35(1):1–19. doi:10.4173/mic.2014.1.1.

Stallman, R., Pesch, R., Shebs, S., et al. *Debugging with GDB*. Free Software Foundation, 2014. URL `http://www.gnu.org/software/gdb/documentation/`.

Zeller, A. *Why Programs Fail: A Guide to Systematic Debugging*. Morgan Kaufmann Publishers Inc., 2nd edition, 2009.