Modelica-based simulation of building and district energy systems

25-27 August 2025

SESSION 1: Welcome & Introduction to system modeling

- Systems, models and simulation
- Building performance simulation tools
- Overview of Modelica

Alessandro Maccarini

SESSION 1: Welcome & Introduction to system modeling

- **Systems, models and simulation**
- Building performance simulation tools
- Overview of Modelica

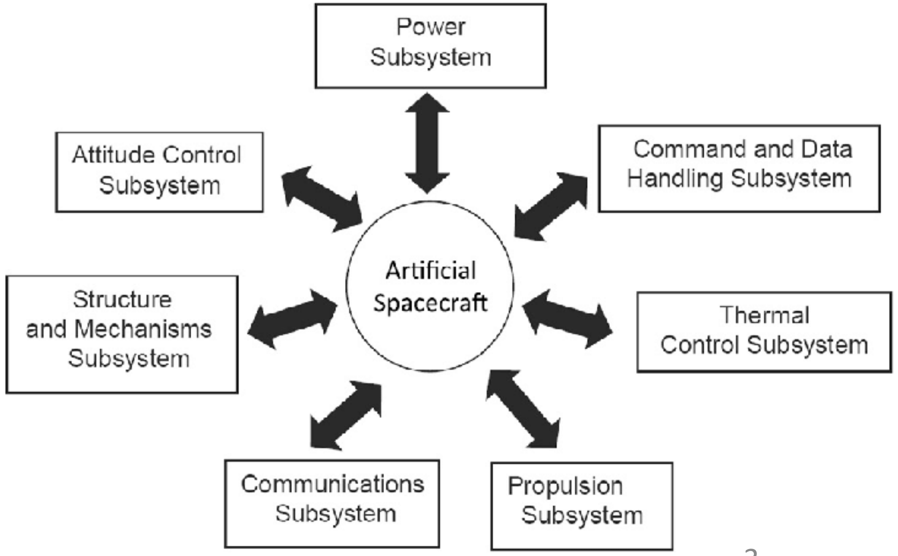
General concepts about systems

- **What is a system?**
  - A systems is an object (or collection of objects) whose properties are of interest
  - Spacecraft, tank, power system, etc.
  - A system can contain sub-systems, which are systems themselves and can contain components (e.g. spacecraft system contains the thermal control subsystem, which contain valves, pumps etc. as components)

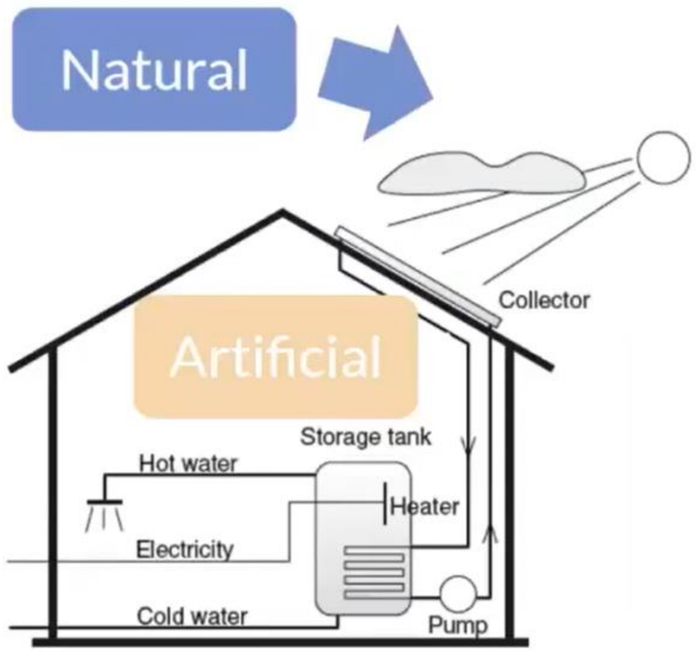
- **Why study a system?**
  - We want to study selected properties of these objects

    -Understand it, in order to build it:
    engineer's point of view

    -Satisfy human curiosity (understand more):
    researcher's point of view



System types and properties

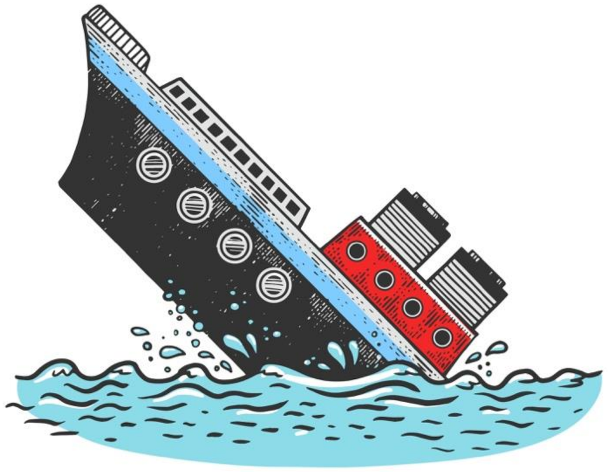


- **Natural and Artificial systems:**
  - A system can occur naturally (e.g. the universe) or artificially (e.g. spacecraft)
  - It can also be a mix of both: the solar-heated water system is artificial, but its performance are related to sun and clouds: natural

- **Properties:**
  - Observability: being able to take measurements of the system during the process
  - Controllability: system behavior can be changed by changing the system input
    - Inputs: variables of the environment that can influence the behavior of the system
    - Outputs: variables that are determined by the system and may influence the surrounding environment

Experiments

An experiment is the process of extracting information from a system by exercising its inputs



- **Challenges**
  - Too expensive: testing ship durability, how many ships we need to build?
  - Too dangerous: experiments in nuclear plant?
  - Not existing: new system not existing (yet)

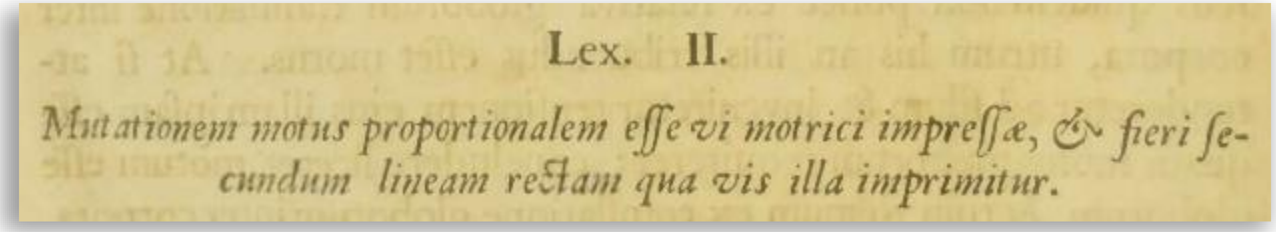
The challenges of experimentation lead us to the development of models!

Models

A model is a simplified representation of a system intended to promote understanding of the real system

- Physical model: physical object that mimic some properties of the real system (mock-up)

- Verbal model: expressed in words



Lex. II.

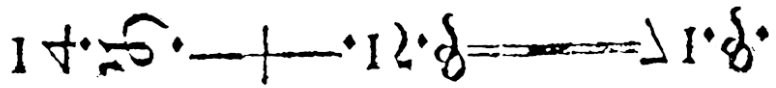
Mutationem motus proportionalem eſſe vi motrici impreſſæ, & fieri ſecundum lineam rectam qua vis illa imprimitur.

- Mathematical model: a description of the system where the relationships between variables are expressed in mathematical form

Mathematical models

- Mathematical models have been developed from physical principles (laws of nature) and/or experiments

- Mathematical models are based on equations
  - First evidence of mathematics from about 3000 BC
  - First equation in modern notation (equality sign) was introduced by Robert Recorde in 1557
  - Newton still wrote text in 1686!



- Type of equations
  - Differential equations contain time derivatives:

$$\frac{\partial x}{\partial t} = x + 3 - a$$

  - Algebraic equations do not include any differential variable:

$$x + b = 5$$

  - Partial differential equations also contain derivative with respect to other variables than time:

$$\frac{\partial x}{\partial t} = \frac{\partial^2 x}{\partial z}$$

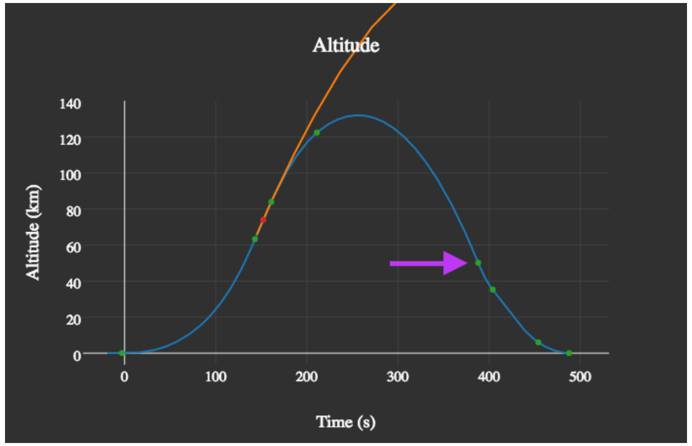
Solving equations is not always easy...

- Analytical methods
- Numerical methods

Computer programs

Simulations

A simulation is an experiment performed on a model



Real physical system

modeling

Computer-based mathematical model

simulation

System behavior

Dangers of models and simulation

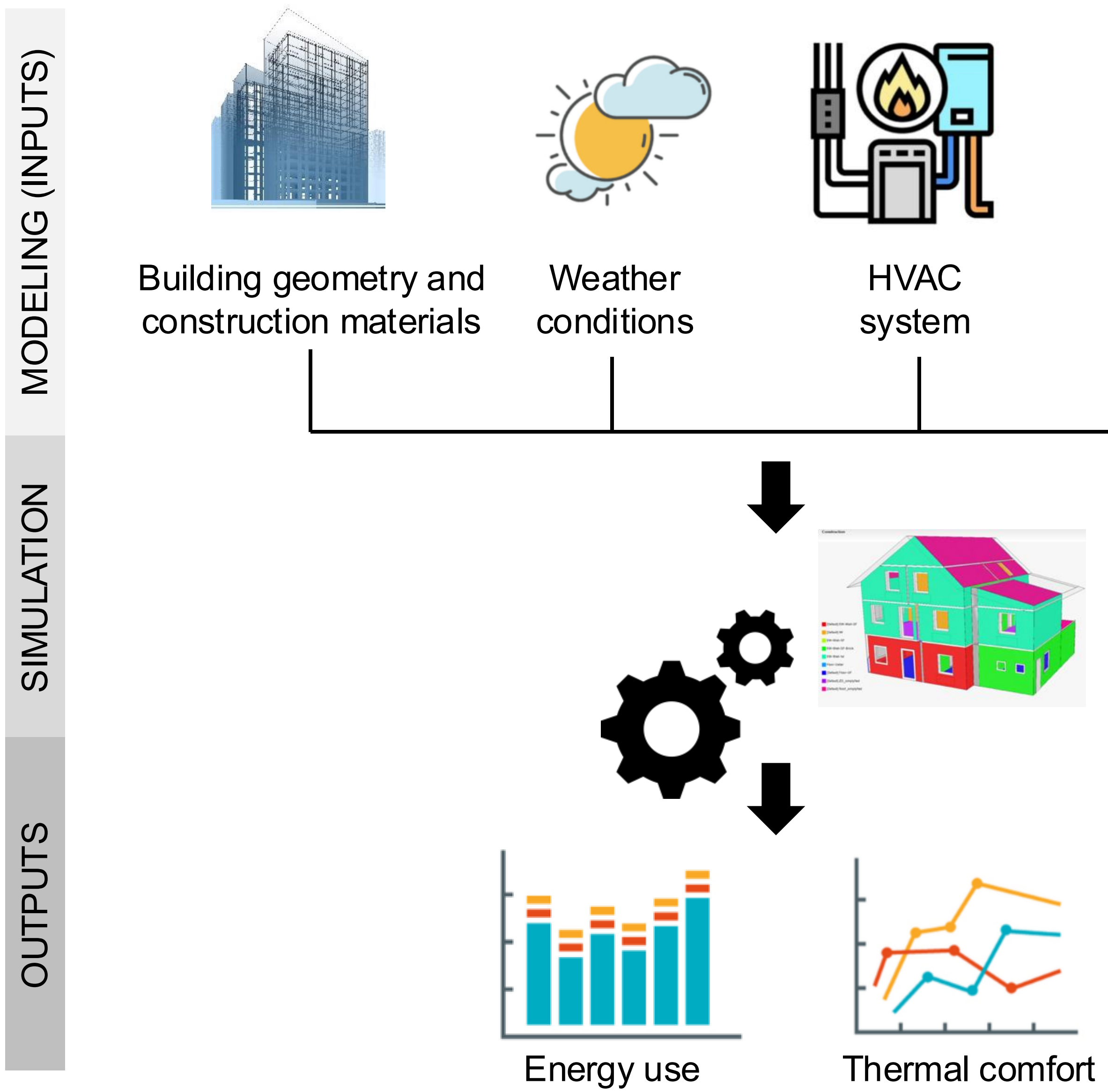
- **Falling in love with a model**
  - The Pygmalion effect: easy to become too enthusiastic about a model and forget that model is not the real world!
  - From the Greek myth of Pygmalion, a king who fell in love with one of his works, a sculpture of a young woman

- **Forgetting the model's level of accuracy**
  - All models have simplifying assumptions

SESSION 1: Welcome & Introduction to system modeling

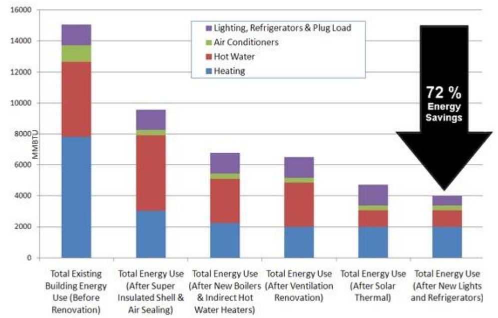
- Systems, models and simulation
- **Building Performance Simulation (BPS) tools**
- Overview of Modelica

What are BPS tools?



BPS tools can quantify aspects of building performance which are relevant to the design, construction, operation and control of buildings

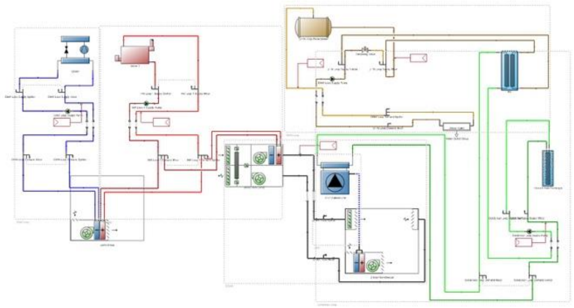
Why BPS tools?



Source: http://www.castledeepenergy.com/

- Predict energy savings associated to analysis of different design options (baseline case and improved cases)

- Check compliance with energy standards



- Identify opportunities for emerging technologies (design and developing new energy systems!)

- Development of digital twins and optimization algorithms

SESSION 1: Welcome & Introduction to system modeling

- Systems, models and simulation
- Building performance simulation tools
- **Overview of Modelica**

What is Modelica?

it is NOT a tool!

- Modelica is a free, equation-based modeling language with a textual definition to describe multi-domain physical systems

- In order that the Modelica modeling language can be used in practice we need:
  1. A graphical editor, to ”hide” the Modelica code behind a graphical representation of the model
  2. To translate the Modelica code in a form that can be simulated (C-code)
  3. To simulate the translated model with standard numerical integration methods

A **Modelica modeling and simulation environment** provides all of the functionalities above, in addition to auxiliary features (e.g. plotting)

The Modelica language and Modelica tools

Modelica

- Modelica Association (open, non-profit)
- Provides:
    - Modelica language
    - Modelica Standard Library (MSL)

OpenModelica

- Supported by Open Source Modelica Consortium (OSMC)
- Open-source software

Modelon Impact

- Developed by Modelon
- Commercial software

Dymola

- Developed by Dynasim AB (acquired by Dassault Systemes)
- Commercial software

These are just a few of the available tools. You can find the full list at: https://modelica.org/tools/

Key Modelica features

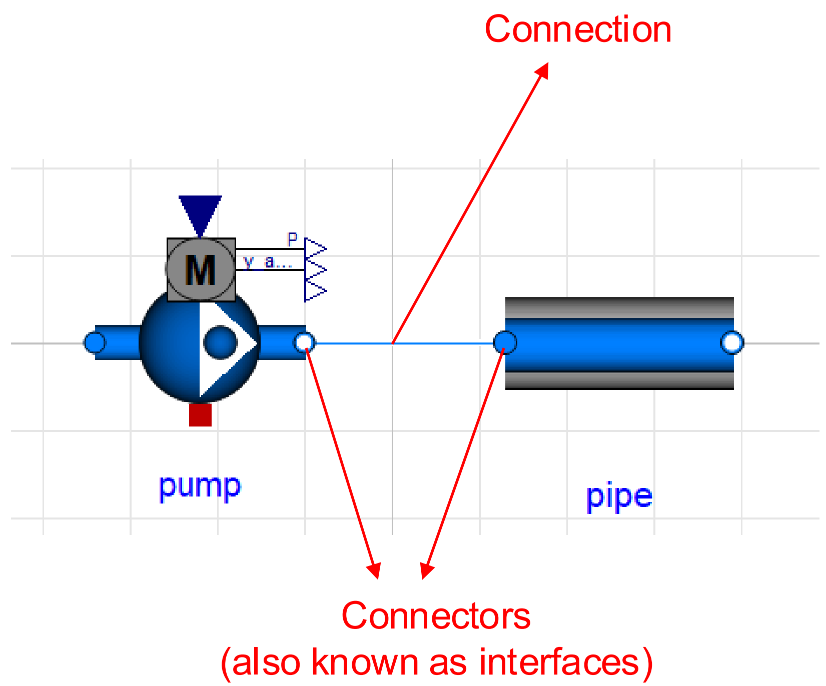
- Graphical modeling
- Equation-based
- Acasual
- Multi-domain

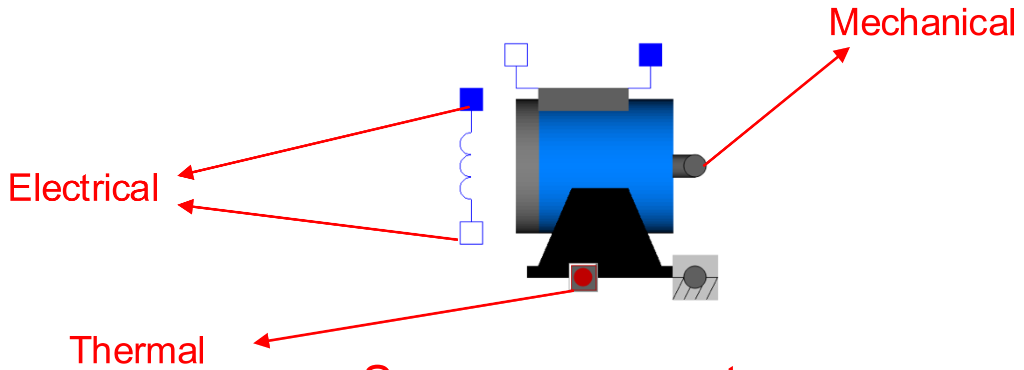
Modelica features (graphical modeling)

- Each icon represent a physical component (heat pump, wind turbine, etc.)

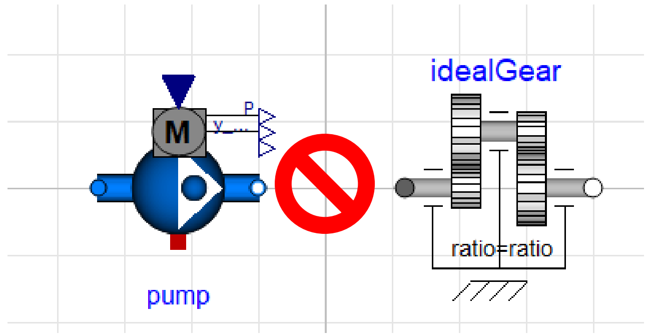
- Lines between components are connections,

- The connectors (or interfaces) defines the interfaces (or ports) i.e. physical properties at the boundaries of the model
  - Electrical: current, voltage
  - Mechanical: torque, angle
  - Fluid dynamic: flow rate, pressure



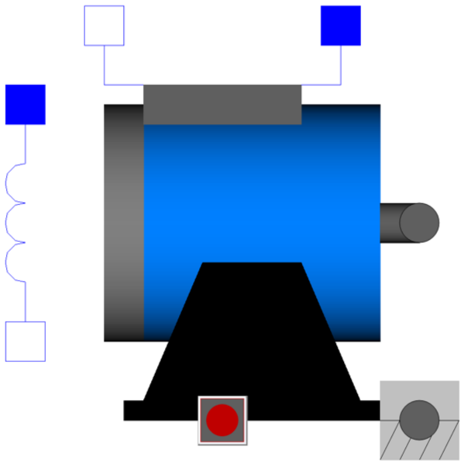


Some components cross between domains



Only components with the same type of interface can be connected!

Modelica features (equation-based)



Graphical view

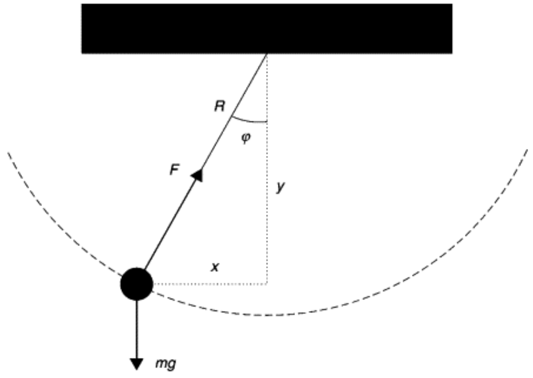
```
model DC_SeriesExcited "Series excited linear DC machine"
  extends Machines.Interfaces.PartialBasicDCMachine(
    wNominal(start=1410*2*pi/60),
    final ViNominal=VaNominal - (Machines.Thermal.convertResistance(
        Ra,
        TaRef,
        alpha20a,
        TaNominal) + Machines.Thermal.convertResistance(
        Re,
        TeRef,
        alpha20e,
        TeNominal))*IaNominal - Machines.Losses.DCMachines.brushVoltageDrop(
        brushParameters, IaNominal),
    final psi_eNominal=Lme*abs(IaNominal),
    redeclare final Machines.Thermal.DCMachines.ThermalAmbientDCSE
      thermalAmbient(final Tse=TeOperational),
    redeclare final Machines.Interfaces.DCMachines.ThermalPortDCSE thermalPort,
    redeclare final Machines.Interfaces.DCMachines.ThermalPortDCSE
      internalThermalPort,
    redeclare final Machines.Interfaces.DCMachines.PowerBalanceDCSE
      powerBalance(final powerSeriesExcitation=ve*ie, final
        lossPowerSeriesExcitation=re.LossPower),
    core(final w=airGapDC.w));

  parameter SI.Resistance Re(start=0.01)
    "Series excitation resistance at TeRef"
    ;
  parameter SI.Temperature TeRef(start=293.15)
    "Reference temperature of excitation resistance"
    ;
  parameter Machines.Thermal.LinearTemperatureCoefficient20 alpha20e(start=0)
    "Temperature coefficient of excitation resistance"
    ;
  parameter SI.Inductance Le(start=0.0005)
    "Total field excitation inductance"
    ;
  parameter Real sigmae(
    min=0,
    max=0.99,
    start=0) "Stray fraction of total excitation inductance"
    ;
  parameter SI.Temperature TeNominal(start=293.15)
    "Nominal series excitation temperature"
    ;
  parameter SI.Temperature TeOperational(start=293.15)
    "Operational series excitation temperature" ;
  output SI.Voltage ve=pin_ep.v - pin_en.v
    "Field excitation voltage";
  output SI.Current ie=pin_ep.i "Field excitation current";

protected
  final parameter SI.Inductance Lme=Le*(1 - sigmae)
    "Main part of excitation inductance";
  final parameter SI.Inductance Lesigma=Le*sigmae
    "Stray part of excitation inductance" ;
equation

end DC_SeriesExcited;
```

Text view
(Modelica code)

Which type of equations?

Physical systems can tpycally be represented by Differential Algebraic Equations (DAE), which are a combination of ordinary differential equations (ODE) and algebraic equations (AE)

- ODEs represent the governing physical equations
- AEs act as constrains



ODE (Newton's second law)

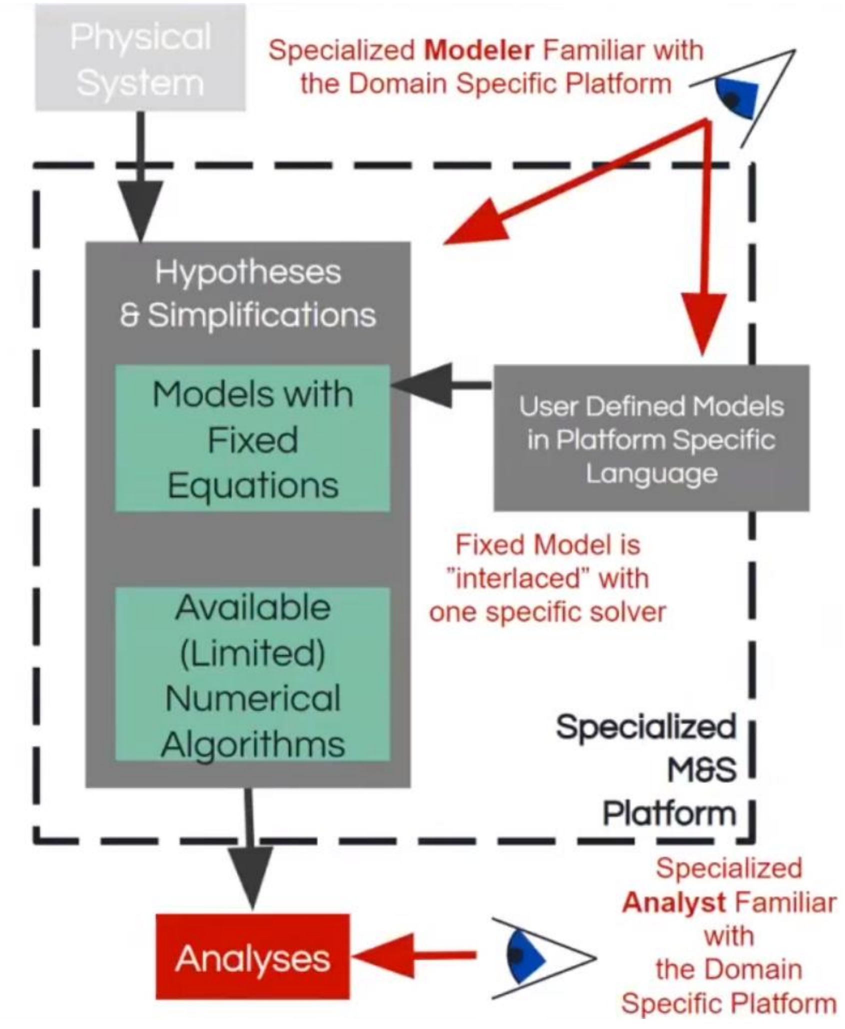
$$M\frac{d^2x}{dt^2} = -F\sin\varphi = -F\frac{x}{R}$$

$$M\frac{d^2y}{dt^2} = -F\cos\varphi - mg = -F\frac{y}{R} - mg$$

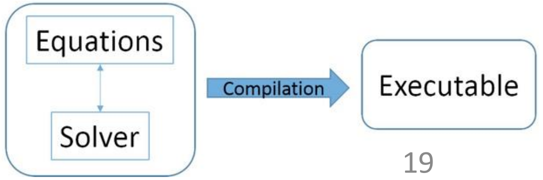
AE (geometric constraint)

$$x^2 + y^2 = R^2$$

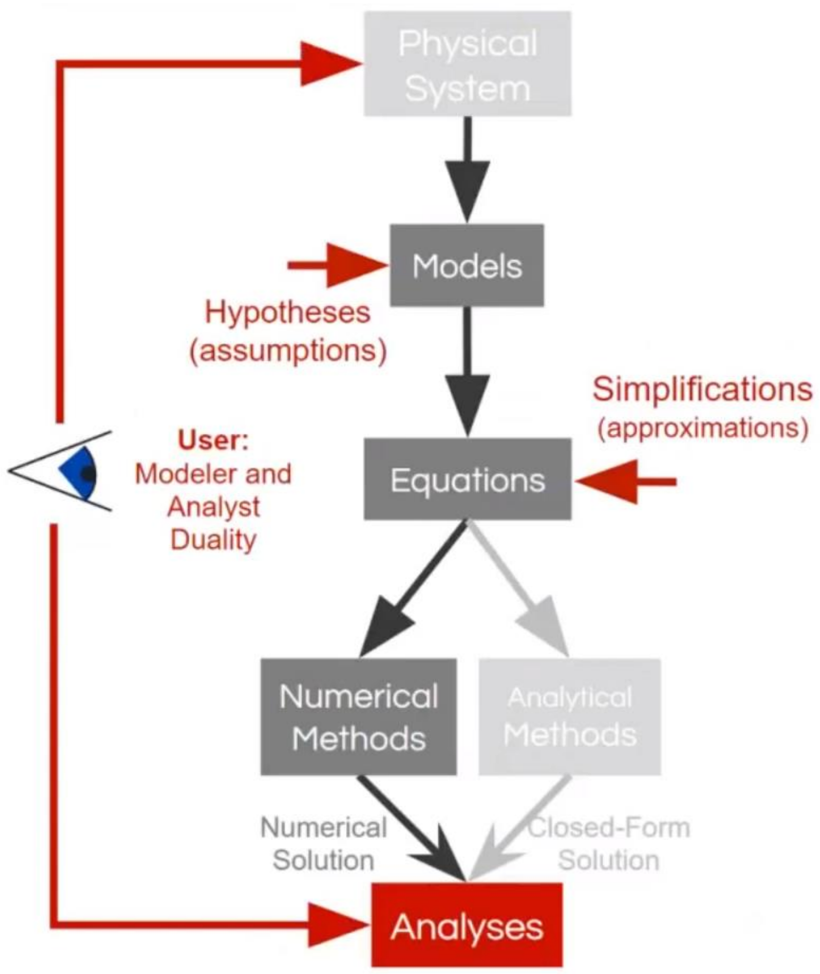
Traditional BPS tools



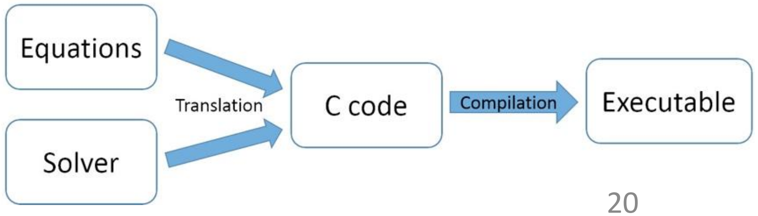
- Developed with imperative programming langauges(Fortran, C)
  - Clearly-defined sequence of instructions to a computer
  - Equations written as assignment statements
  - Example: R*i=v can be used in three ways:
    - i:=v/R
    - v:=R*I
    - R:=v/i
- Examples are EnergyPlus, ESP-r, DOE-2 etc
- We have a developer/modeler, who develops the software and the models, and we have user/analysis, who use the software
- Model equations are ”interlaced” with the solver
  - Difficult to make new models
  - Difficult to support new use cases



More “natural” approach



- Developed using equation-based modeling langauges
  - Equations written as equations!
  - Example: R*i=v can be written in this way no matter which are the variables/parameters
- Example is Modelica
- We have a single user, who is both a developer/modeler and an analyst
- Separation between model equations and solver: we can focus on model equations!

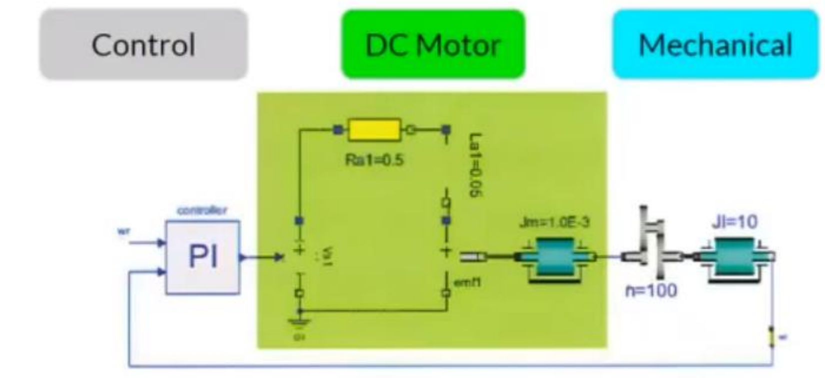


Modelica features (Acasual)

Acasual physical modeling (Modelica) vs. Block-oriented modeling (Simulink)

Acasual physical modeling (Modelica)

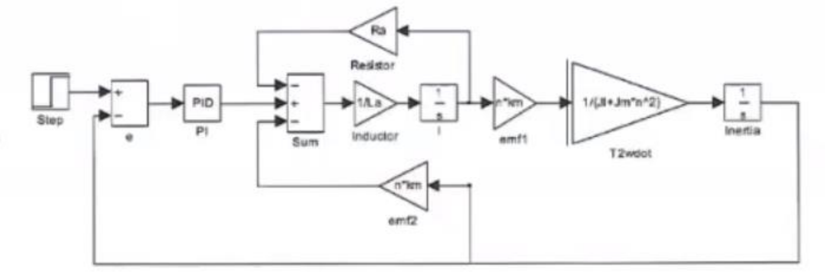
- The developer has to define the problem at a higher level and leaves the solution to the simulation tool



Acasual: Keep the physical structure, easy to understand

Block-oriented (Simulink)

- The developer has to define the order of calculations
- Loss of physical meaning

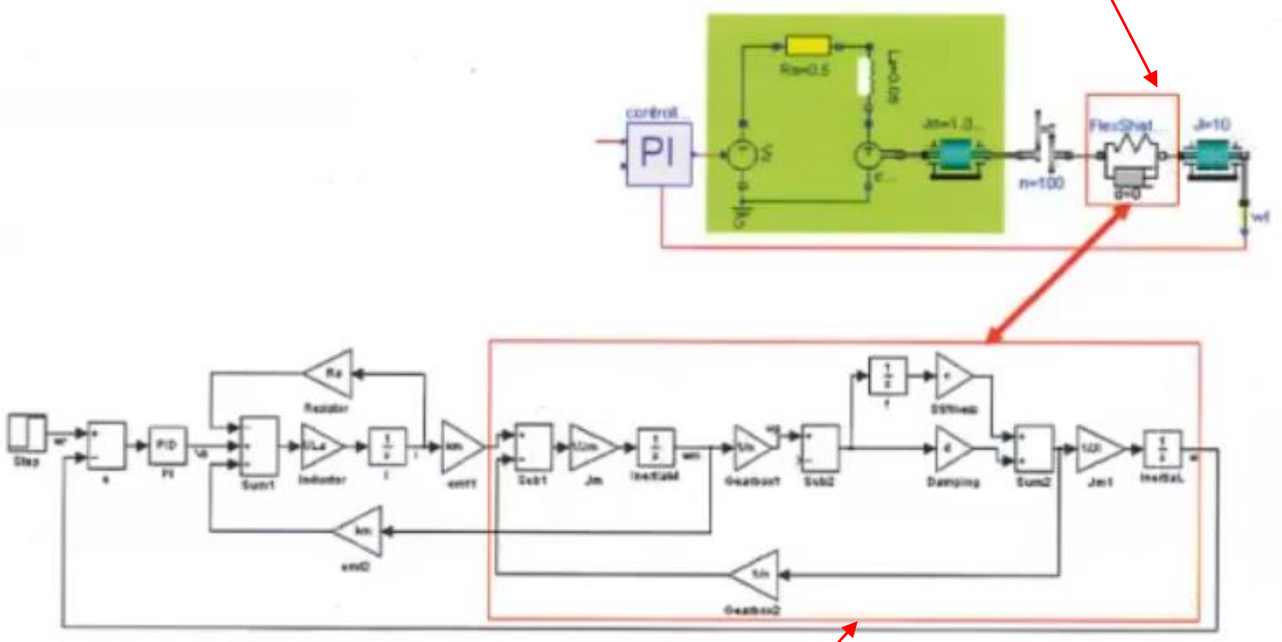


Block-oriented: signal-flow model, hard to understand

Modelica features (Acasual)

Problem: we want to study the vibration of the shaft

Acasual: only add one component between gear and load

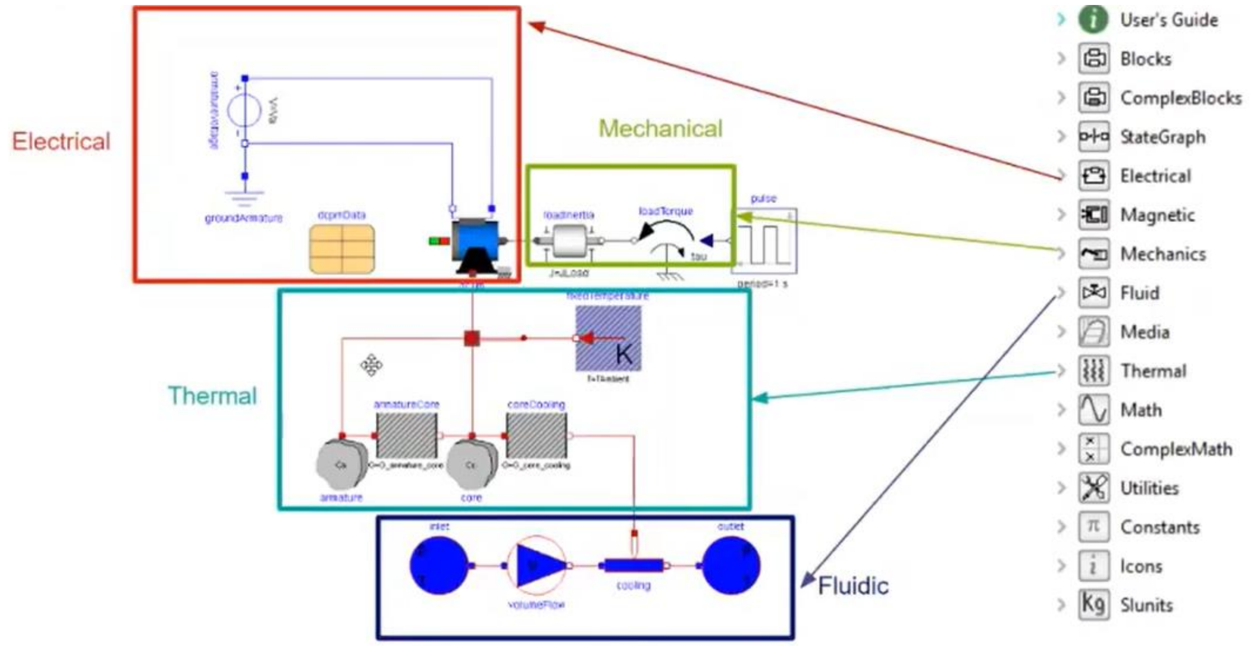


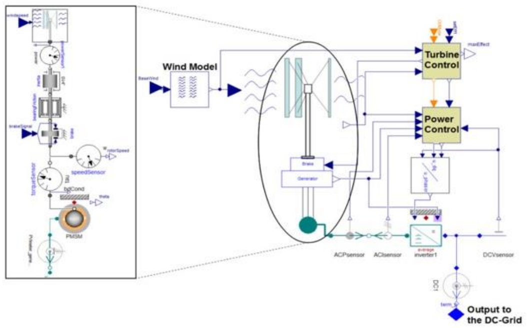
Simulink: needs to modify the entire signal flow

Not suitable for large-scale physical modeling!

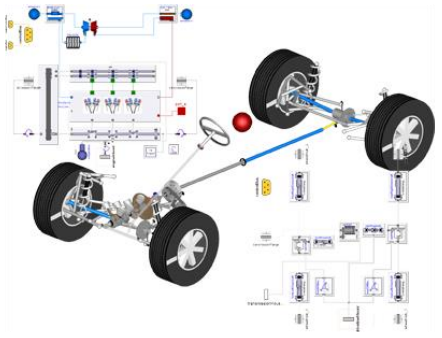
Modelica features (multi-domain / multi-engineering systems)

- Modelica allows to develop multi-domain models
- Modelica is used in several different engineering domains

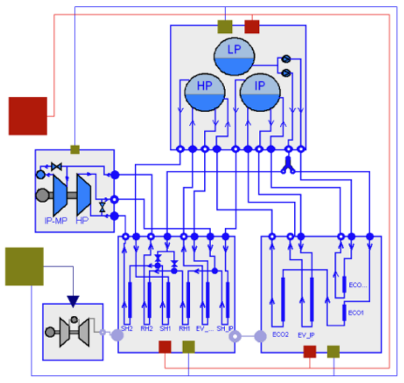




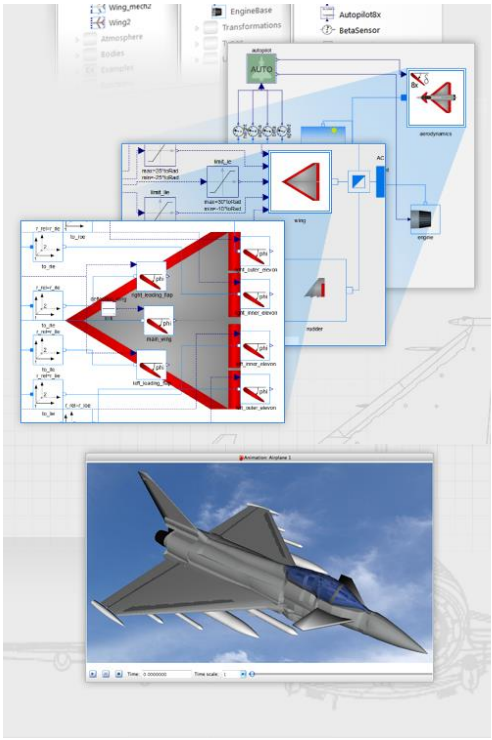
Wind turbine



Veichle



Power plant



Aircraft

References

Fritzson P. Object-oriented modeling and simulation with Modelica. Wiley-IEEE Press.

De Wilde, Pieter (2018). Building Performance Analysis. Chichester: Wiley-Blackwell.

Vanfretti Luigi. Online presentation CHEETA Webinar 1 - Introduction to Modeling and Simulation using the Modelica and FMI Standards