

# Integrated Building System Modeling & Simulation

Michael Wetter

Simulation Research Group

Building Technologies Department

Energy and Environmental Technologies Division

Lawrence Berkeley National Laboratory

April 21, 2009



# Overview

- Introduction

Dynamics – Complexity – Multidisciplinary Systems

- Problems with Building Simulation

Modeling vs. Simulation





- Modeling for Complex Heterogeneous Systems

Modelica

- Need for Co-Simulation

Building Controls Virtual Test Bed

# Introduction – Building Dynamics

Gebäudetyp	$T_{AK} [h]$
geringe Wärmespeicher-masse 	15
mittlere Wärmespeicher-masse 	40
grosse Wärmespeicher-masse 	100
hoch-wärmege-dämmte Gebäudehülle 	200

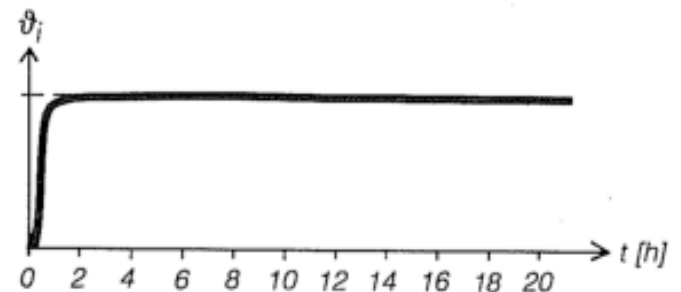
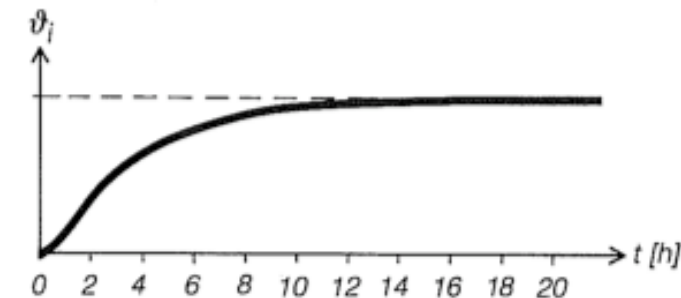
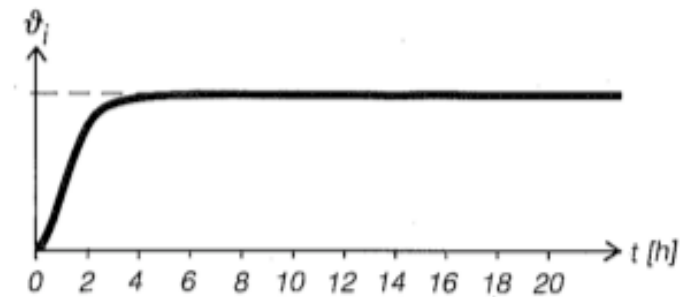
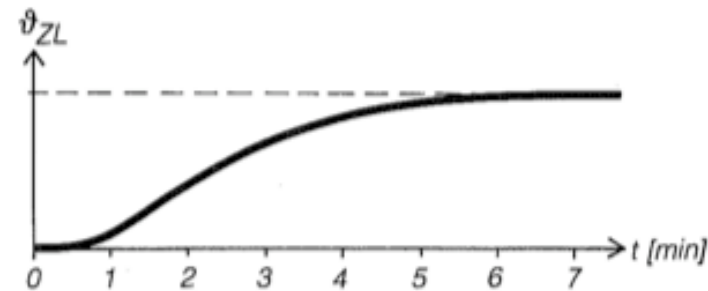
$$\tau = C/UA$$

15 h

40 h

100 h

200 h



# Introduction – Building Dynamics

1<sup>st</sup> order approximation to building temperature

$$\frac{dT}{dt} = \frac{K}{C} (T_{\infty} - T) + \frac{1}{C} \dot{Q}_d + \frac{1}{C} u$$

$$\tau = \frac{C}{K}$$

Low energy buildings

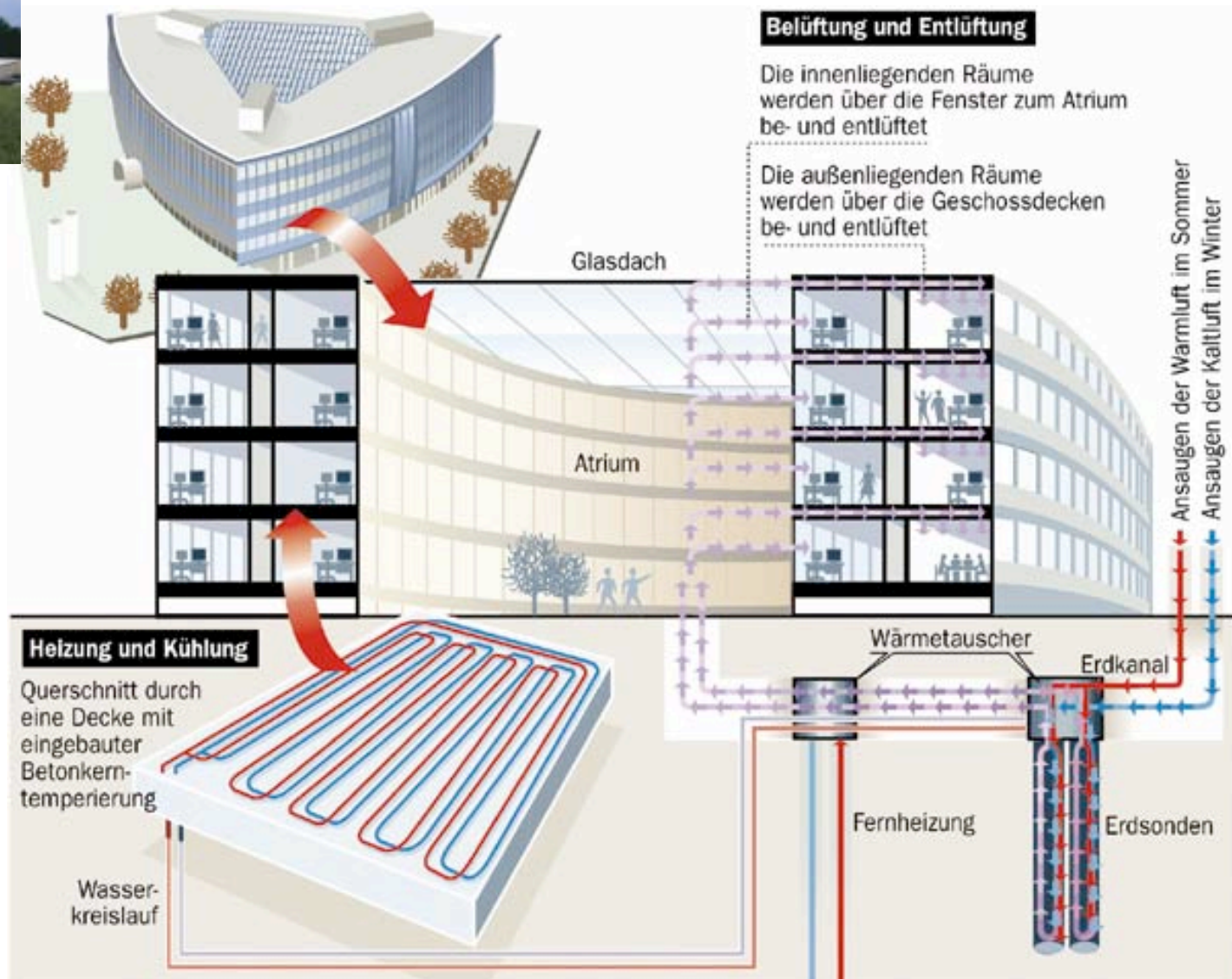
$$\begin{array}{c} K \\ C \\ \tau \end{array} \begin{array}{c} \searrow \\ \nearrow \\ \nearrow \end{array}$$

Approximate equation

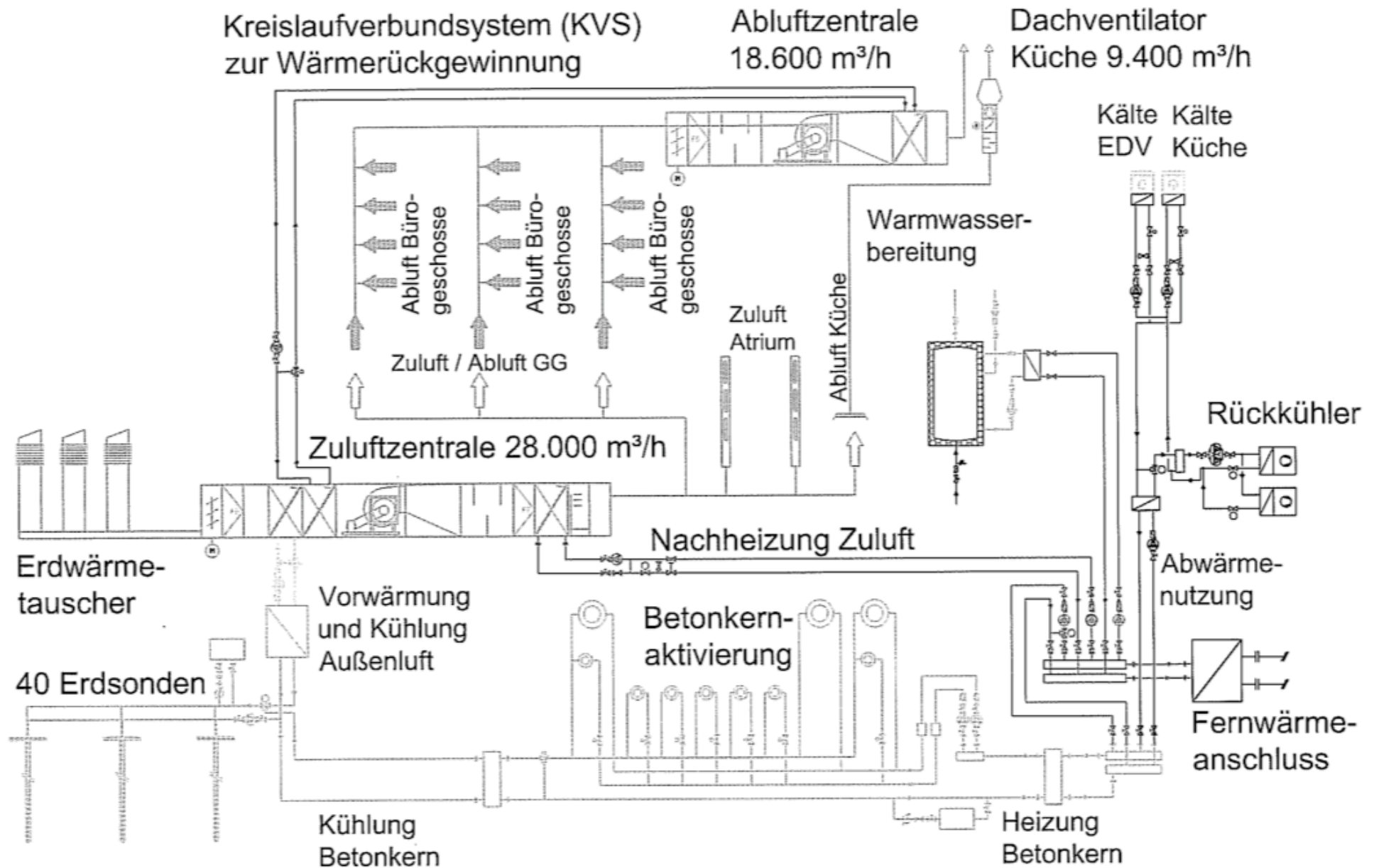
$$\longrightarrow C \frac{dT}{dt} = \dot{Q}_d + u$$



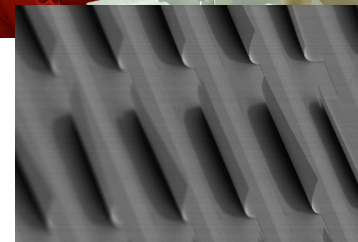
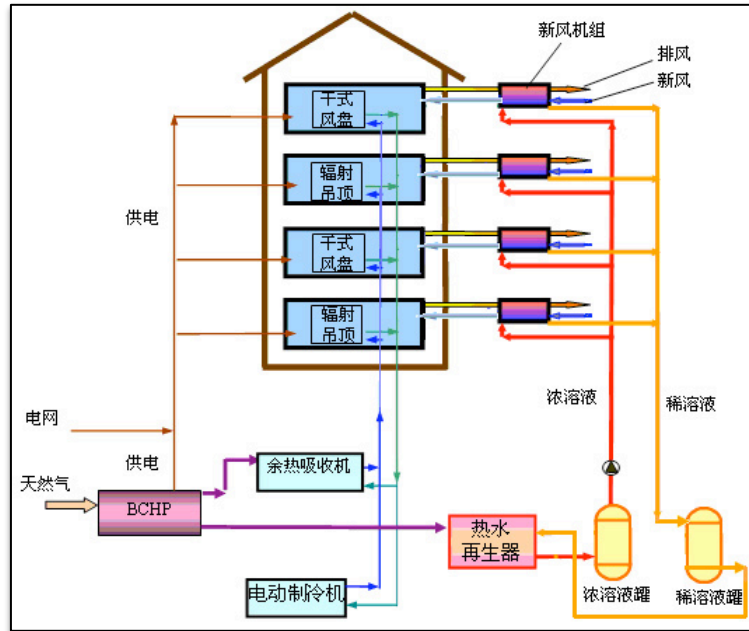
# Introduction – Complexity



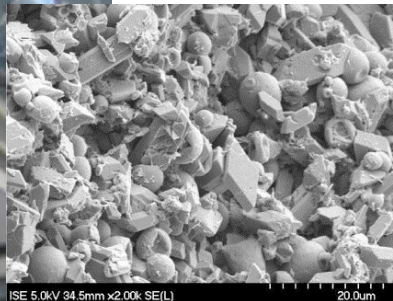
# Introduction – Complexity



# Introduction – Interdisciplinary Nature



100  $\mu\text{m}$





# Overview

Most innovation happens at the interface between disciplines.

System-level analysis becomes of increasing importance.

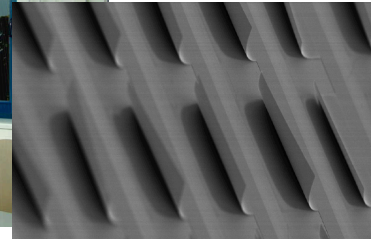
Computational Science and Engineering reduces cost and time to market, but needs flexible tools

- for rapid prototyping
- to identify and fix mistakes early

Buildings become a computationally rich environment.



[www.micronal.de](http://www.micronal.de)



# Modeling Tools



energy

controls

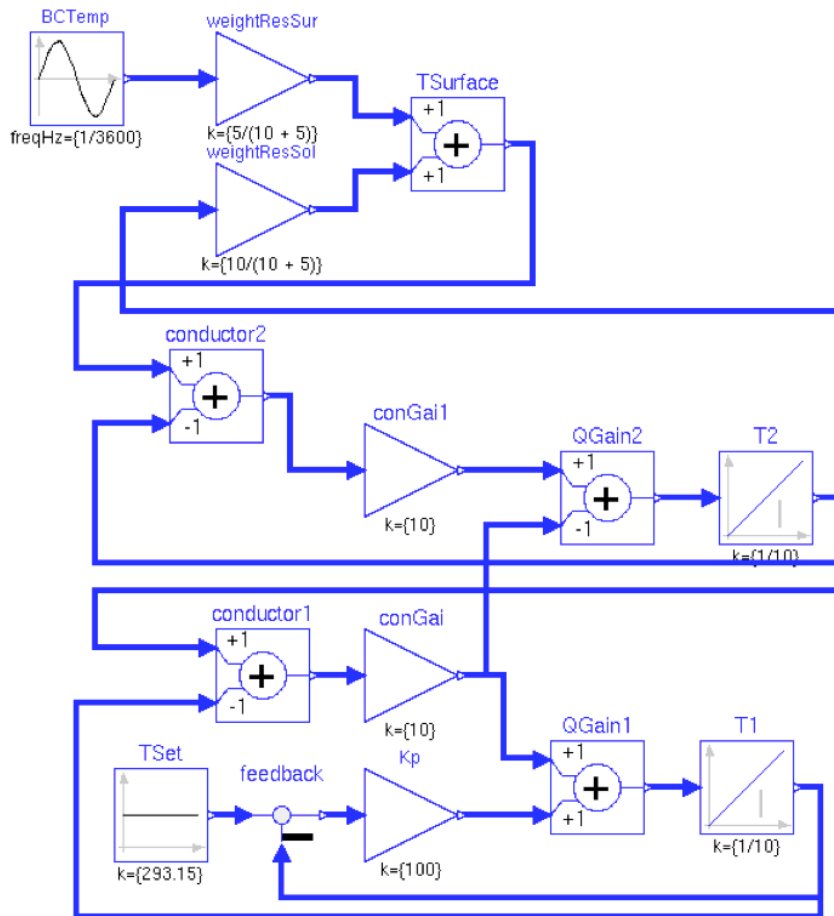
lighting

communi-  
cation

airflow

- Fragmentation within domain
- Little integration across domains
- Designed for time domain simulation
- Little outreach to other disciplines

# Modeling Tools



- Not transparent
- Physics is acausal, but we write causal models
- Little hierarchy

```

open('log', 'w', 'results.txt')
! Perform integration
do i = 1, N, 1
    TBC = T0 + amp * dsin(2*3.14159*time/3600)
    TSur = T2 * ( G / (G+h) ) + TBC * ( h / (G+h) )
    QSou = Kp * (TSet - T1)
    QCon1 = G * (T2 - T1)
    QCon2 = G * (TSur - T2)
    derT1 = 1/C * ( QCon1 + QSou )
    derT2 = 1/C * ( -QCon1 + QCon2 )
    if (i.EQ.iCom) then
        write(lun,FMT) time, T1, T2, QSou
        iCom = iCom + NCom
    endif
! Update variables

```

# Next Generation System Modeling

## Enables rapid prototyping.

## Allows intuitive modeling.

## Accelerates system-level innovation.

## Procedural modeling $\approx$ 1970

```

program euler
implicit none
double precision, parameter :: tFin = 7200    † Final time
integer, parameter :: N = 72000              † Number of steps
integer, parameter :: NCon = 10               † Communication interval
integer :: iCon                                † Communication counter

double precision dT                             † Time step

double precision, parameter :: T0 = 293.15    † Initial temp.
double precision :: T1, T2                    † Temperature
double precision :: TBC                       † Temp. boundary condition
double precision :: TSur                      † Surface temperature
double precision, parameter :: TSet = 293.15   † Set point temp.
double precision :: derT1, derT2              † Temperature derivative

double precision :: QSou, QCon1, QCon2        † Heat flux

double precision, parameter :: Kp = 100       † P Gain
double precision, parameter :: h = 5          † Convective heat transfer coefficient
double precision, parameter :: G = 10        † Conductivity
double precision, parameter :: C = 10        † Capacity

integer :: i                                  † Loop counter
integer, parameter :: lun = 6                 † Logical unit number

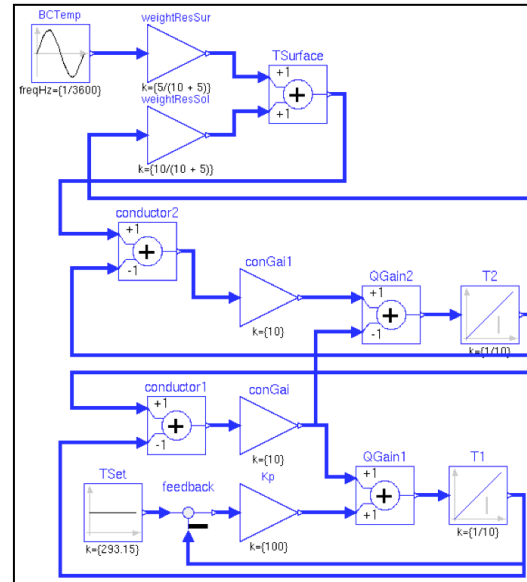
double precision, parameter :: amp = 5        † Amplitude
double precision :: time                      † Simulation time

character(LEN=*) , parameter :: FMT = "(4F14.7)"

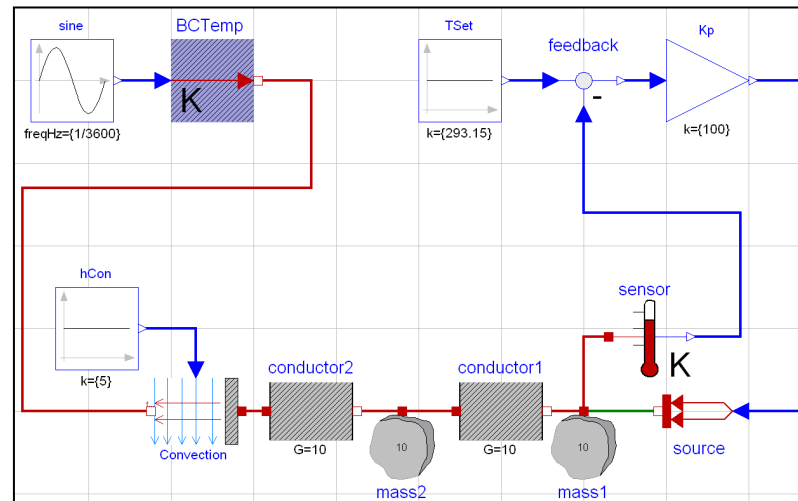
† Initialize variables
dt = tFin/N
T1 = T0
T2 = T0
time = 0
iCon = 1
open (lun, FILE='results.txt')
† Perform integration
do i = 1, N, 1
    TBC = T0 + amp * dsin(2*3.14159*time/3600)
    TSur = T2 * ( G / (G+h) ) + TBC * ( h / (G+h) )
    QCon1 = G * (T2 - T1)
    QCon2 = G * (TSur - T2)
    derT1 = 1/C * ( -QCon1 + QSou )
    derT2 = 1/C * ( -QCon1 + QCon2 )
    if (i.EQ.iCon) then
        write(lun,FMT) time, T1, T2, QSou
        iCon = iCon + NCon
    endif
† Update variables
    T1 = T1 + dt * derT1
    T2 = T2 + dt * derT2
    time = time + dt
end do
write(lun,FMT) time, T1, T2, QSou
close(lun)
write(*,*) 'Program finished'
end_program

```

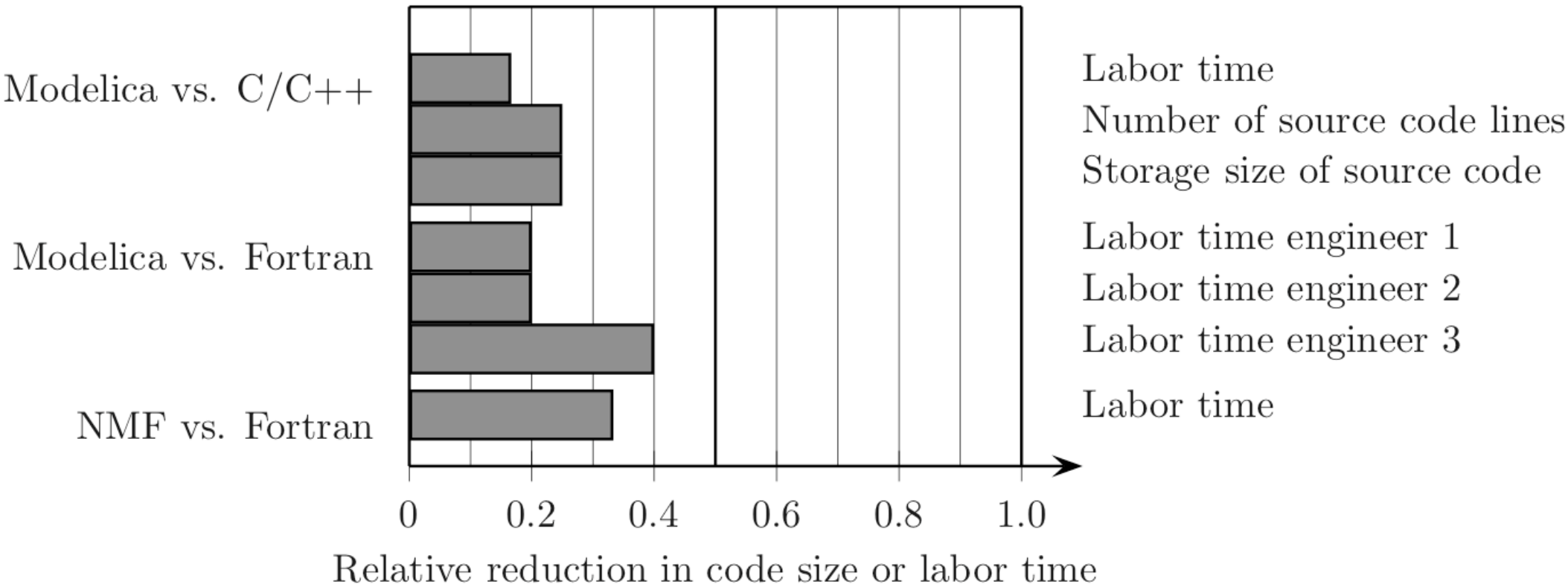
## Block diagram modeling ≈ 1990



## Equation-based, object-oriented modeling ≈ 2000

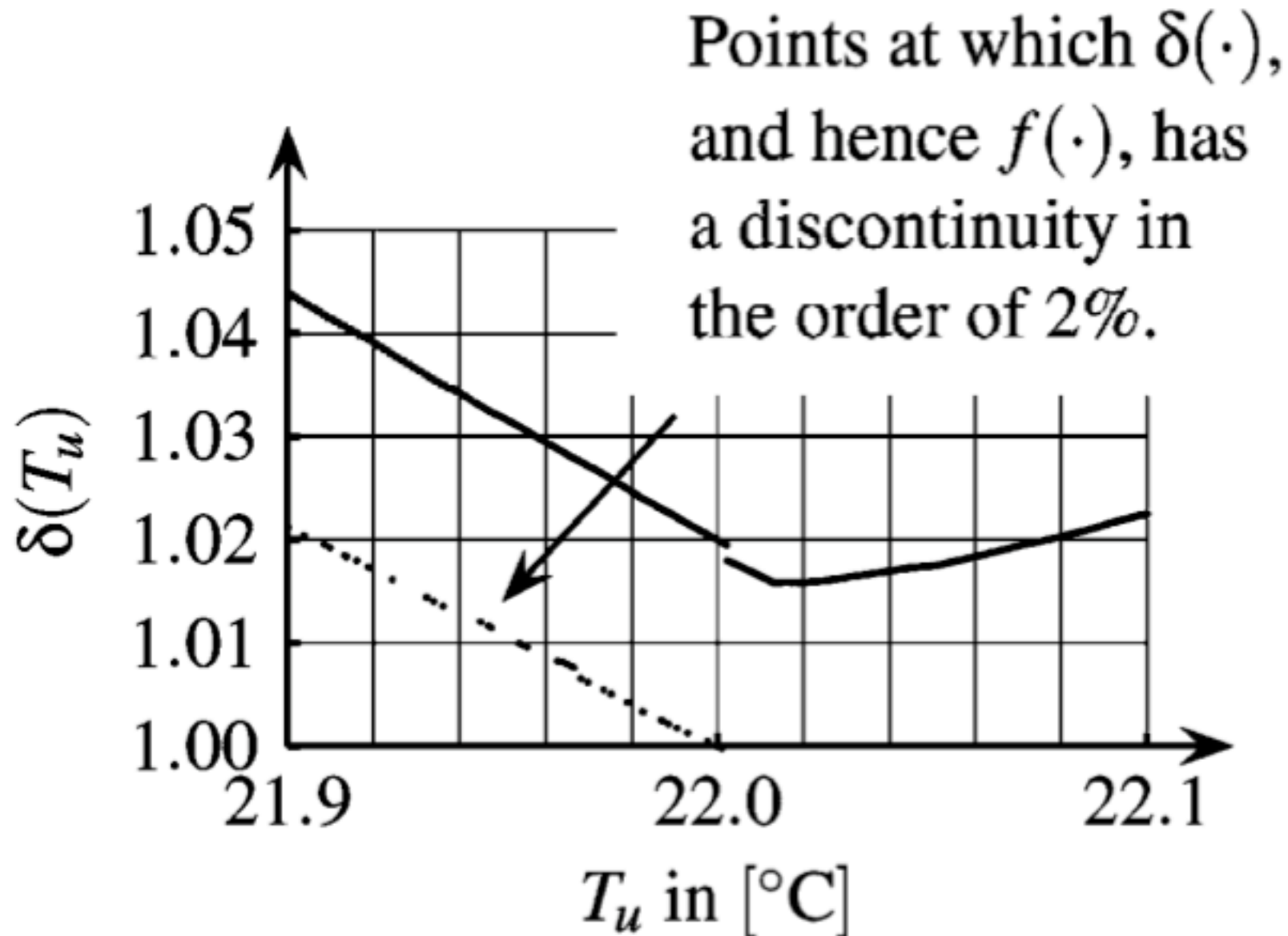


# Model Development Time

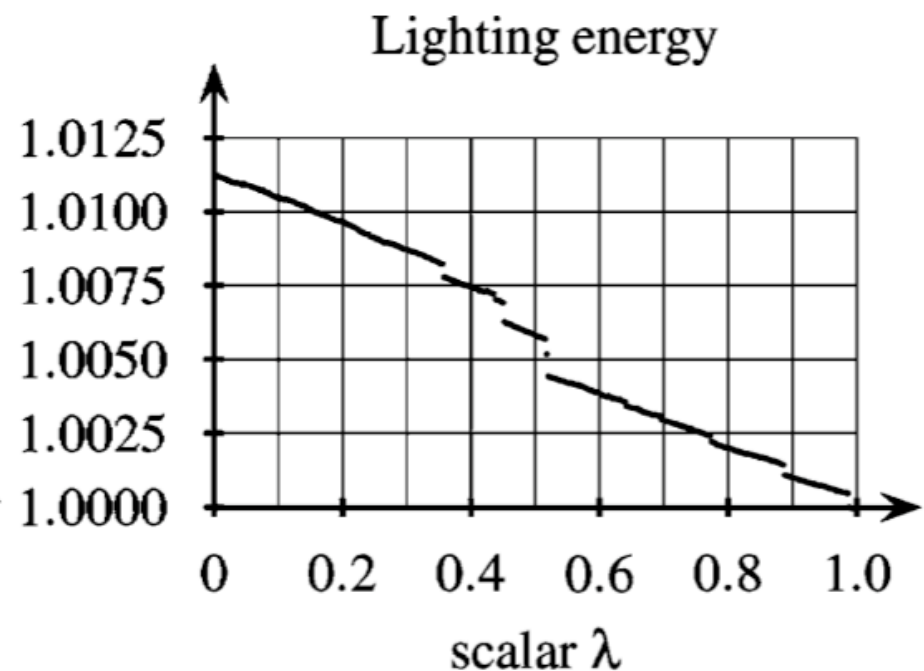
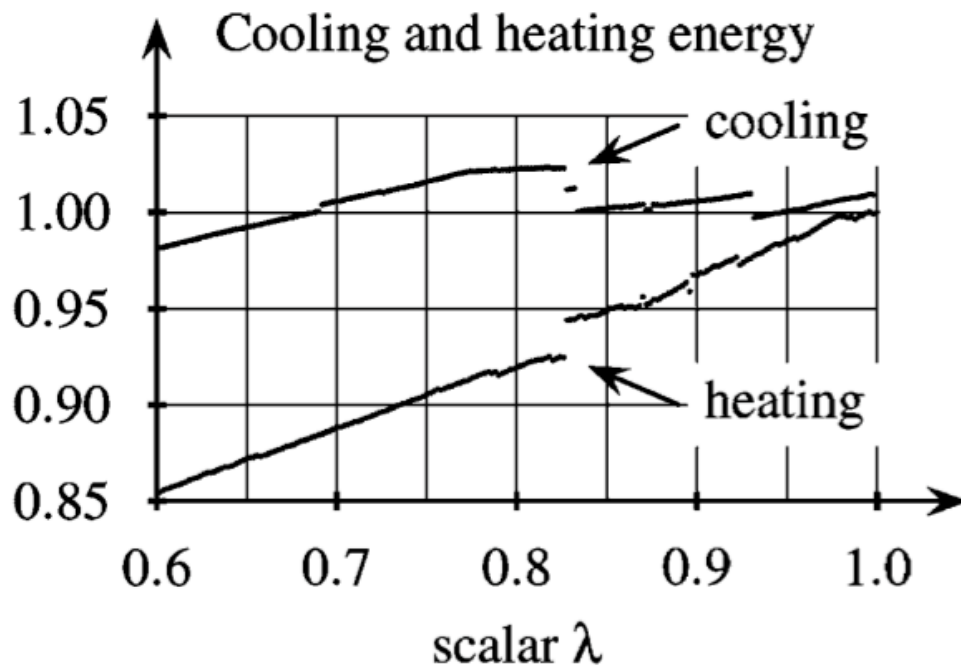
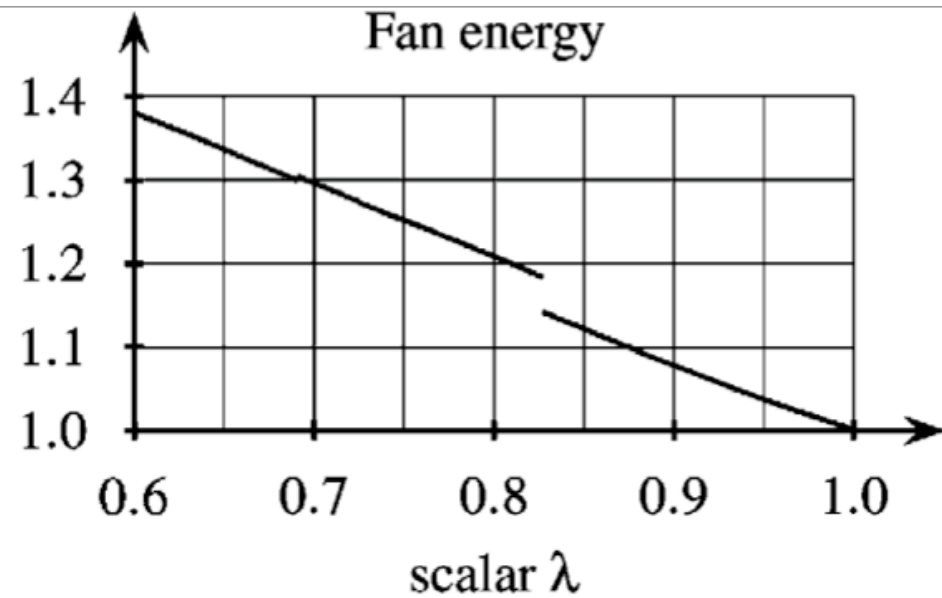
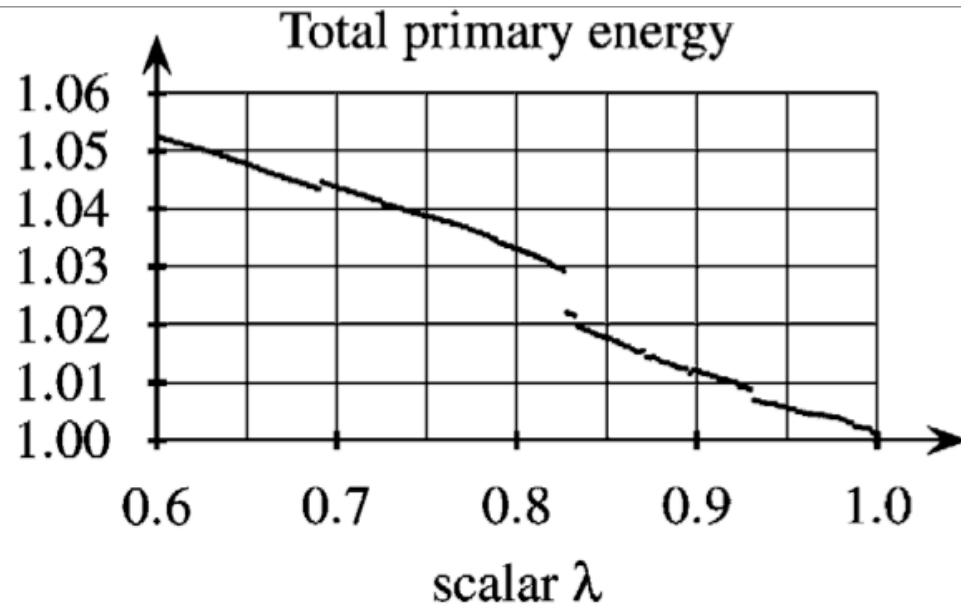




# Numerical “Noise” in Building Simulation



# Numerical “Noise” in Building Simulation

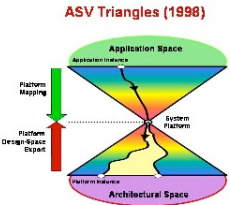


# Problem

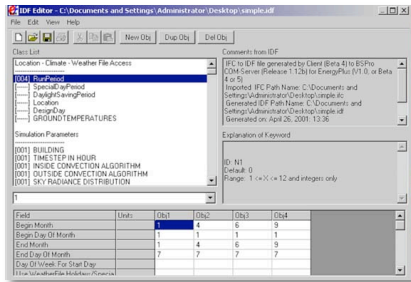
- Building simulation programs are not designed for multi-disciplinary analysis
- Adding models takes months
- Sharing models & data is hard
- Controls representation has little in common with actual controls
- Tools are difficult to use for
  - analysis
  - innovative systems
- Rely heavily on expensive full scale experiments

# Computational Science and Engineering to Accelerate Innovations for Buildings

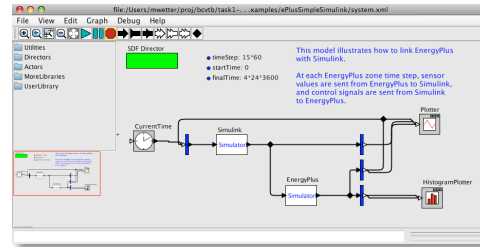
## Platform Based Design



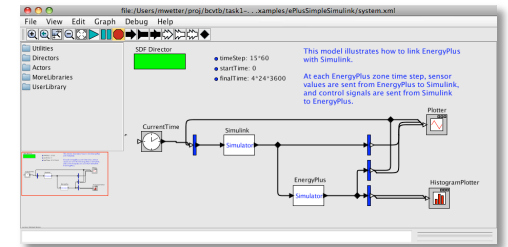
## Envelope



## Integrated simulation & analysis

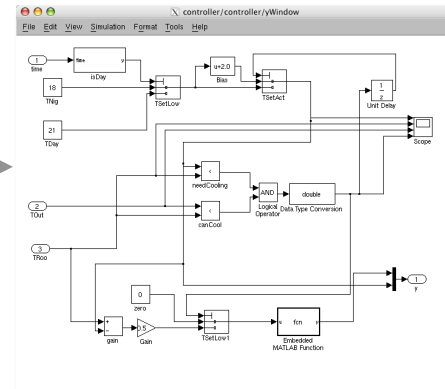


## Hardware in the loop

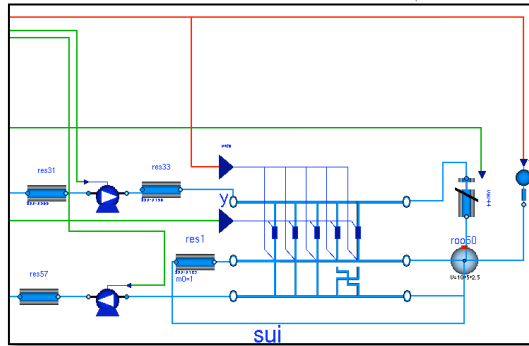


## Realtime synchronization

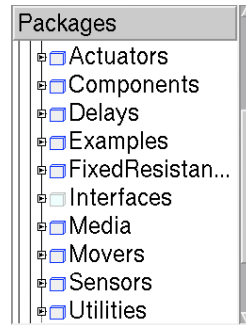
## Controls



## HVAC



## Component library



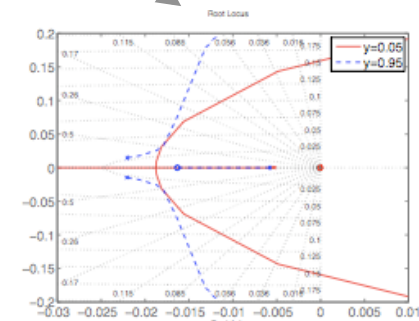
rapid prototyping  
for HVAC systems

## verification



code generation

## design and analysis

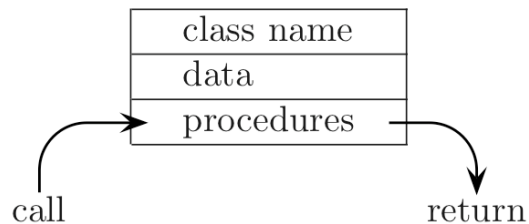


# Separation of Concerns

Structure the problem the way you think, not how you compute a solution

**Traditional approach** – Describes how to **compute**

*procedural*



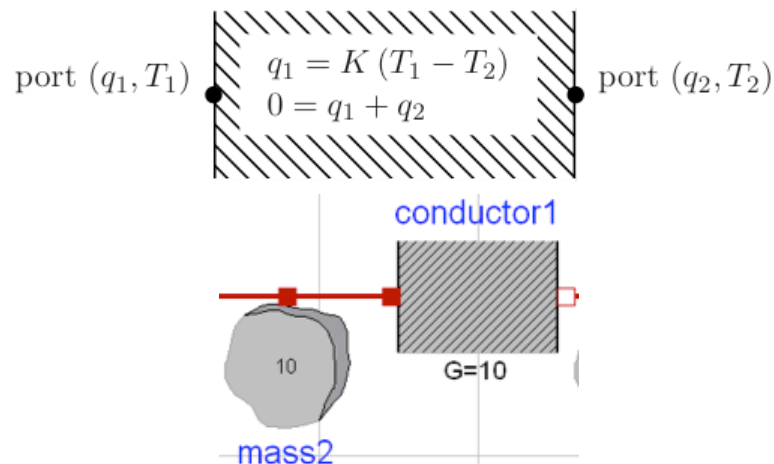
```
IF (GetMixerInputFlag) THEN
    CALL GetMixerInput
    GetMixerInputFlag=.false.
ENDIF

CALL GetConnectorList(ConnectorListName,Connectoid,Conn
IF (Connectoid%ConnectorType(1) == MIXER) THEN
    Count=FindItemInList(Connectoid%ConnectorName(1),Mixe
    IF(PRESENT(MixerNumber)) MixerNumber = MixerNumber +
    IF (Count == 0) THEN
        CALL ShowFatalError('GetLoopMixer: No Mixer Found='
    ENDIF
```

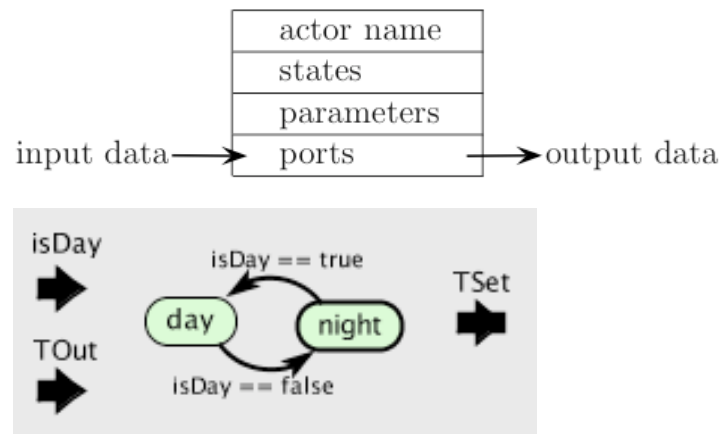
80% of source code is for data management.

**New approach** – Describes **physics** & **control logic**

*equation-based*



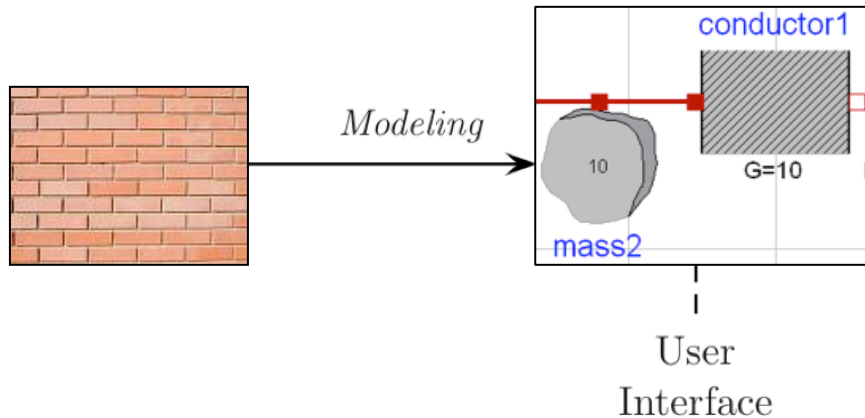
*actor-oriented*



3 studies show 3-5 times faster development time using equation-based modeling.

# Modeling vs. Simulation

## Modeling

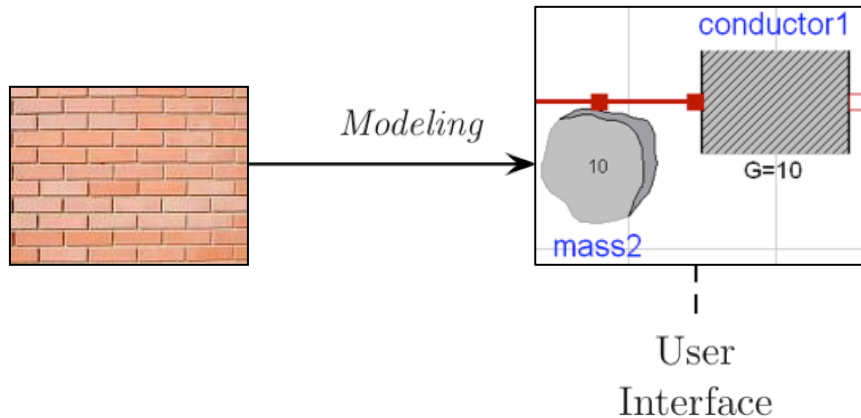


*Describes the phenomena*

- Standardized interfaces
- Acausal models
- Across & through variables
- Hierarchical modeling
- Class inheritance

# Modeling vs. Simulation

## Modeling



*Describes the phenomena*

- Standardized interfaces
- Acausal models
- Across & through variables
- Hierarchical modeling
- Class inheritance

## Compilation & Simulation

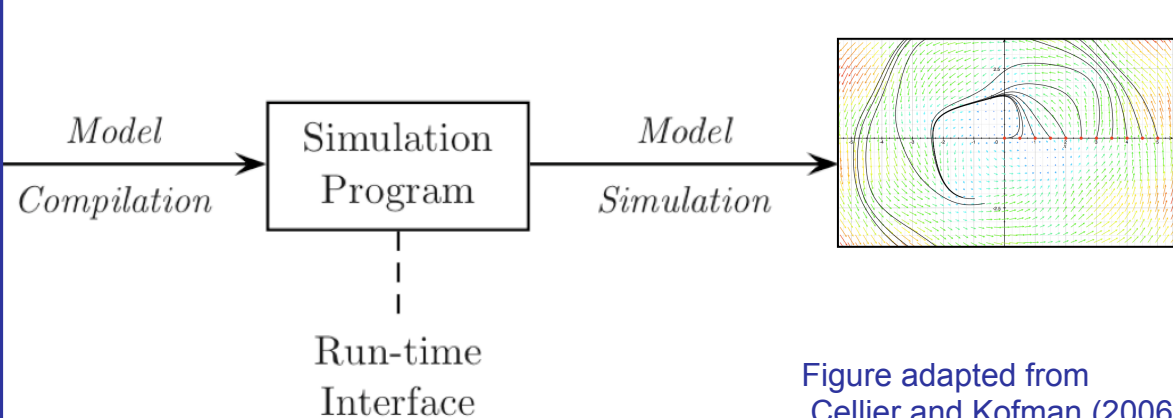
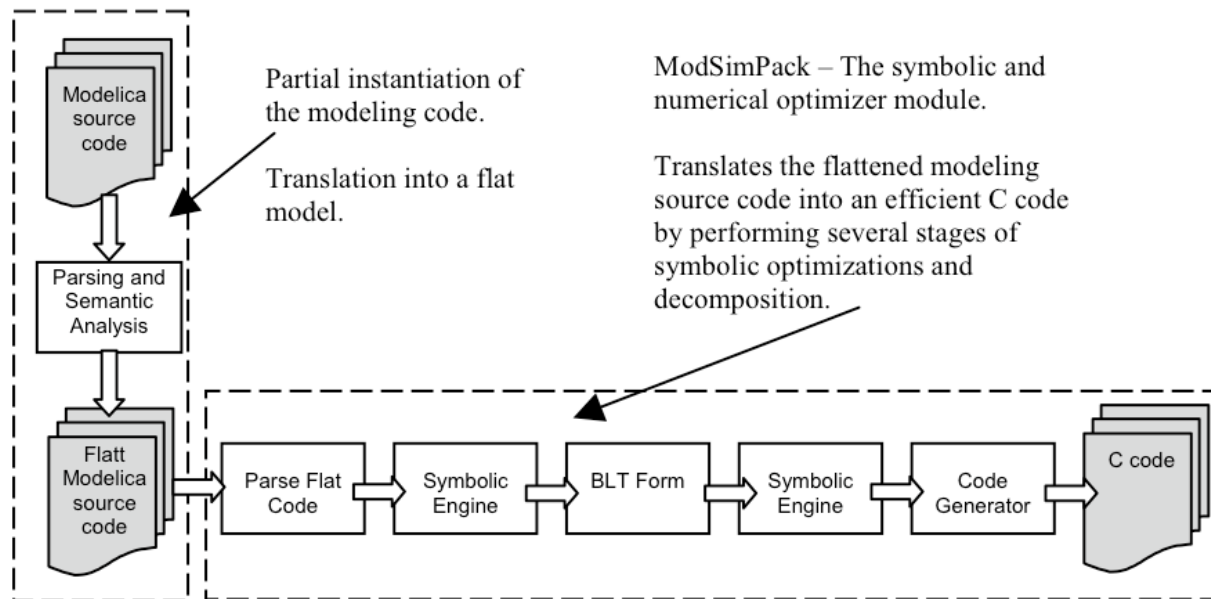


Figure adapted from  
Cellier and Kofman (2006)

*Solves the equations*

- Partitioning
- Tearing
- Inline integration
- Adaptive solver
  - Integration
  - Nonlinear equations

# Exploit Advances in Computer Science



Symbolic processing is key to run-time efficiency.

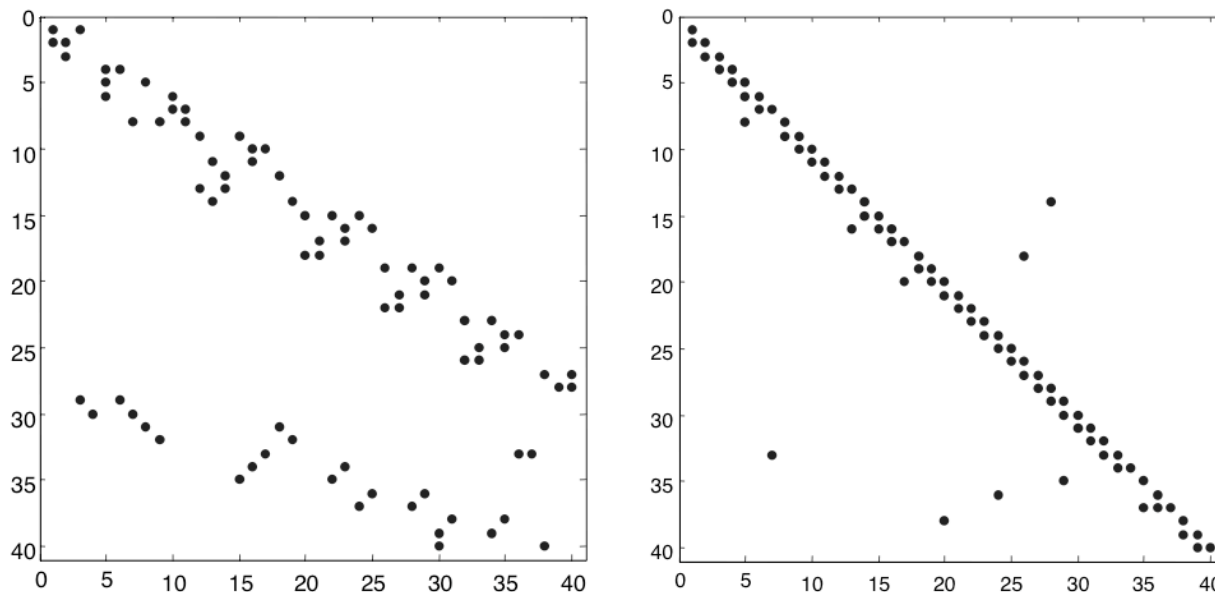
Can

- control the “numerical noise” (important for optimization etc.)
- invert models
- interface with analysis packages
- use optimized libraries

Many options for parallel computing

Recall:

Computing time  
 $\sim O(n^2)$  to  $O(n^3)$ .



Figures from Bunus and Fritzson (2004)



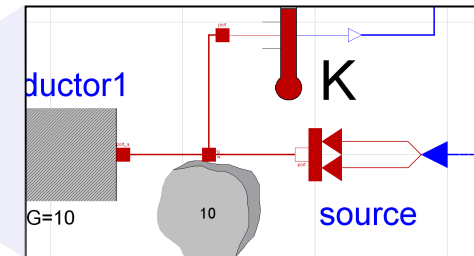
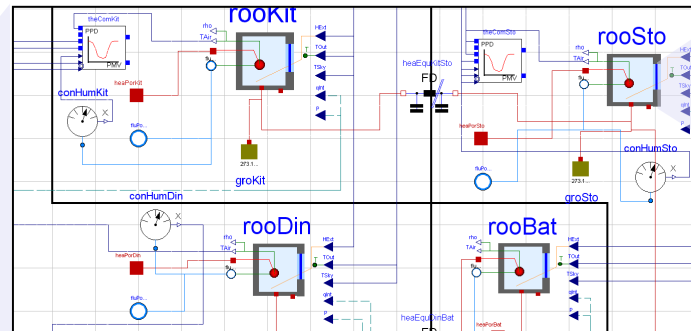
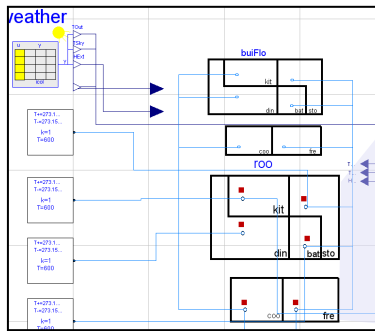
# Modelica: Promising Language for Model Exchange

- Developed since 1996 by Modelica Association
- Designed by developers of
  - Allan, Dymola, NMF, ObjectMath, Omola, SIDOPS+, Smile
- Well positioned to become de-facto standard for modeling multiengineering systems
  - e.g.: ITEA2: 285 person years investment in Modelica-related technologies over next three years.
- Supports differential, algebraic and discrete equations
- Equation-based
- Object-oriented
- Automatic documentation

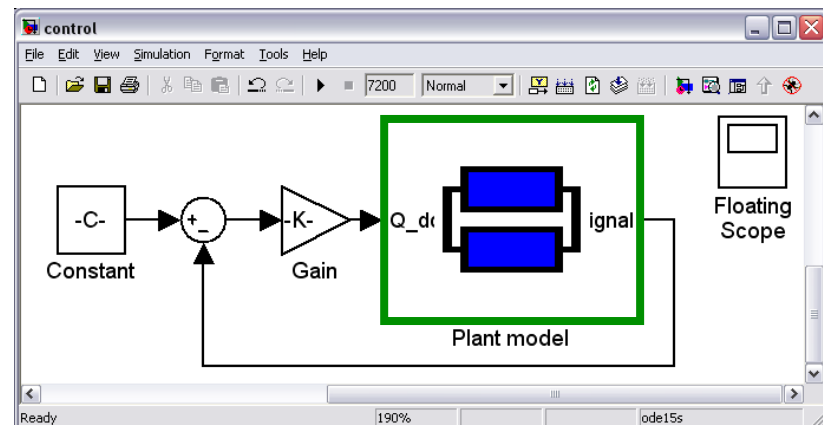
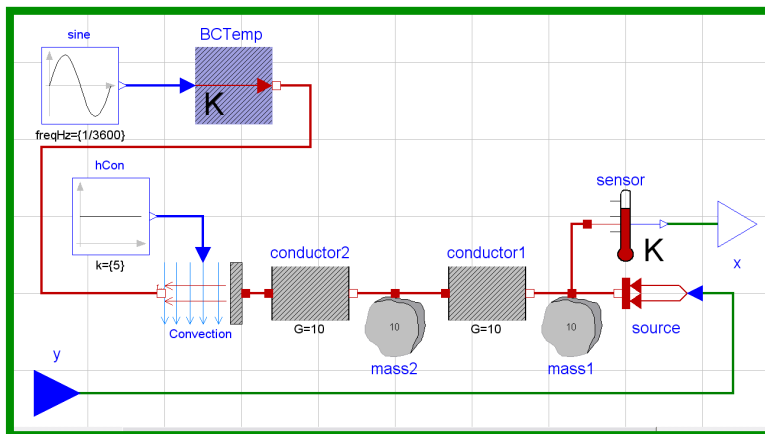


# Modelica Modeling

- Construct models **graphically**
- Use **hierarchies** to manage complexity



- Separate equations and solvers to enable
  - model reuse** for controls & operation
  - advanced symbolic and numerical **mathematics**



# LBNL Buildings Library

**Open-source, free:** <https://gaia.lbl.gov/bir>

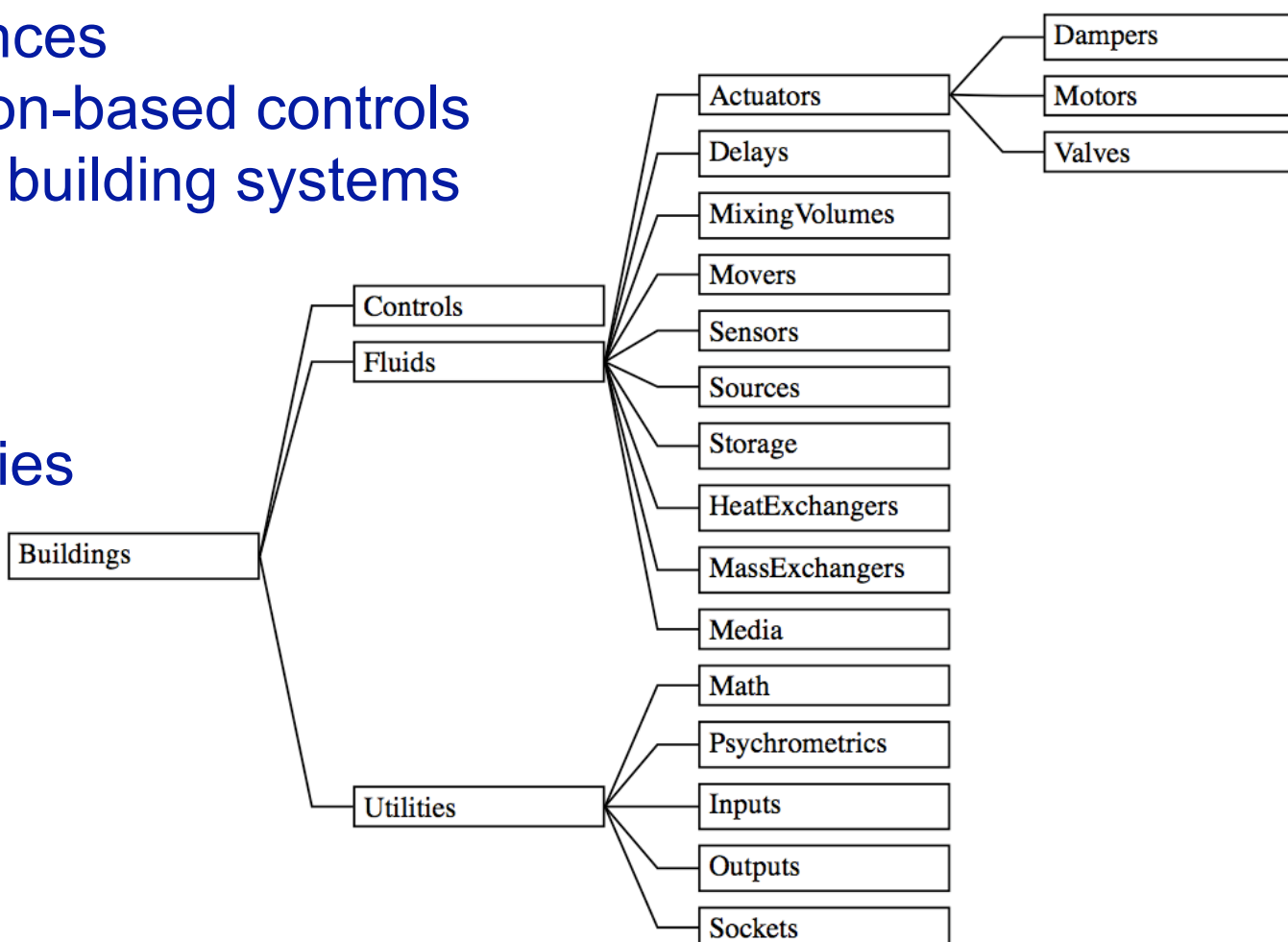
Component models with **60 examples** (to be further developed).

## Initial applications

- Test control sequences
- Develop optimization-based controls
- Analyze innovative building systems

## Initial target audience

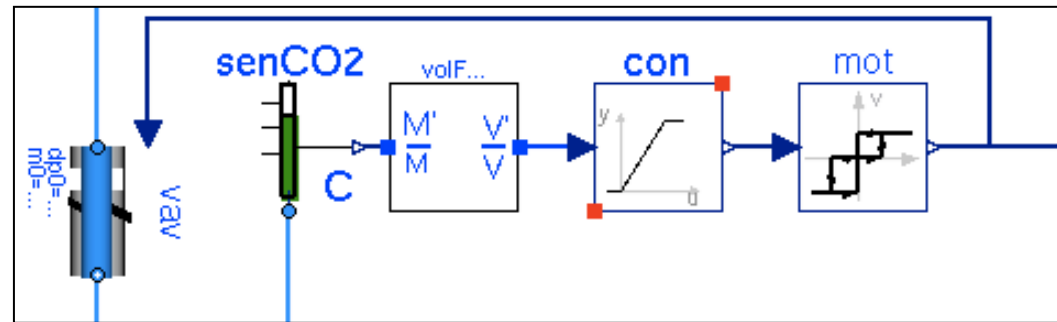
- R&D
- Innovative companies
- Education



# Usage Levels

## Model user

Drags & drops component models to form system model



## Model developer

Reuses base models to implement new models

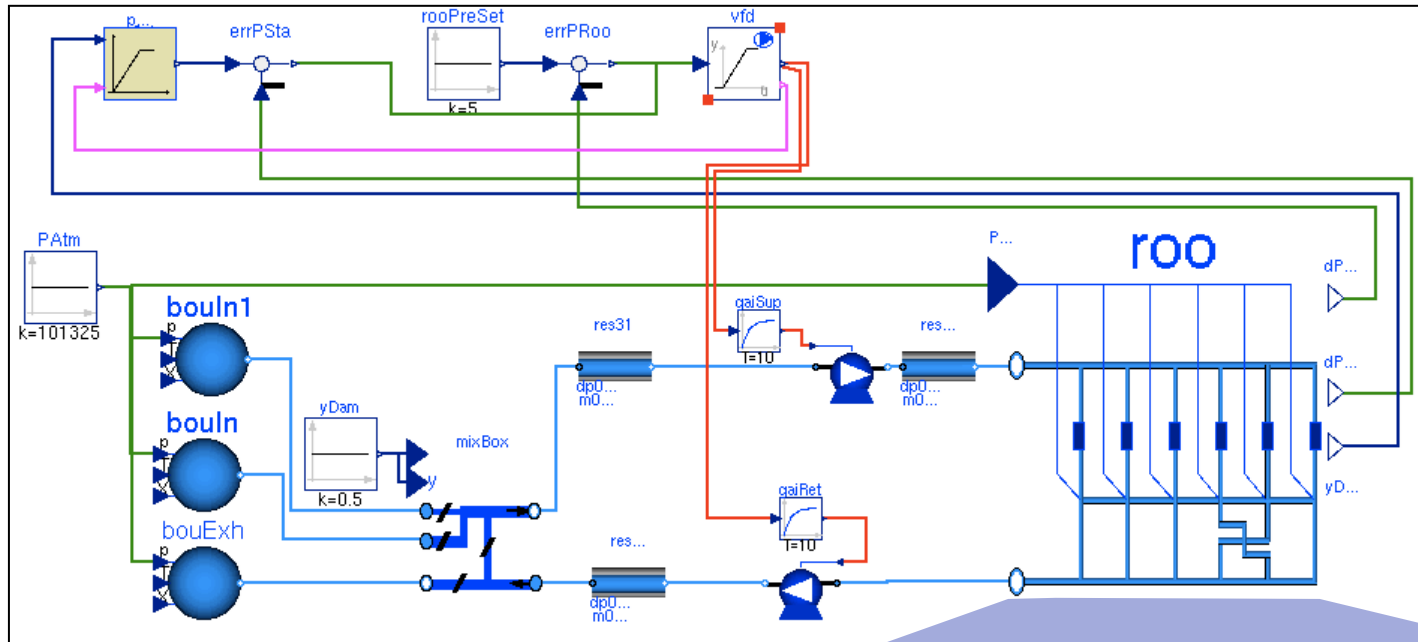
```
Q_flow = Q0_flow * u;  
mXi_flow = zeros(Medium.nXi);
```

## Library developer

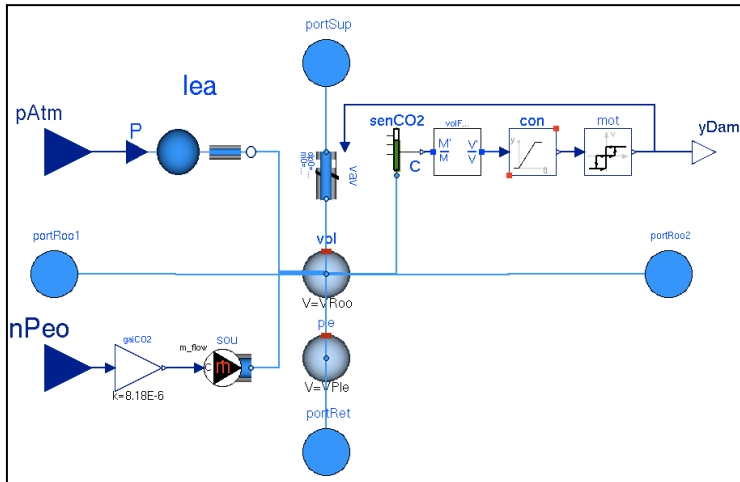
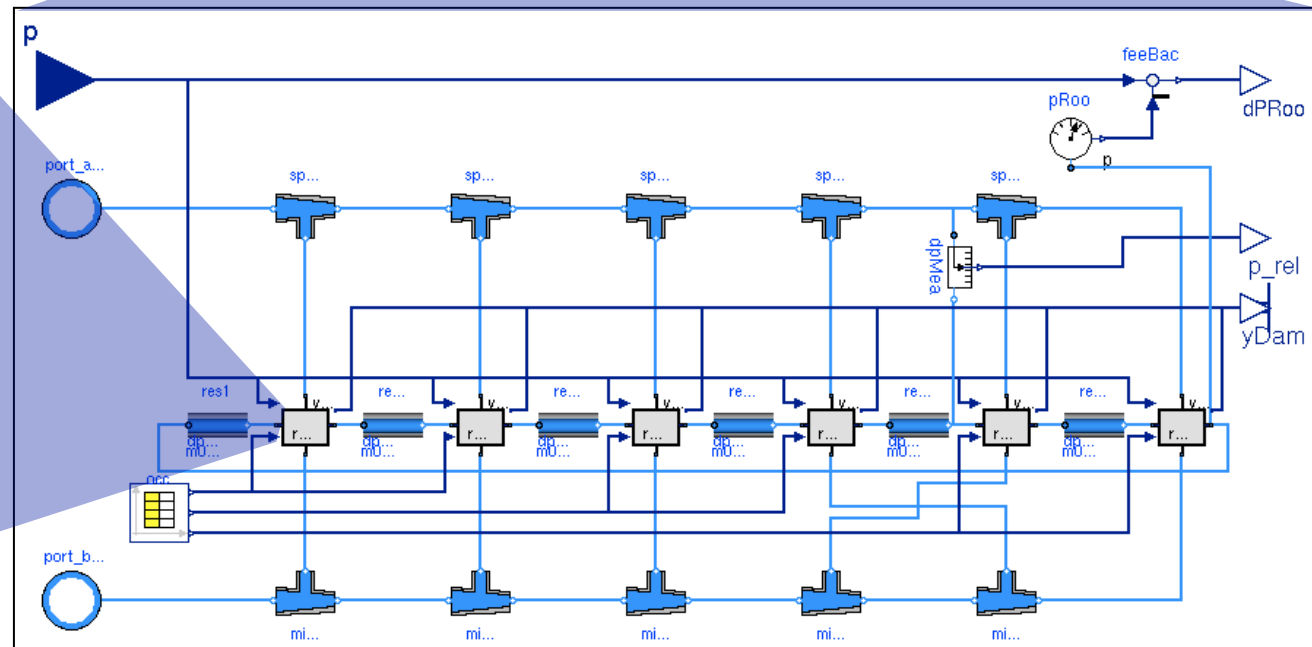
Develops base models for characteristic components

```
port_a.m_flow*port_b.h_outflow +  
port_b.m_flow*inStream(port_a.h_outflow) = Q_flow;  
port_a.m_flow + port_b.m_flow = -sum(mXi_flow);
```

# Modeling for Controls Analysis



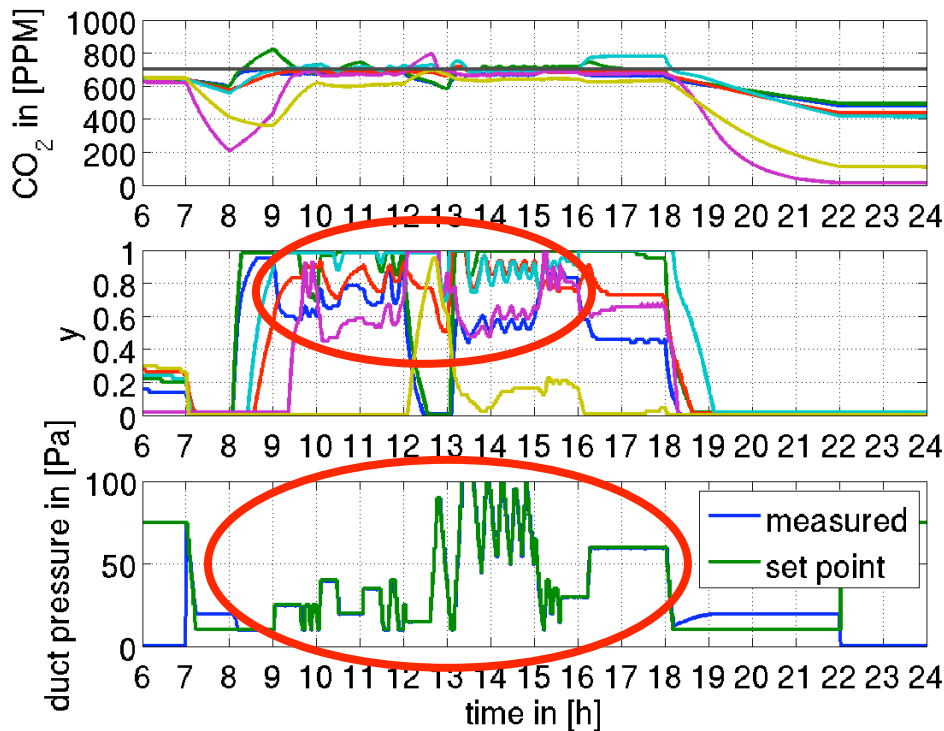
## VAV System (ASHRAE 825-RP)



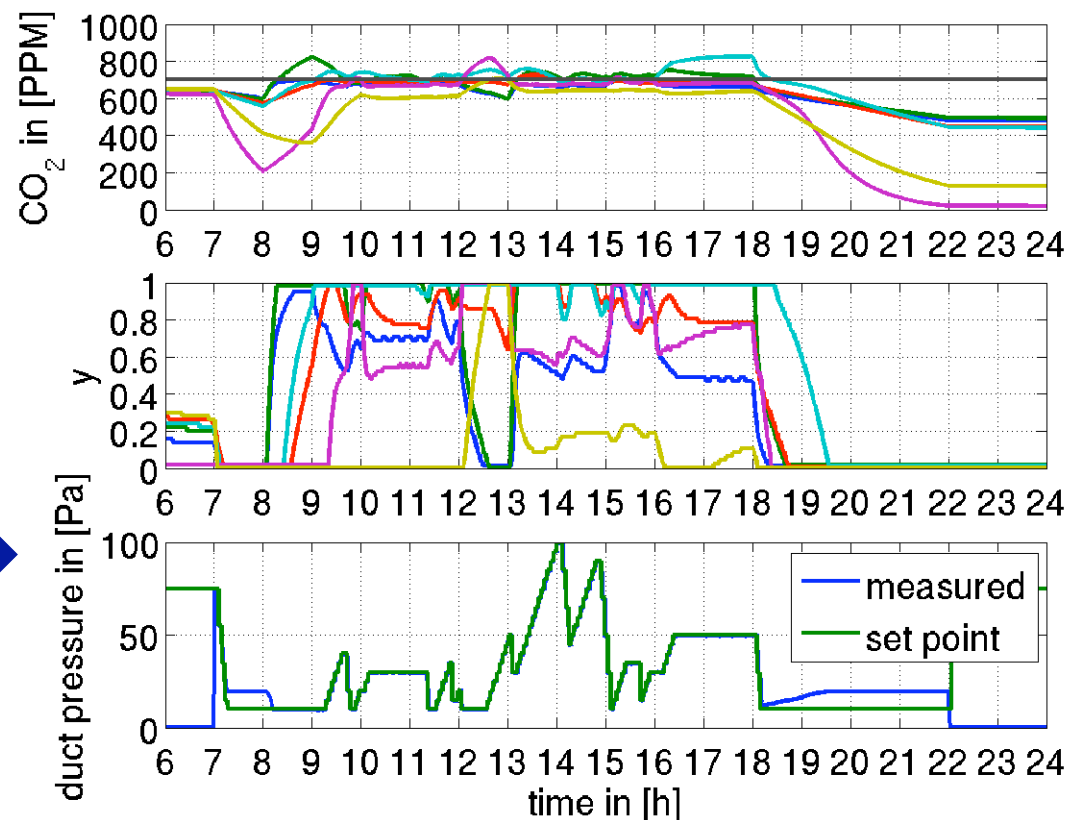
# Modeling for Controls Analysis

## Controls Pre-Commissioning for VAV Pressure Reset

VAV System  
(ASHRAE 825-RP)

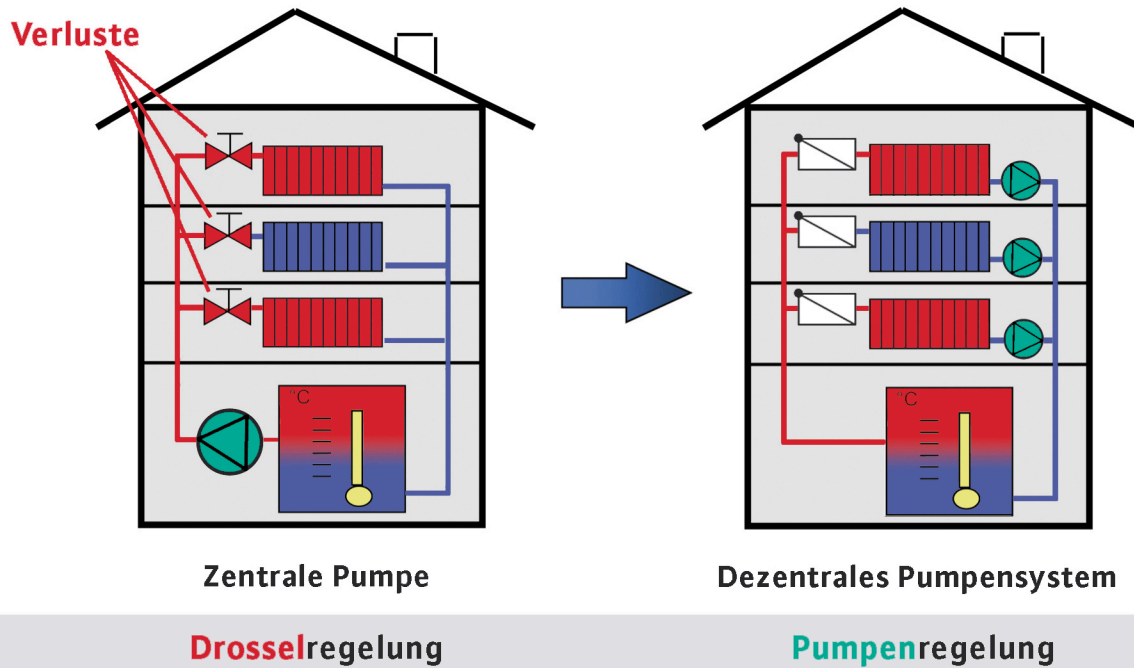


**GenOpt**  
Generic Optimization Program



# Rapid Prototyping: Wilo GENIAX (Introduction 2009)

## Systemidee: Von der Angebots- zur Bedarfsheizung



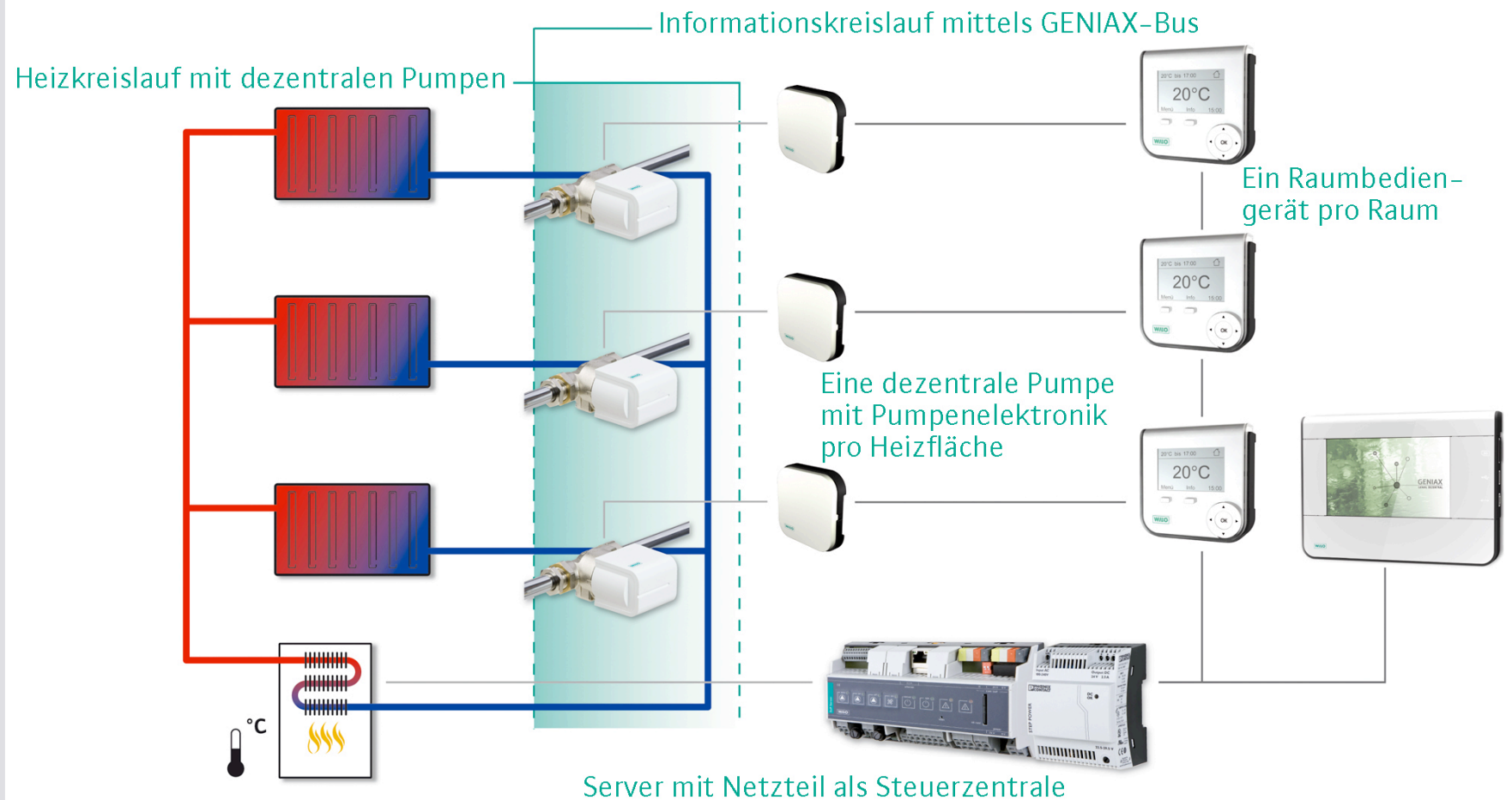
© WILO SE





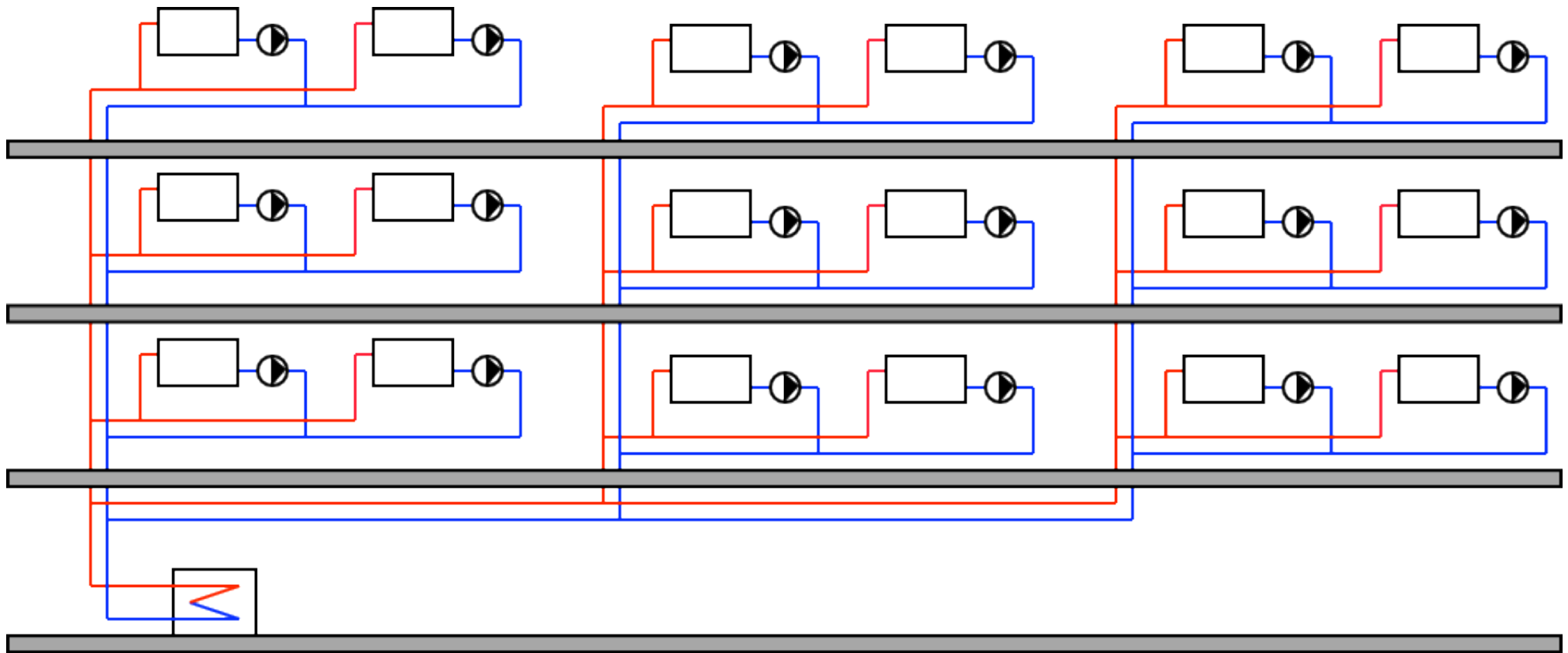
# Rapid Prototyping: Wilo GENIAX (Introduction 2009)

## Heiz- und Informationskreislauf des Dezentralen Pumpensystems GENIAX





# Rapid Prototyping: Wilo GENIAX (Introduction 2009)



## Original system model

2400 components

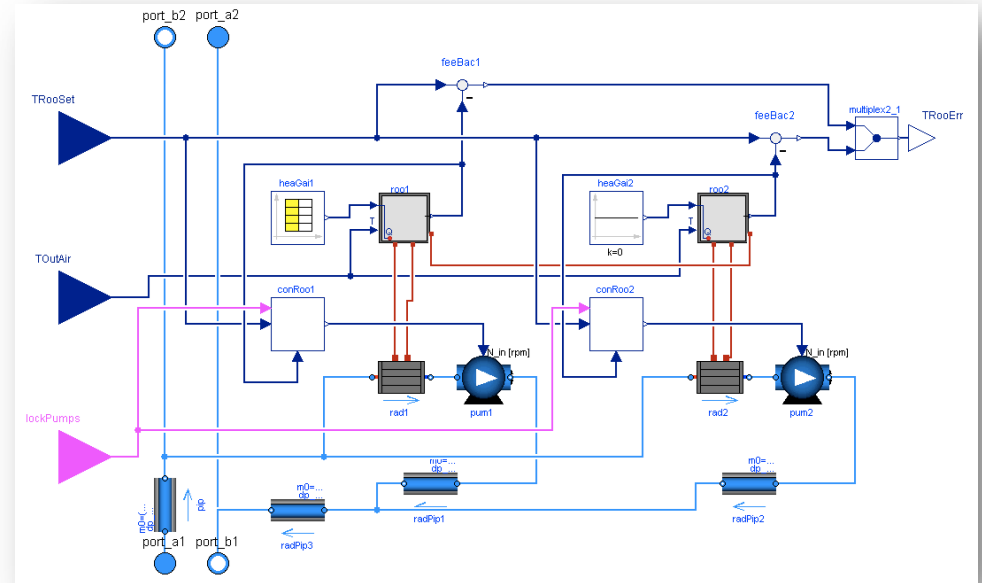
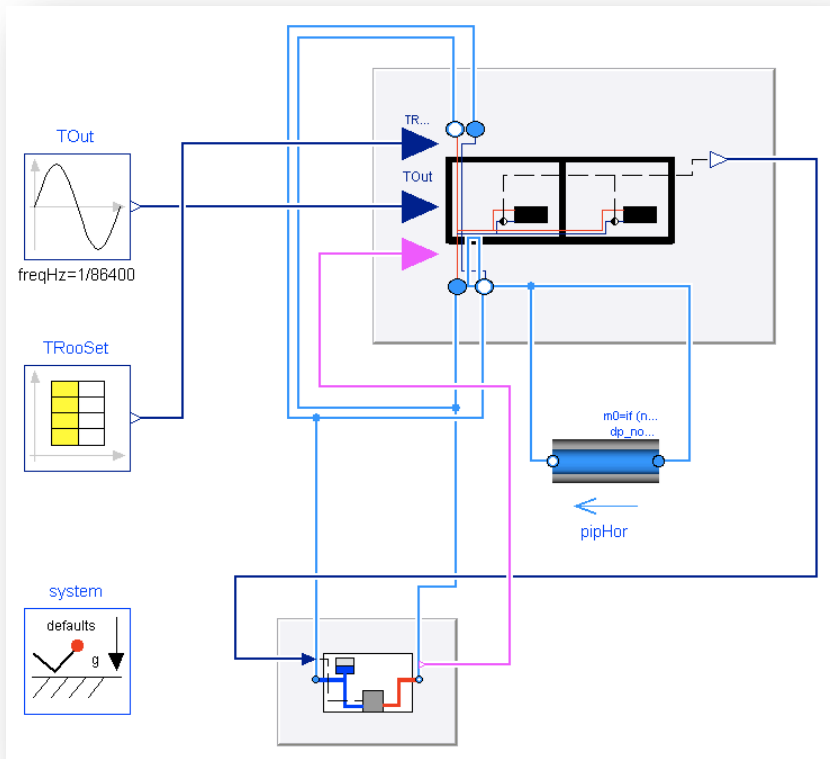
13,200 equations

## After symbolic manipulations

300 state variables

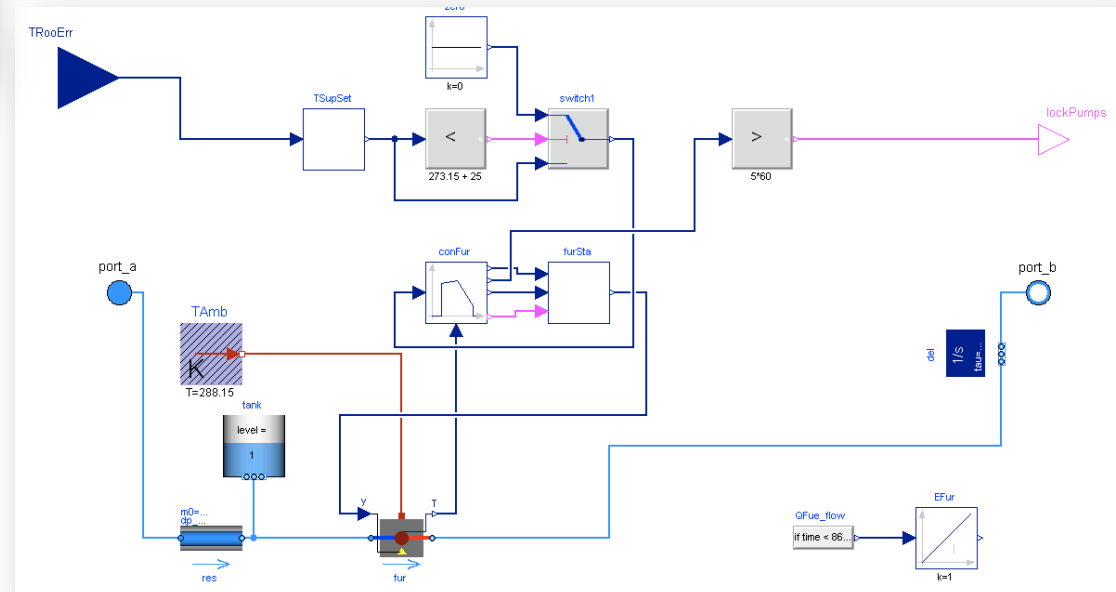
8,700 equations

# Rapid Prototyping: Wilo GENIAX (Introduction 2009)



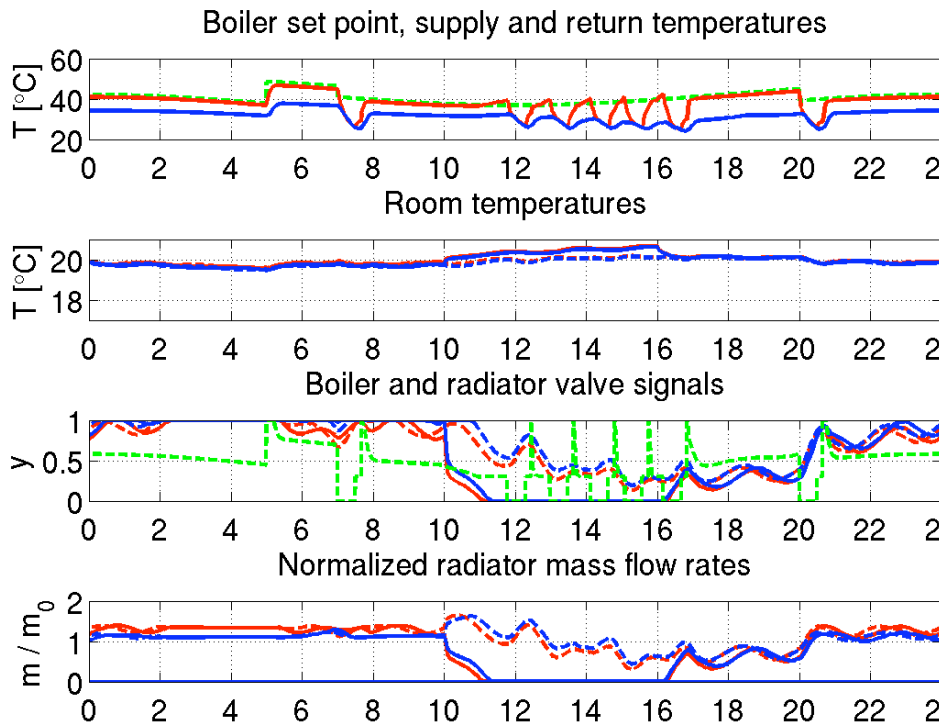
How would you design and control such a system optimally?

How do you stabilize multi-level controls?

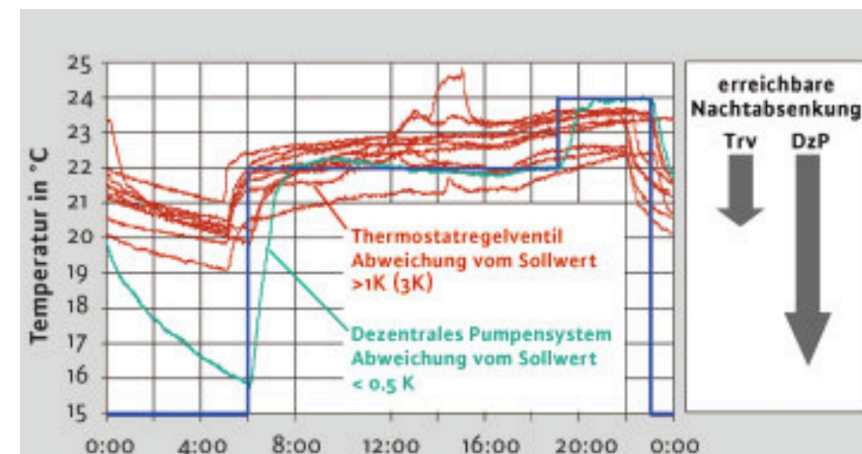
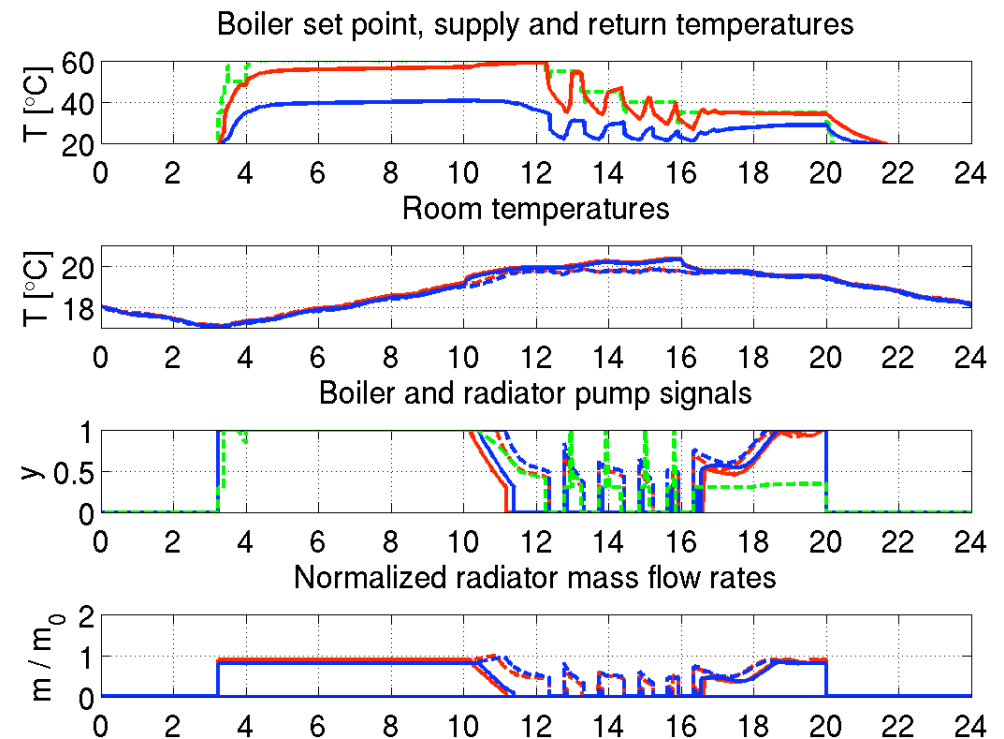


# Rapid Prototyping: Wilo GENIAX (Introduction 2009)

## Thermostatic radiator valves

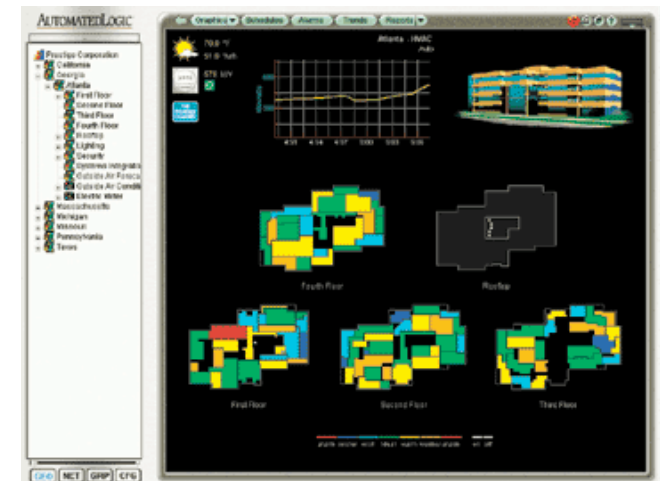
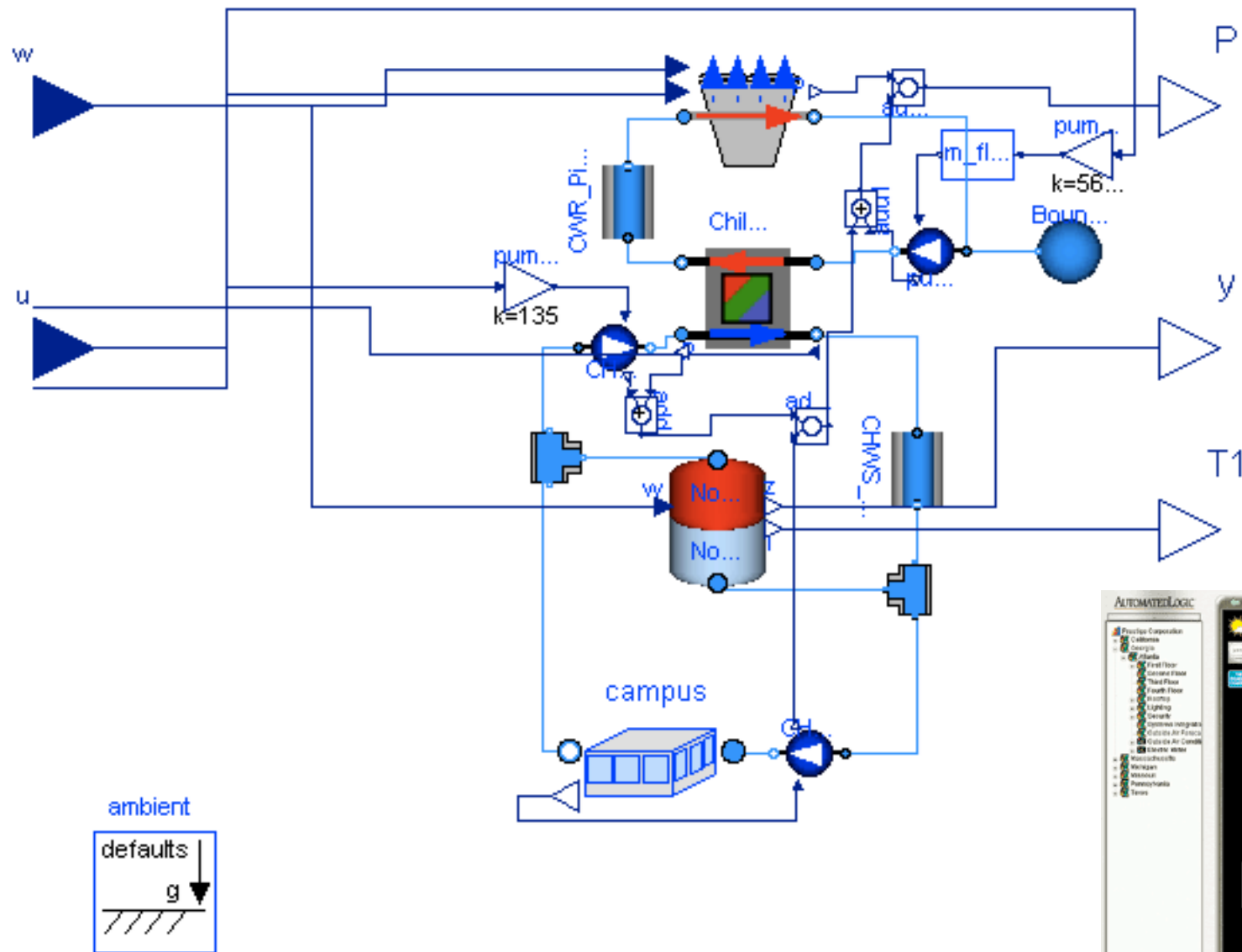


## Radiator pumps



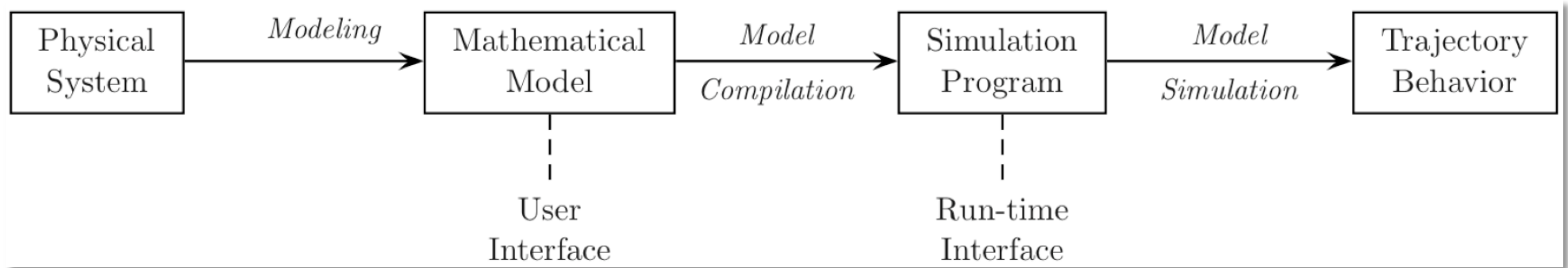
# UC Merced Chilled Water Plant Optimization

## Development and testing of Model Predictive Control Algorithms



# Need for co-simulation

## Modeling vs. simulation



Cellier & Kofman, 2006

However...

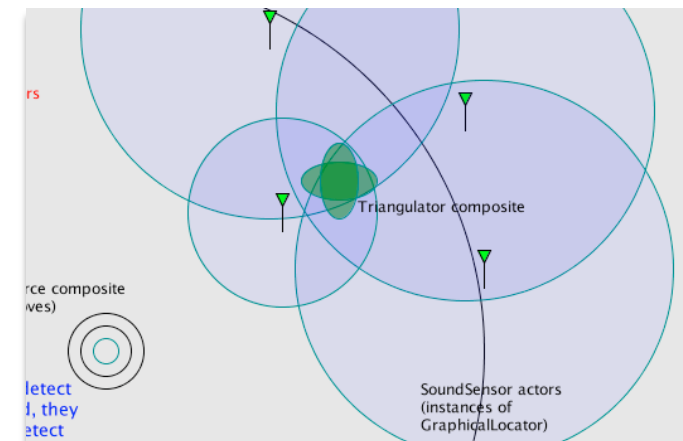
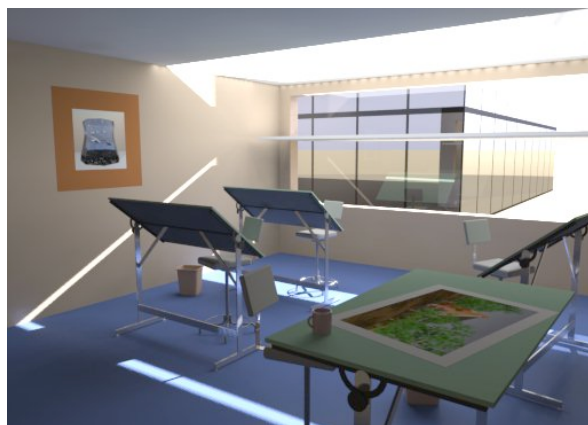
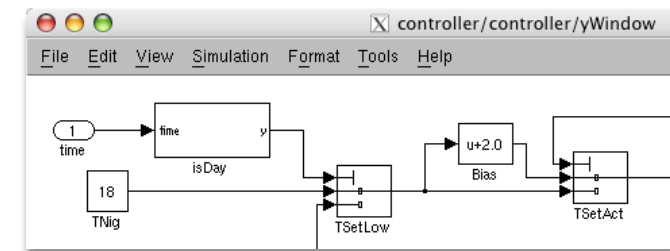
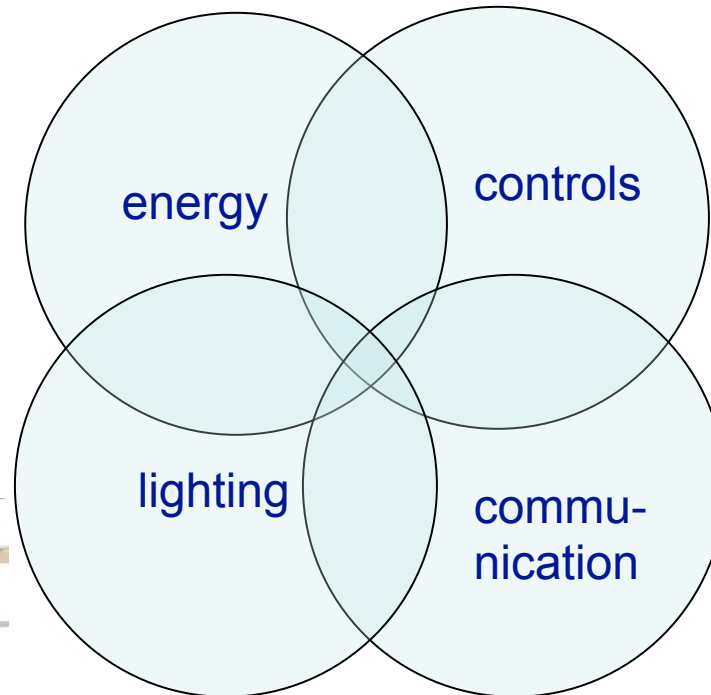
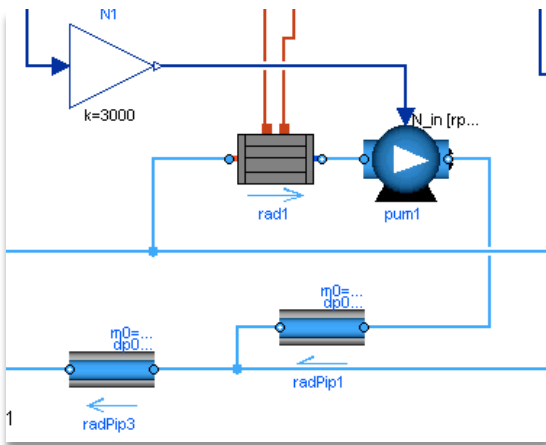
- There is still a need to use **legacy** simulation programs.
- How do we **link** new modeling approaches with legacy simulation programs using co-simulation?

Legacy simulation tools:

- 100+ man years development time.
- Some models have been extensively validated.

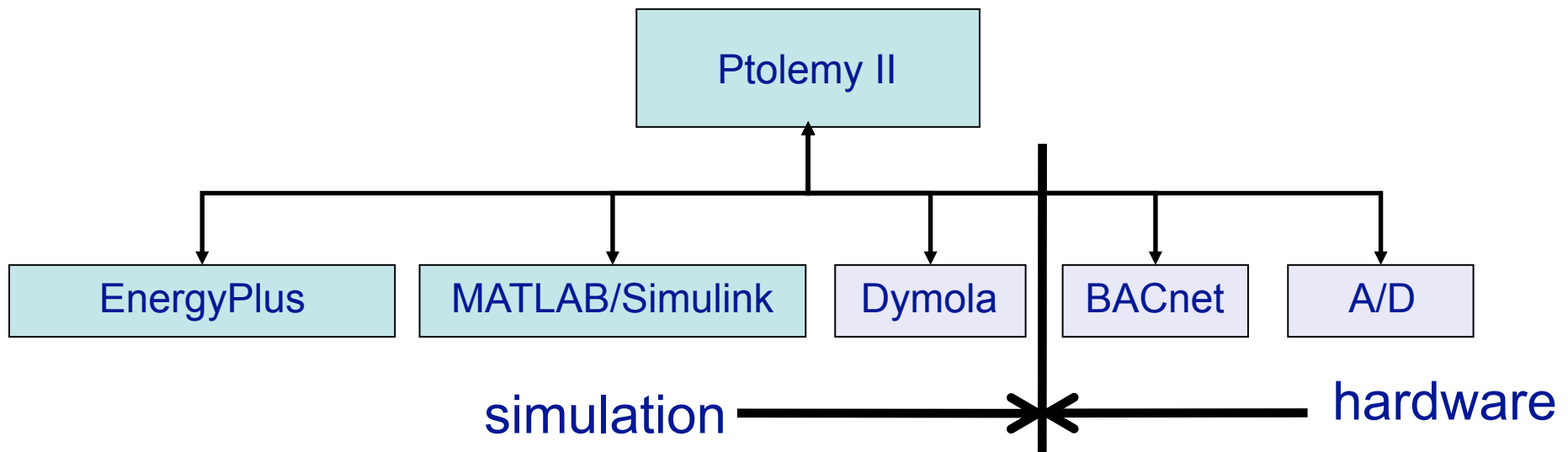
# Multi-Scale, Multi-Domain Computation

- Enable
- 1) Integrated multidisciplinary analysis
  - 2) Model-based system level design & optimization for design phase
  - 3) Model deployment during building operation to optimize performance



# Heterogeneous Systems

**Building Controls Virtual Test Bed:** Modular architecture built using Ptolemy II

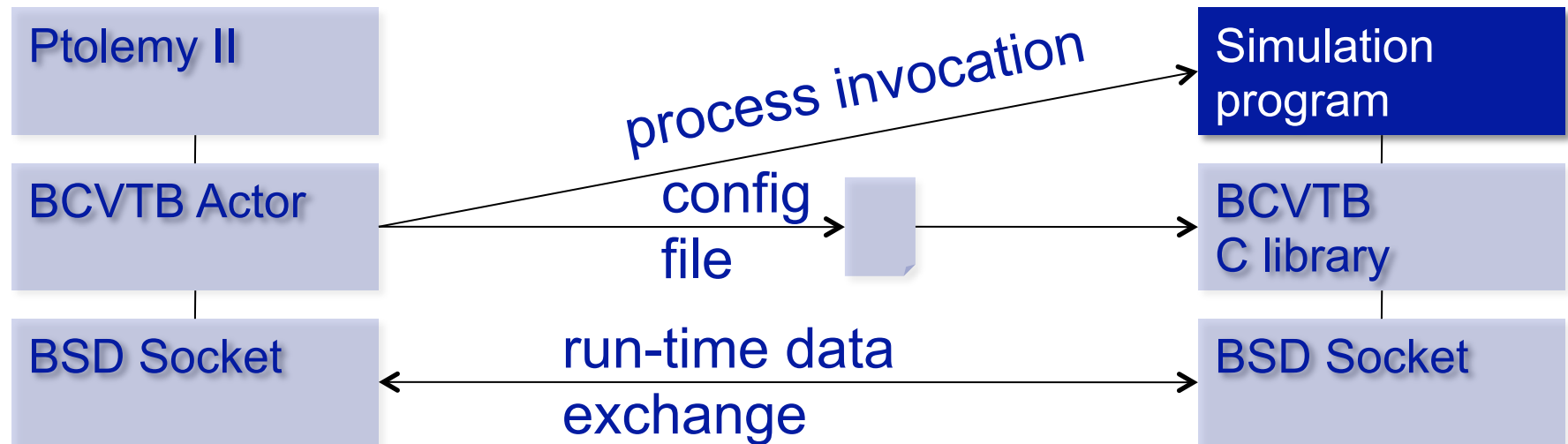


## Legend

 Implemented  
 Planned

Download  
<https://gaia.lbl.gov/bcvtb>

# BCVTB Architecture



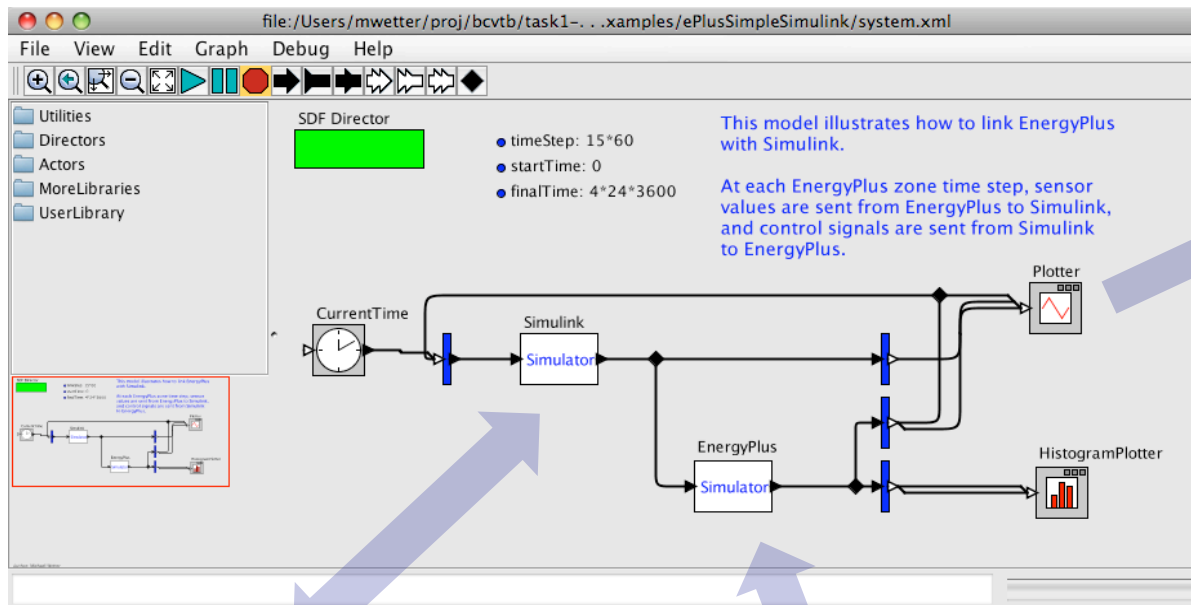
```
// Establish the client socket
const int sockfd = establishclientsocket("simulation.cfg");
if (sockfd < 0){
    fprintf(stderr,"Error: Failed to obtain socket file descriptor.\n");
    exit((sockfd)+100); }
// Simulation loop
while(1){
    // assign values to be exchanged
    for(i=0; i < nDblWri; i++)  dblValWri[i]=TRoo[i];
    // Exchange values
    const int retVal = exchangewithsocket(&sockfd, &flaWri, &flaRea,
        &nDblWri, &nIntWri, &nBooWri, &nDblRea, &nIntRea, &nBooRea,
        &simTimWri, dblValWri, intValWri, booValWri,
        &simTimRea, dblValRea, intValRea, booValRea);
    // Check flags
    if (retVal < 0){
        printf("Simulator received value %d when reading from socket. Exit simulation.\n", retVal);
        closeipc(&sockfd);  exit((retVal)+100); }
```



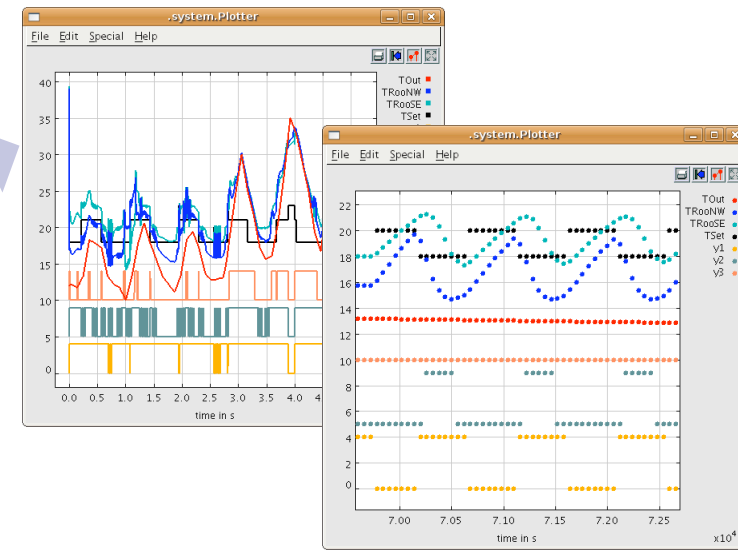
# Ex.: Natural Ventilation in SF Fed. Bldg.

All software modules reusable without code modification

Ptolemy II

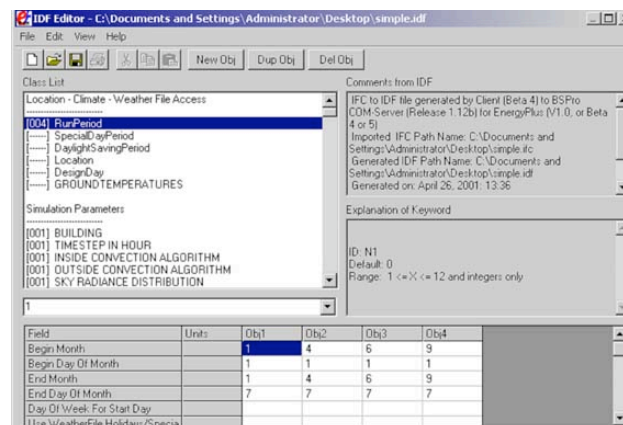
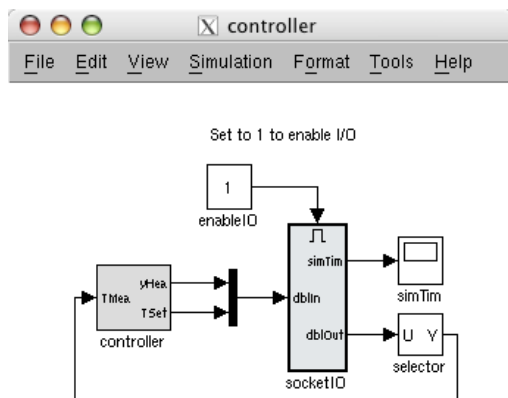


Real-time output



MATLAB/Simulink

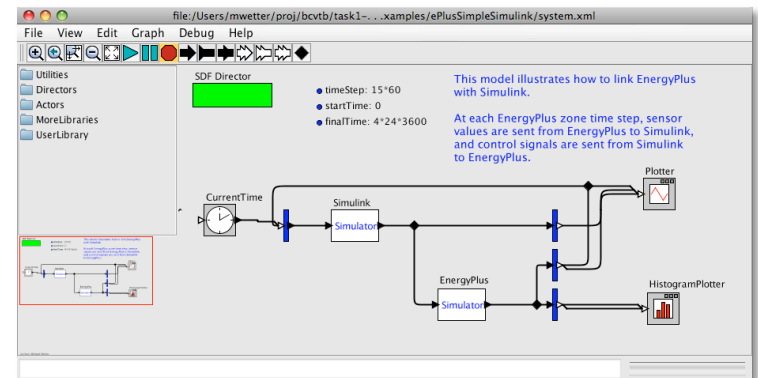
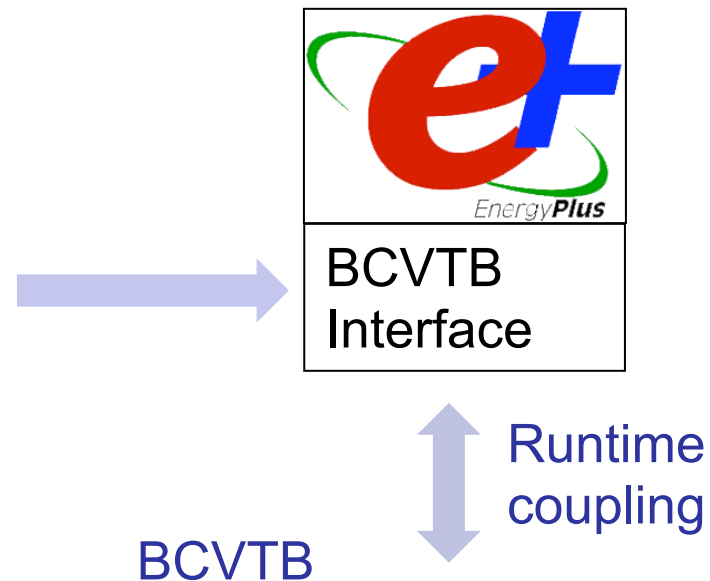
EnergyPlus



# Modeling of Buildings

## EnergyPlus linked to BCVTB

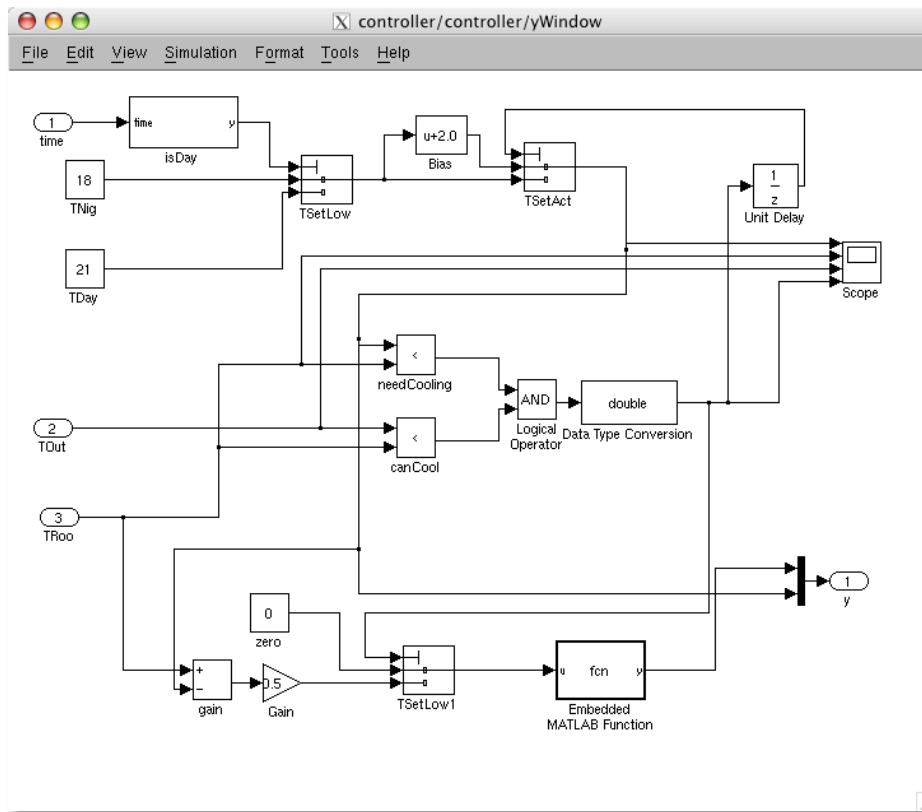
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<BCVTB-variables>
  <variable source="EnergyPlus">
    <EnergyPlus name="ZONE SOUTH"
      type="ZONE/SYS AIR TEMPERATURE"/>
  </variable>
  <variable source="Ptolemy">
    <EnergyPlus dayschedule="SP-TCooling"/>
  </variable>
  <variable source="Ptolemy">
    <EnergyPlus dayschedule="SP-THeating"/>
  </variable>
</BCVTB-variables>
```



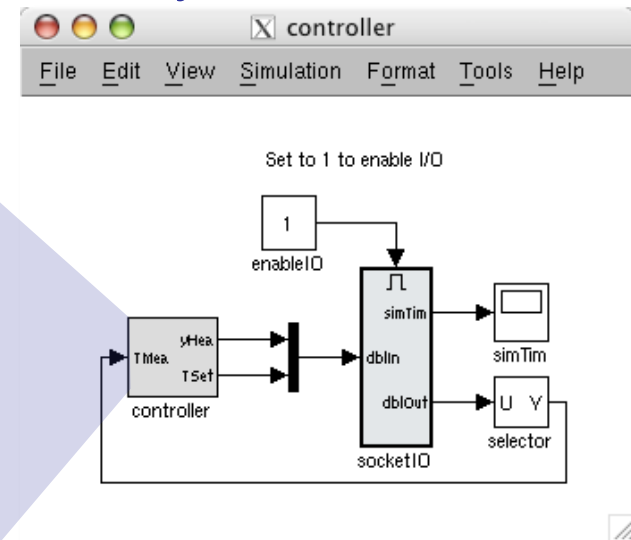
# Modeling of Controls

## MATLAB/Simulink-based controls development, linked to BCVTB

### Implementation of controls algorithm

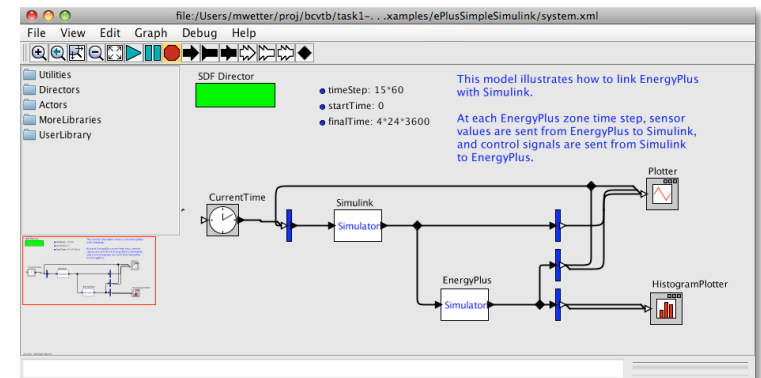


### Library with BSD socket interface



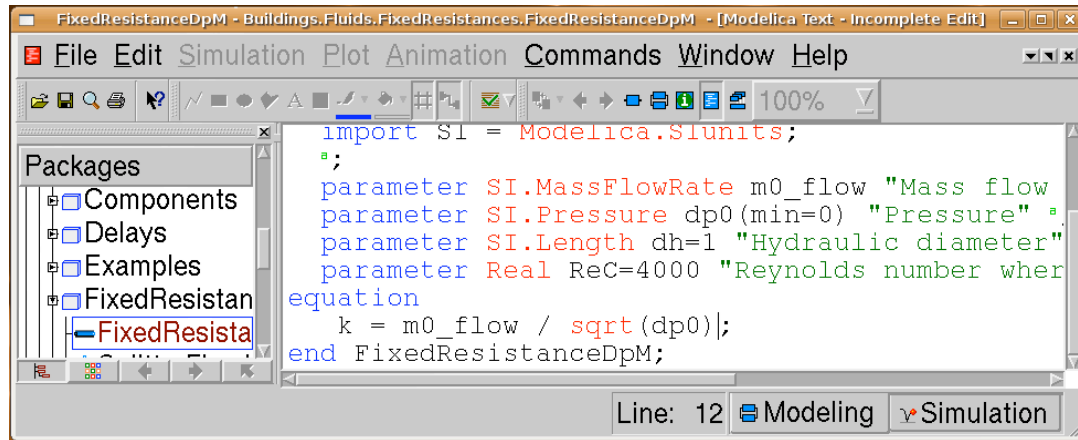
Runtime coupling

BCVTB

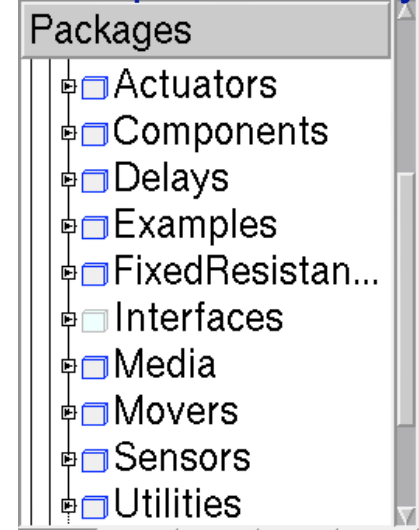


# Modeling of Mechanical System

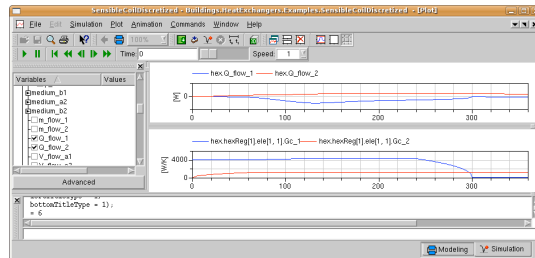
## Textual & graphical model editor



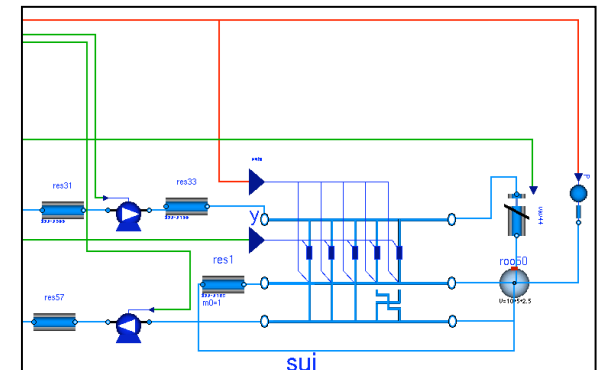
## Component library



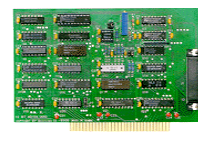
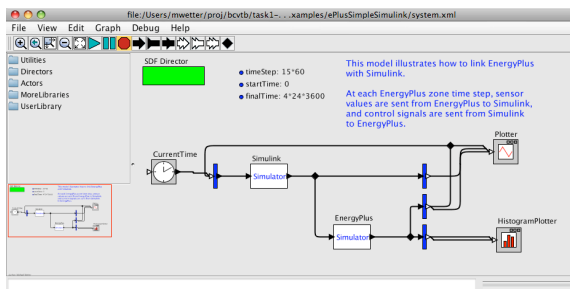
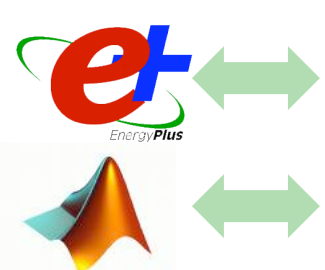
## Modelica simulation environment



## Graphical modeling environment

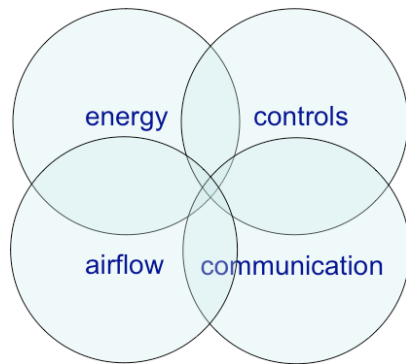


Runtime coupling

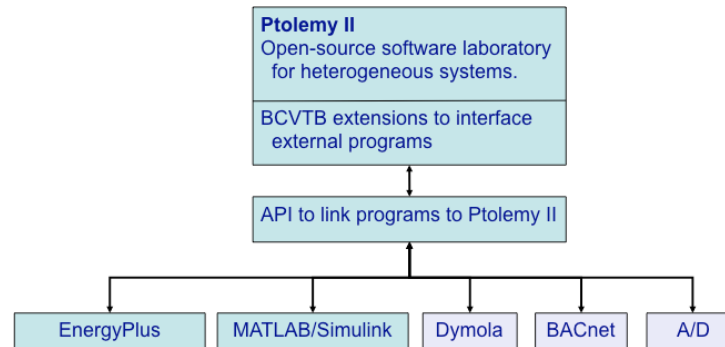


# Tool Needs to Support Innovation

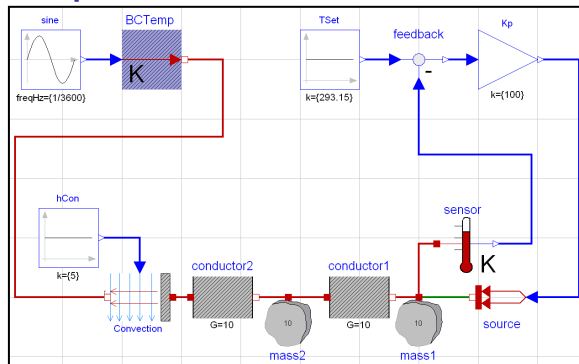
Better integration across disciplines



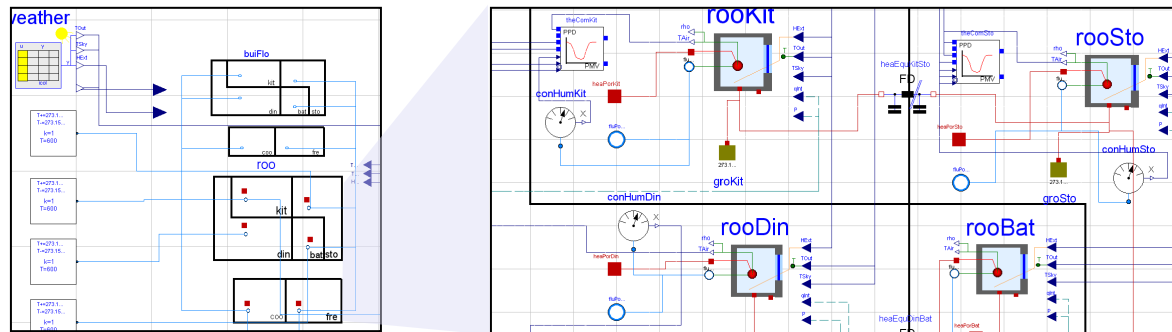
Integration of simulation & hardware



More natural system representation.  
Separation of concerns.



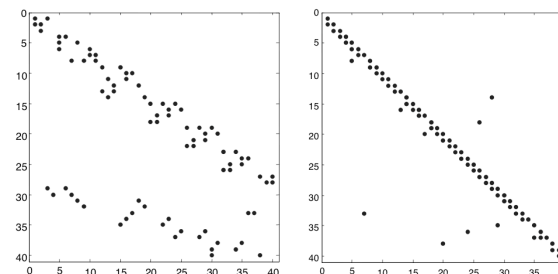
Hierarchical modeling to manage complexity



Standardized modelling language  
to share development effort



Computer algebra, modern solvers & parallelization



# Where could EECS help...?

Reduced order model extraction (from measurements and/or detailed simulation model), amendable for controls and retrofit decisions

Multi-level control algorithms (supervisory – local loop)

Robust simulation of nonlinear DAE systems (open-source Modelica environment)

Protocols for integrated building controls

Platform-based design for building systems

