Generic simulation and graphical display of arbitrary Lab-on-the-Chip platforms

Axel Emmer[1*], Reinhold Schäfer[1], Friedhelm Schönfeld[2]

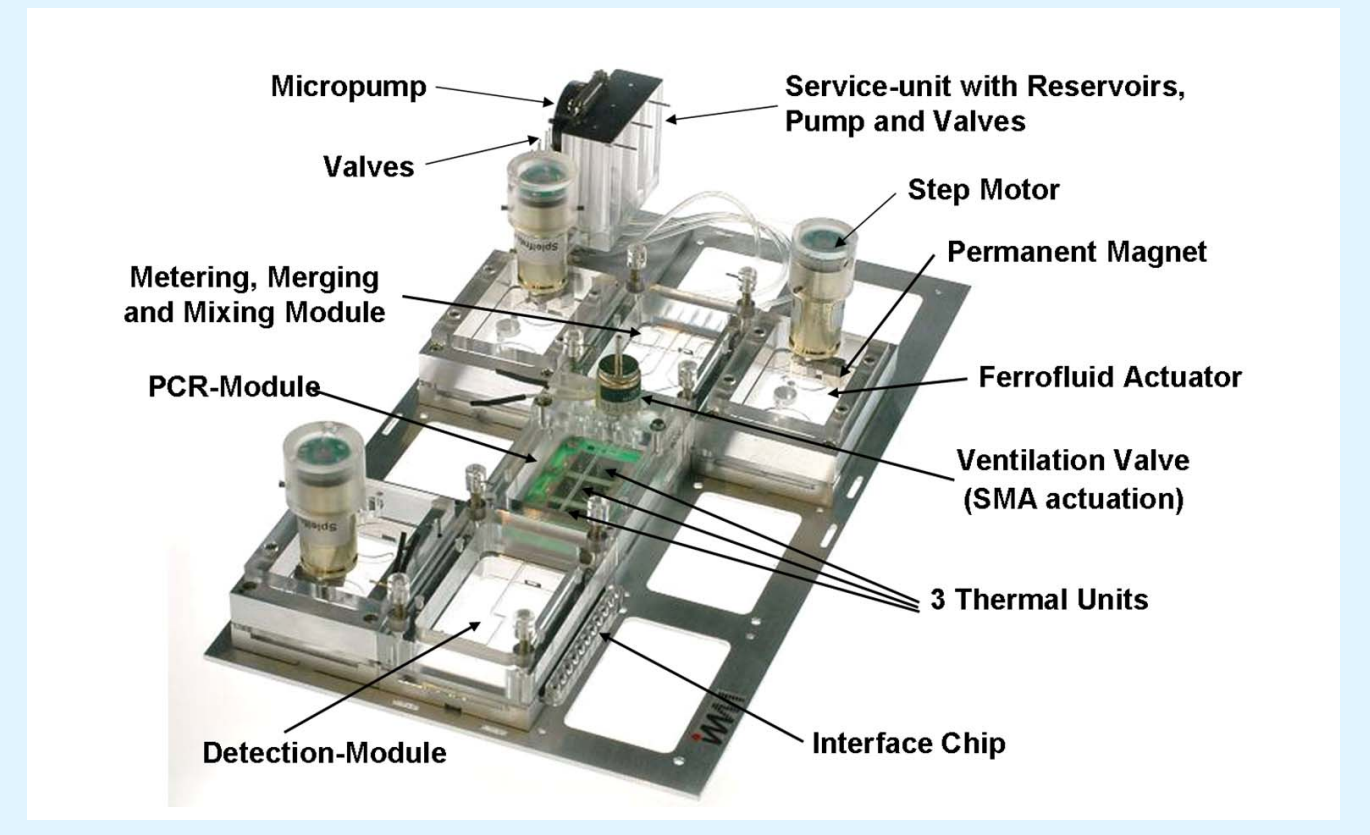
[1]Fachhochschule Wiesbaden - University of Applied Sciences (FHW), Kurt-Schumacher-Ring 18, 65197 Wiesbaden, Germany

[2]Institut für Mikrotechnik Mainz GmbH (IMM), Carl-Zeis-Strasse 18-20, 55129 Mainz, Germany

[*]Phone: +49 - 611 - 2367589, Fax: +49 - 611 - 2405334, Email: axel@emmernet.de

Goal

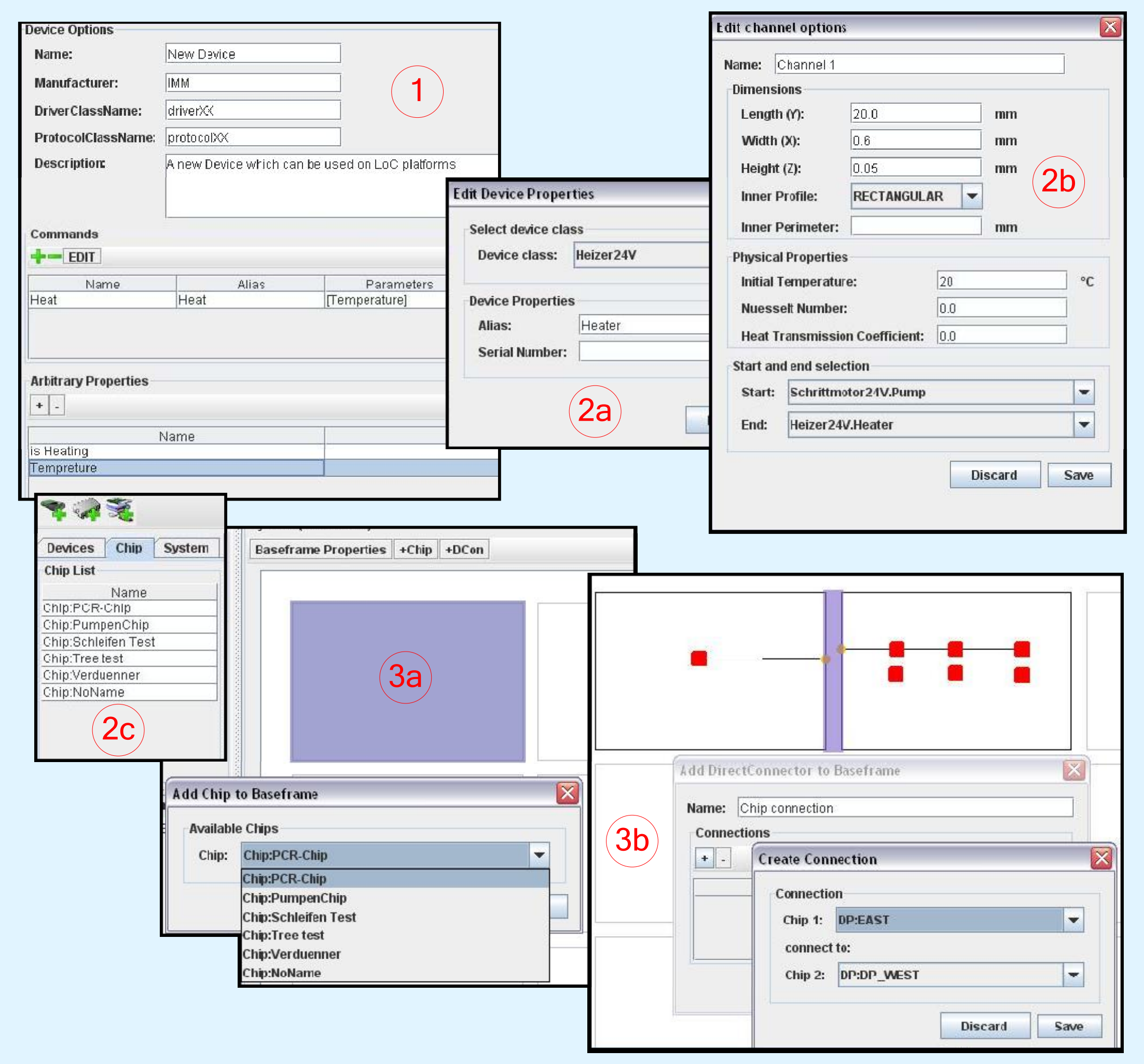
- Develop a software package to support construction and simulation of arbitrary Lab-on-the-Chip (LoC) platforms.
- Reduce development time for new LoC setups from days to hours.
- Generate a flexible library for channels, devices, sensors and chips as components for the system.
- Use a modeling software for fluid dynamic simulation.
- Display samples on the chips graphically.



Method

- Extend an existing software package controlling an arbitrary Lab-on-the-Chip platform with a generic workflow.
- Implement a two phase system
  - ➢ Definition phase:
    - ▪ Special editors for channels, devices, sensors, chips, and platforms.
    - ▪ Variety of attribute settings for all objects.
    - ▪ Modeling of fluidic chip characteristics with Modelica, an open source modeling software, providing a powerful algebraic mathematical framework. (http://www.modelica.org)
    - ▪ User-defined libraries for all objects.
  - ➢ Run time phase:
    - ▪ Automated generation of Modelica classes using the relevant entities defined in phase one editors.
    - ▪ Simulation of the physical model in Modelica.
    - ▪ Preparation of results for display.
    - ▪ Graphical 2D animation of "working" platform
- Increase efficiency utilizing the continually growing database of prototype elements.
- Use Java as an operating system independent language and mySQL as data storage tool.

Setting up a Platform



1. Create devices and sensors to be used on platforms. Each device may consist of arbitrary commands and properties, defined in editors. (Editor 1: Device and sensor prototypes)

2a. Place devices on a chip in drag and drop mode. These instances can be named individually.

2b. Connect devices with channels and set channel attributes, e.g. length, height, and width. (Editor 2)

2c. Store completed chips in database as chip prototypes for the LoC platform. (Editor 2: Chip prototype)

3a. Place chip instances on a platform of any desired layout.

3c. Couple placed chips via dedicated connectors.

System simulation with Modelica

Modelica ia an object-oriented modeling language for modeling heterogeneous and complex physical systems which are "automatically" translated into differential, algebraic or discrete equation systems.

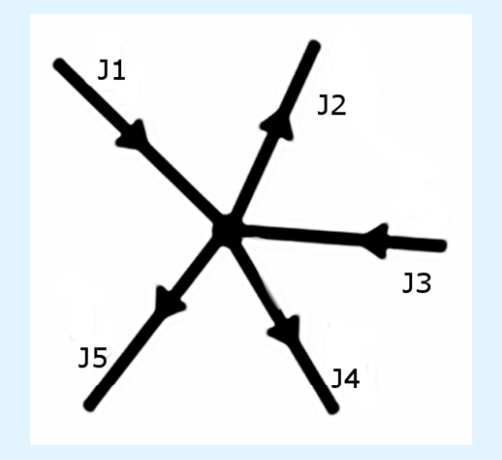
We exploit the similarity between electrical circuits and micro fluidic networks.

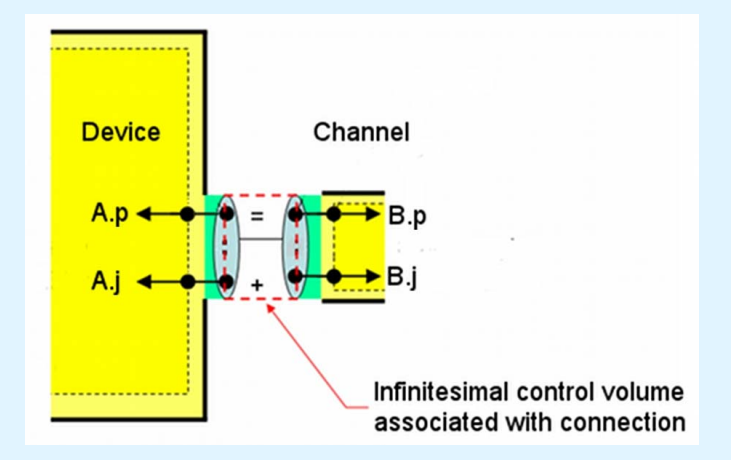
$$U = R_{el} * I \Leftrightarrow \Delta P = R_h * J$$

U	= Voltage	ΔP	= Pressure loss
R_{el}	= Resistance	R_h	= Hydraulic resistance
I	= Current	J	= Flow rate

Kirchhoff's mesh and nodal rules are seamlessly applied to Modelica objects by means of appropriate connection operators.

$$J1 + J3 = J2 + J4 + J5$$





- Define basic element Pin: Representation of a connection point between elements or channels. Pins are the interfaces which communicate all relevant data between two elements or channels.

```
connector Pin
  Pressure p;
  flow Flowrate j;
end Pin;
```

- Define basic element Pin interconnection (Class TwoPin), the physical behavior of all chip elements with two pins is identical:
  a) ΔP denotes the difference between the two pins.
  b) The sum of the flow rates is 0.

```
partial class TwoPin
  Pin    i,o;  // Inlet, Outlet
  Pressure dp; // delta P
  Flowrate j;
equation
  dp=i.p-o.p; //a
  0=i.j+o.j;  //b
  j=i.j;
end TwoPin;
```

- Define all objects based on Pin or TwoPin e.g. channel, pump, ambient, including their specific physical behavior using algebraic notation.

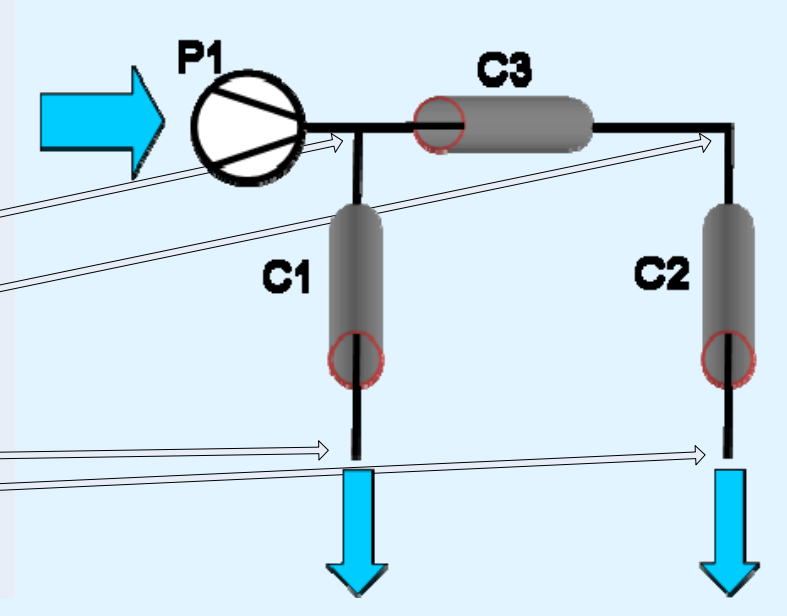
```
class Channel
  extends TwoPin;
  parameter Modelica.SIunits.Distance h = 1;
  parameter Modelica.SIunits.Distance b = 1;
  parameter Modelica.SIunits.Distance l = 1;
  parameter Real n = 20e-6;
equation
  dp = 2 * (4.7 + 19.64 * (( 1 + (h/b)^2)/((1+h/b)^2))) *
         n * ( l / ((2 * ((h*b)/(h+b)))^2)) * (j/(b*h));
end Channel;
```

```
class Ambient
  Pin i;
equation
  i.p=0;
end Ambient;
```

```
class Pump
  Pin o;
  parameter Pressure Press;
equation
  o.p=Press;
end Pump;
```

- Define the LoC network: using the various element attributes stored in the prototype database.
- Define connections between all element pins.

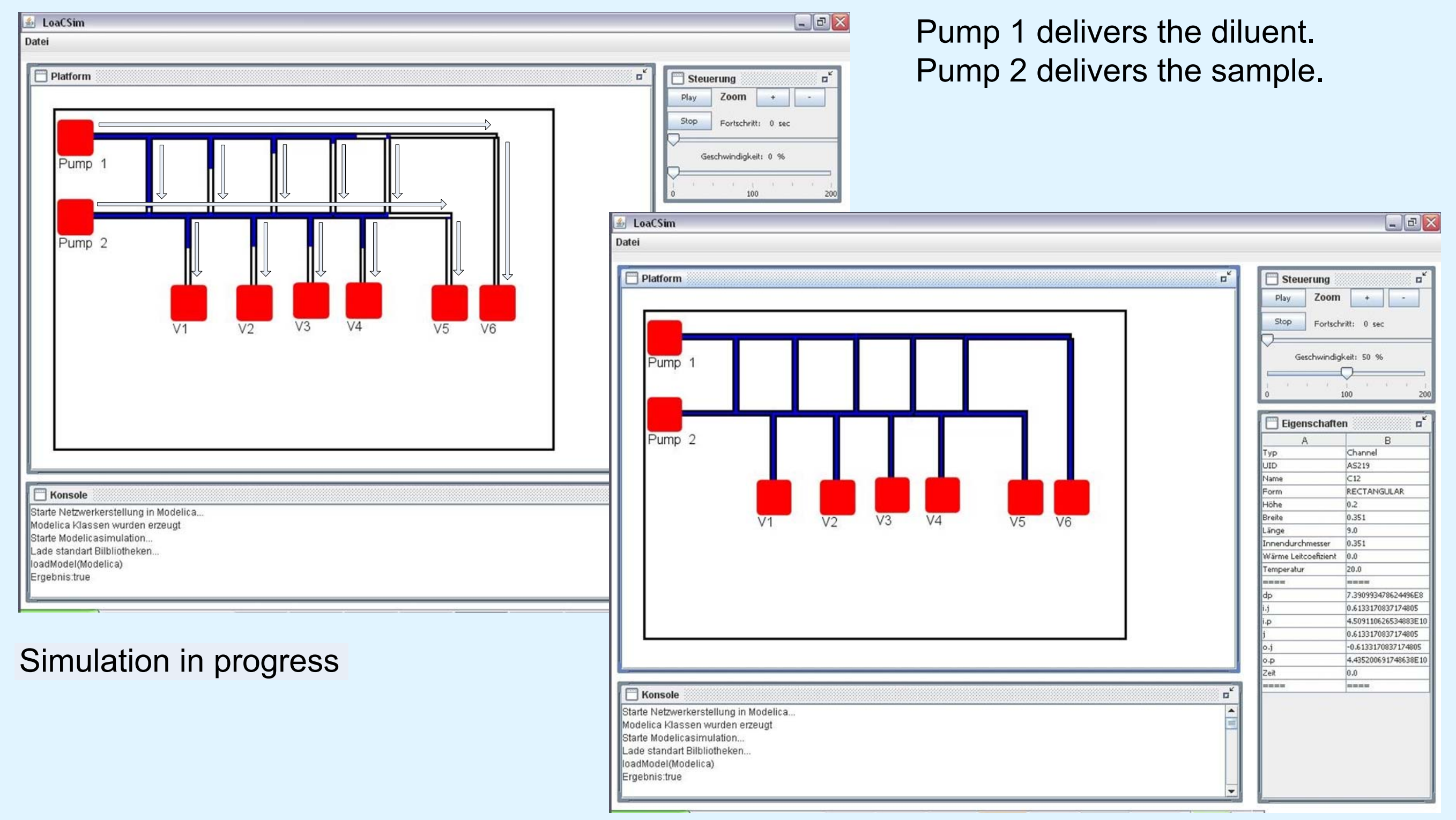
```
class generatedNetwork
  Channel   C1  (b=5.0E-4, h=2.0E-4, l=0.0090);
  Channel   C2  (b=2.5E-4, h=2.0E-4, l=0.0090);
  Channel   C3  (b=2.5E-4, h=2.0E-4, l=0.0090);
  Pump      P1 (Press=10);
  Ambient   Amb;
equation
  connect(P1.o, C1.i);
  connect(P1.o, C3.i);
  connect(C3.o, C2.o);
  connect(C1.o, Amb.i);
  connect(C2.o, Amb.o);
end generatedNetwork;
```



Results

This example shows a μ-fluidic mixing network for continuous delivery of a delusion series where two inlet streams are mixed in ratios $1:2^m$ $(0 \leq m \leq 4)$.

Pump 1 delivers the diluent.
Pump 2 delivers the sample.



Konsole:

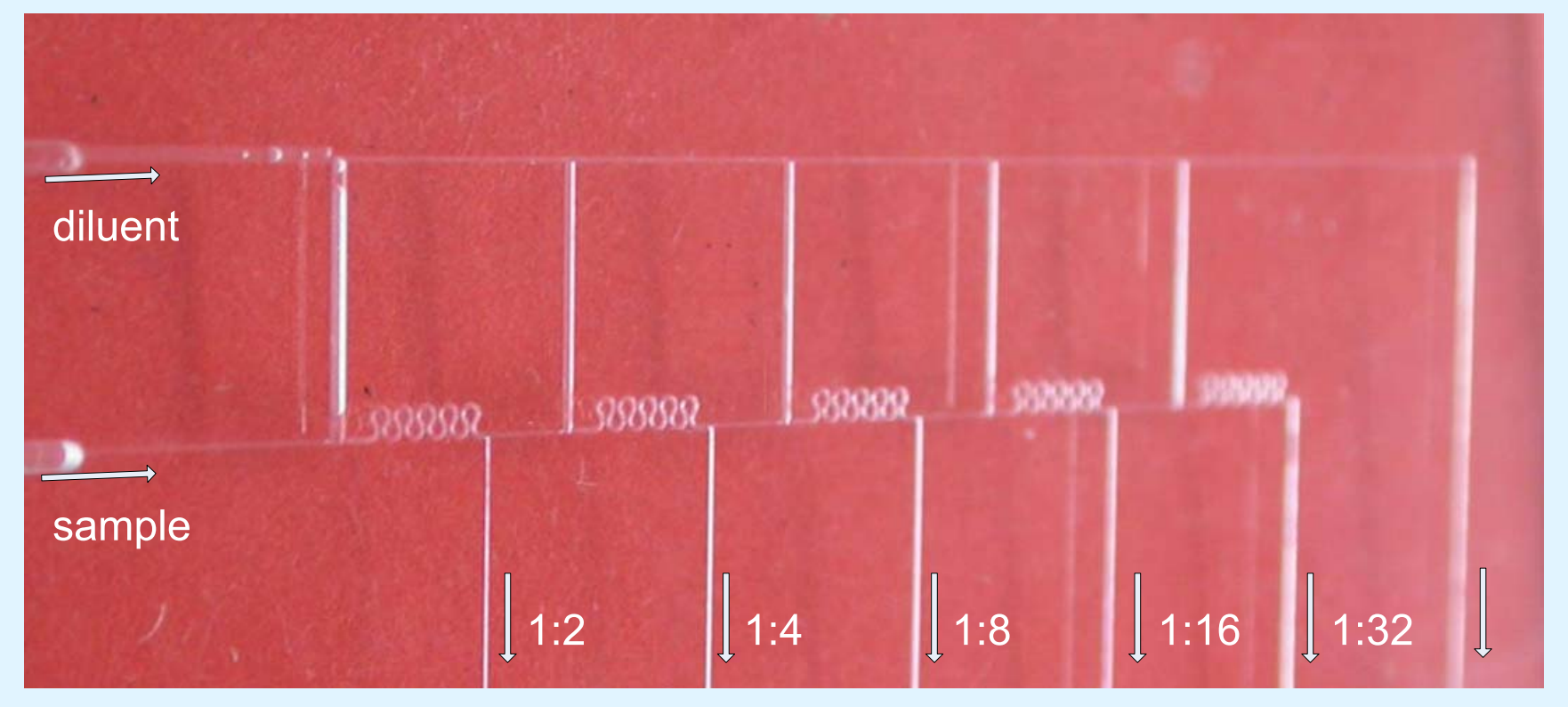
```
Starte Netzwerkerstellung in Modelica...
Modelica Klassen wurden erzeugt
Starte Modelicasimulation...
Lade standart Bilbliotheken...
loadModel(Modelica)
Ergebnis:true
```

A	B
Typ	Channel
UID	AS219
Name	C12
Form	RECTANGULAR
Höhe	0.2
Breite	0.351
Länge	9.0
Innendurchmesser	0.351
Wärme Leitcoeffizient	0.0
Temperatur	20.0
====	====
dp	7.390993478624496E8
i.j	0.6133170837174805
i.p	4.509110626534803E10
j	0.6133170837174805
o.j	-0.6133170837174805
o.p	4.435200691740638E10
Zeit	0.0
====	====

Simulation in progress

Simulation end

- Modelica generates its results as text file.
- Results are parsed and stored into our database.
- Graphical animation of micro fluidic behavior is fed by Modelica results and topological information from the prototype database.
- Actual property values of platform elements can be displayed by mouse click selection.
- The actual Modelica calculation status is displayed in a log window.



Photograph of a prototype designed for dilution of a sample fluid

Conclusions

- The object and physical model oriented programming language Modelica has been successfully applied for a generic description of Lab-on-a-Chip systems and platforms.
- By using this software simulation the effort for calculation and demonstration of results can be significantly simplified and accelerated with a minimum of prototyping.
- An extensible library, i.e. a collection of predefined Modelica classes, has been set up.
- Software provides not only simulation but also workflow management, physical control, data acquisition, and graphical display of real LoC platforms.
- The software solution was sufficiently tested on several examples with different complexity.

Outlook

The software solution is going to be developed further on. There are many options to extend it with additional features.

- The workflow to physical control real LoC platforms will be adapted to also control the simulation.
- Elements changing their behavior at runtime will be implemented, e.g., a pump changing its output pressure.
- Capillary filling of channels will be implemented.
- As a further feature a 3D graphical animation of the platform is planed.
- Manipulation of devices at runtime, e.g. closing a valve by mouse click, will be added.
- Mixing and chemical reactions will be incorporated.

Acknowledgements

The great work of Jens Barkoczy (FHW), Ronny Jopp (FHW), and Malte Weniger (IMM) is highly acknowledged. They built the workflow management, physical control, and data acquisition of real LoC platforms.