Challenges for Model-Based System Engineering

Past, Present and Future

www.xogeny.com

Background

Professional

- Studied Engineering, paid for school doing Software
- B.M.E., M.S. and Ph.D. all in Mechanical Engineering
- Lifelong fascination with modeling and simulation
- Worked in Powertrain Research at Ford for 10 years.
- V.P. at Emmeskay (acquired by LMS)
- WW Director of Marketing for PLM: Systems (Dassault Systèmes)
- Founder and President of Xogeny

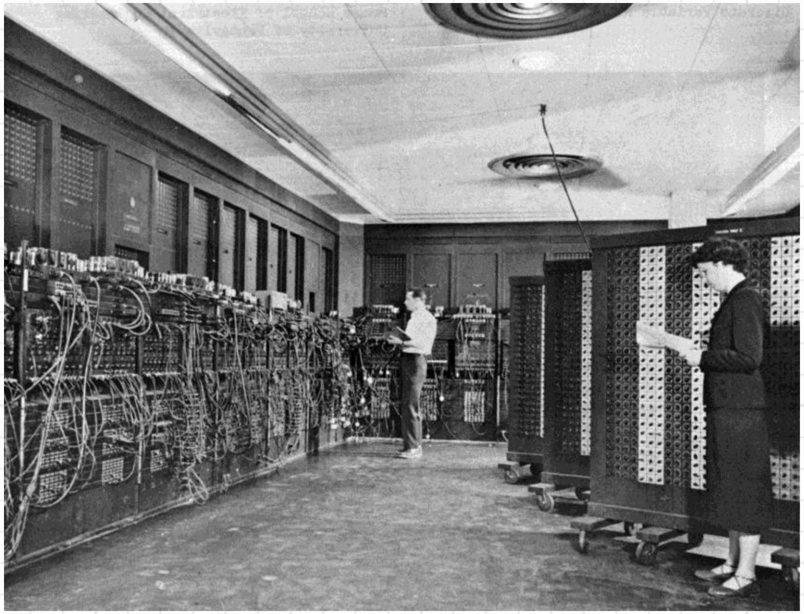
Modelica

- Author of the first book on Modelica, “Introduction to Physical Modeling with Modelica”
- Secretary of the Modelica Association
- Co-author of the Modelica specification
- President of the North American Modelica Users’ Group (NA-MUG)
- Working on a second book on Modelica (crowdfunded through Kickstarter) titled “Modelica By Example”

The Past

In the beginning…

» What were computers invented for?



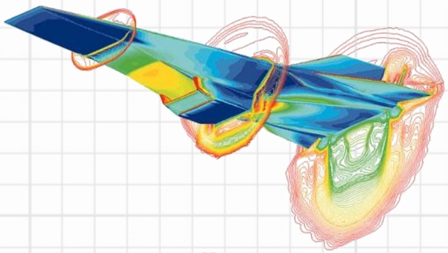
ENIAC (circa 1947-1955)

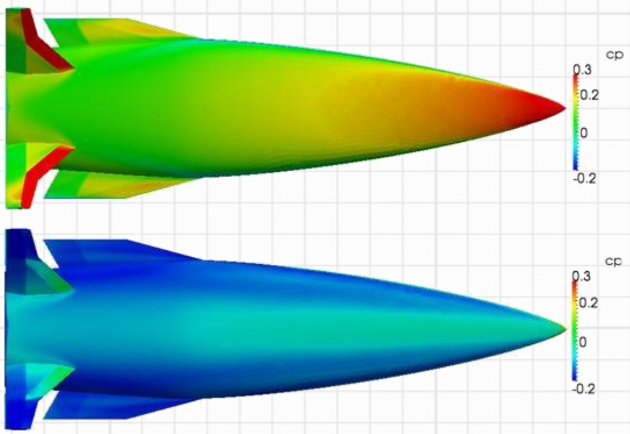
"The Giant Brain"

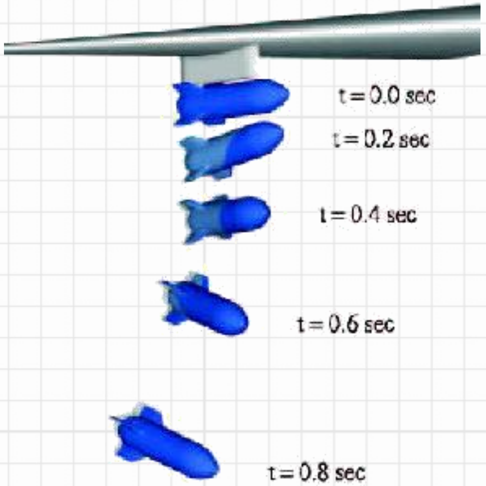
Artillery Firing Tables

» Simulation is as old as computing itself.

How did that work out?







» More computing power allowed greater and greater geometric detail.

Numerical Solvers

» At some point, people starting thinking about software architecture.

» Clever developers built solvers with a clean interface to "plug-in" different types of problems.

» "I give you a number, you give me a number"

» Most problems ended up in the general form:

$$\dot{x}(t) = f(x, u, t)$$

Software Pulls Ahead

» When I was an undergraduate:
  > I already knew C and C++
  > Engineering homework had to be done in Fortran
  > Engineering faculty were much more focused on the engineering side, not the software side
  > Where were engineering students going to learn about software?

» Engineering is an inherently conservative field.

» Strong emphasis is given to making sure you get the correct answer.
  > As it should be
  > But it doesn't need to be at the exclusion of everything else

DAE Algorithms

» 1978: H. Elmqvist, “A Structured Model Language for Large Continuous Systems”

» 1998: C.Pantelides, “The Consistent Initialization of Differential-Algebraic Systems”

» 1993: S.E. Mattson and G. Söderlind, “Index Reduction in Differential-Algebraic Equations Using Dummy Derivatives”

» CPU and memory were major constraints

The Present

Declarative

- In many aspects of computing, there is a shift away from “imperative” approaches toward declarative ones.
  - Imperative approaches obscure the underlying intent
- Focus on problem statement, less on solution methods.
- Declarative problems transcend solution methods.
- Focus is on relationships, not computations.

Modelica

- From the software world:
  - Object-Oriented (inheritance, encapsulation, interfaces)
  - Declarative
- From the engineering side
  - Mathematical representation
  - Causal and acausal relationships
- General Form: Hybrid DAE
  - Far more natural way to describe physical behavior
  - Continuous behavior
  - Discrete behavior (clock based as of Modelica 3.3)

Learning vs. Doing

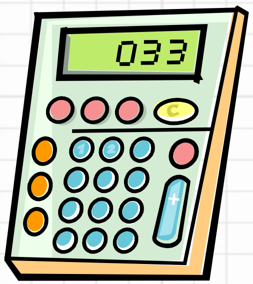
Block Diagrams

- Textbook equations have to be constantly reformulated depending on context.
- Different "blocks" with different combinations of inputs and outputs.
- Tedious, time-consuming and error prone.
- Long-division

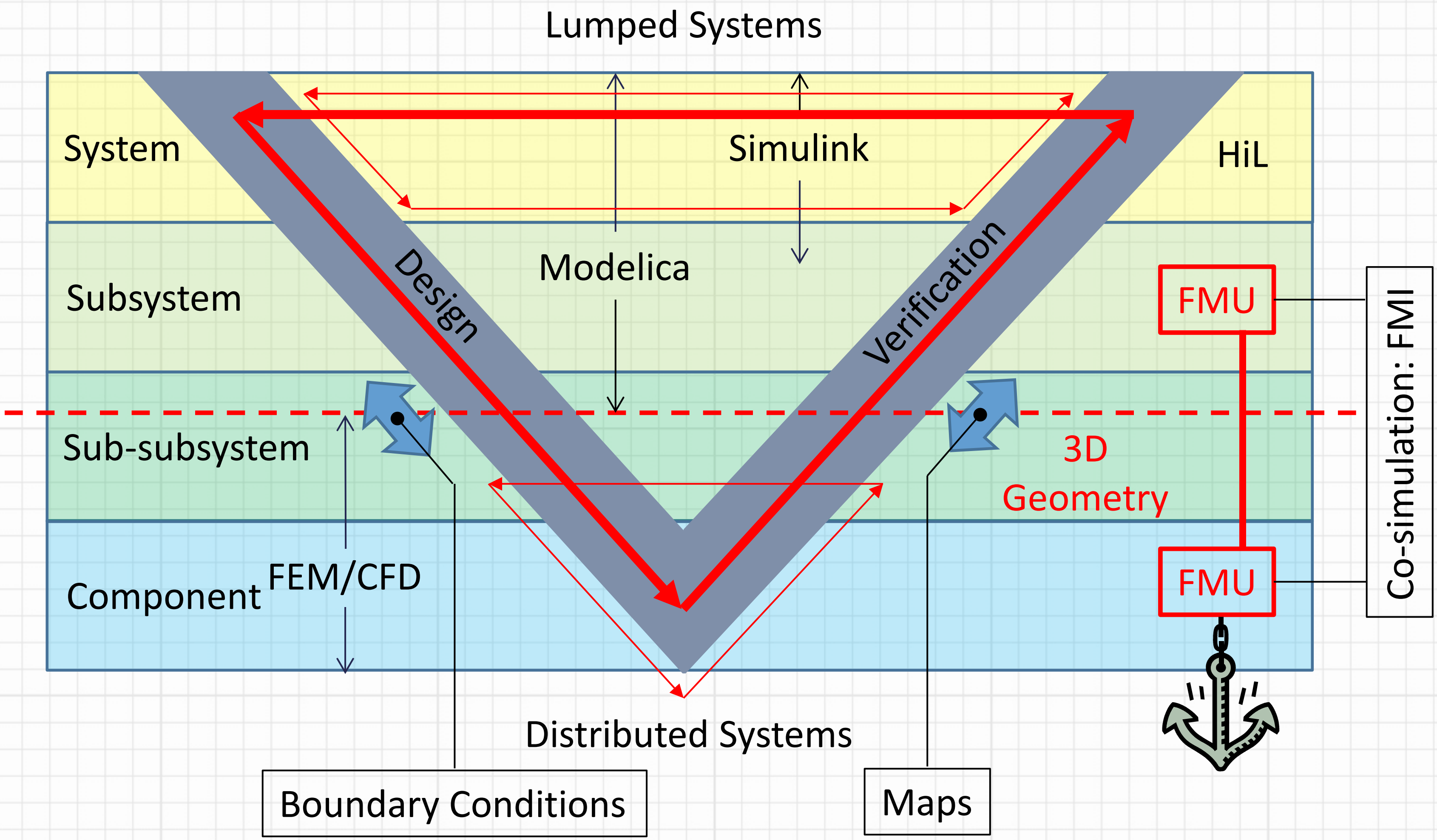
```
              0.1428
           7 )1.000000
1x7=7         - 7
                30
4x7=28        - 28
                20
2x7=14         - 14
                 60
8x7=56         - 56
```

Acausal Modeling

- Textbook equations are captured in reusable object-oriented component models.
- A single component for all causalities (e.g. planetary gear).
- Fun, fast and automated (and efficient!)
- Calculator

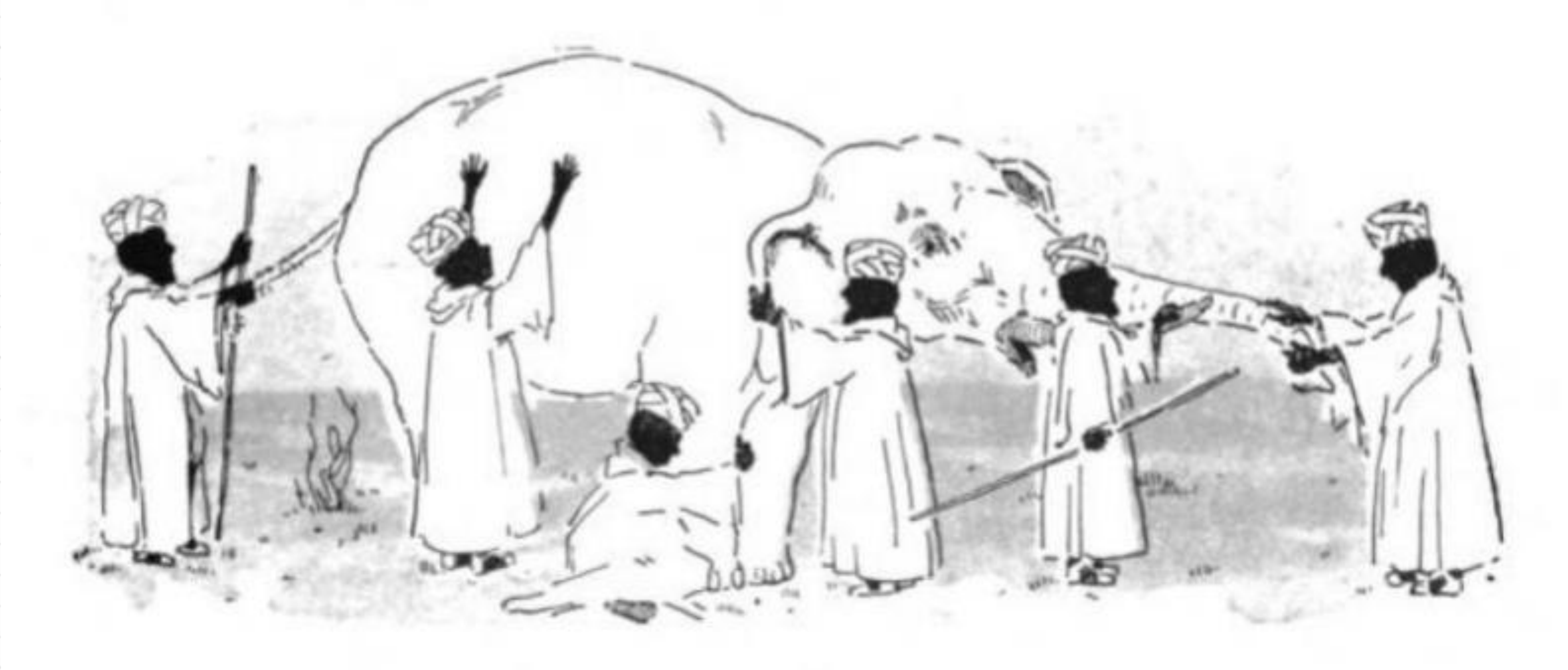


Modeling and "the V"

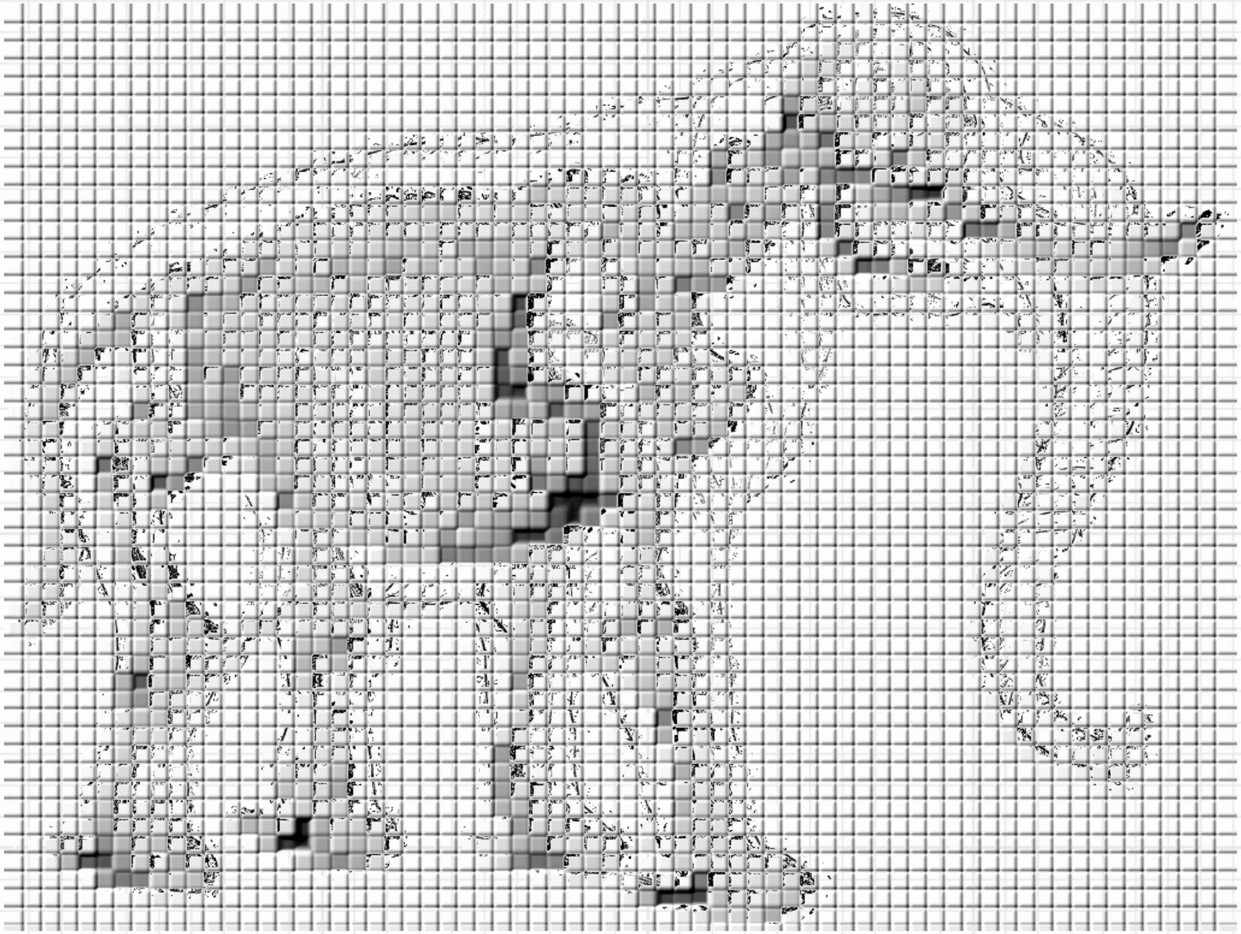


The Elephant

» The Six Blind Men and The Elephant



The Modeling Elephant



I give you the numbers, you give me MORE numbers.

Symbolic Manipulation

- Umbrella topic for:
  - Equation sorting
  - Index reduction
  - State selection
  - Substitutions
  - Tearing

  Requires structural information
- Goal is not a symbolic/analytical solution
- Reduces the DAEs down to ODEs
  - More natural way to express behavior
  - Reuse established numerical solvers
  - Heavily optimize evaluation costs
- Opinion: it will be impossible for purely numerical tools to compete.

Component Models

```
connector ElectricalPin
  import Modelica.SIunits.*;
  Voltage v;
  flow Current i;
end ElectricalPin;
```

```
partial model TwoPin
  import Modelica.SIunits.*;
  ElectricalPin p, n;
  Voltage v = p.v-n.v;
  Current i = p.i;
equation
  p.i + n.i = 0;
end TwoPin;
```

```
model Resistor
  extends TwoPin;
  import Modelica.SIunits.*;
  parameter Resistance R;
equation
  i*R = v;
end Resistor;
```

```
model Capacitor
  extends TwoPin;
  import Modelica.SIunits.*;
  parameter Capacitance C;
equation
  i = C*der(v);
end Capacitor;
```

```
model Inductor
  extends TwoPin;
  import Modelica.SIunits.*;
  parameter Inductance L;
equation
  der(i)*L = v;
end Inductor;
```

Generating Equations

$$step.n.v = resistor.n.v$$
$$resistor.n.v = inductor.n.v$$
$$inductor.n.v = capacitor.n.v$$
$$capacitor.n.v = ground.n.v$$

Across variables (equated)

$$step.n.i + resistor.n.i + inductor.n.i + capacitor.n.i + ground.n.i = 0$$

Through variables (summed)

$$step.p.v = resistor.p.v$$
$$resistor.p.v = inductor.p.v$$
$$inductor.p.v = capacitor.p.v$$
$$step.p.i + resistor.p.i + inductor.p.i + capacitor.p.i = 0$$

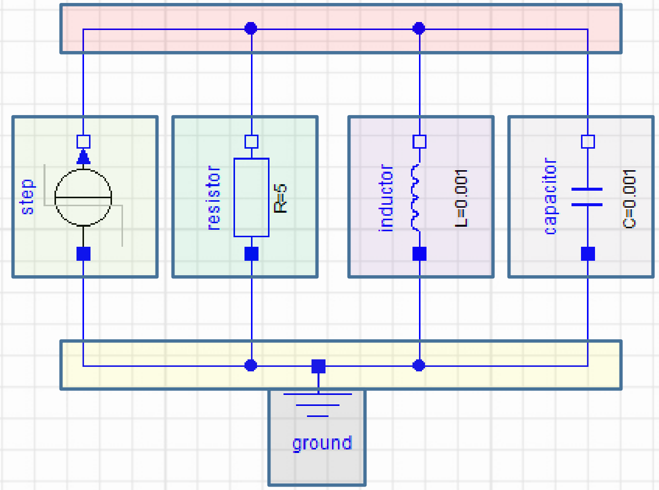
$$step.p.i + step.n.i = 0$$
$$step.p.i = f(t)$$

$$resistor.p.i + resistor.n.i = 0$$
$$resistor.p.i * resistor.R = resistor.p.v - resistor.n.v$$

$$inductor.p.i + inductor.n.i = 0$$
$$der(inductor.p.i) * inductor.L = inductor.p.v - industor.n.v$$

$$capacitor.p.i + capacitor.n.i = 0$$
$$capacitor.p.i = capacitor.C * [der(capacitor.p.v) - der(capacitor.n.v)]$$

$$ground.n.v = 0$$



Equation Structure

$step.n.v = resistor.n.v$

$resistor.n.v = inductor.n.v$

$\underline{inductor.n.v} \Leftarrow \underline{capacitor.n.v}$

$\underline{capacitor.n.v} \Leftarrow \underline{ground.n.v}$

$step.n.i + resistor.n.i + inductor.n.i + capacitor.n.i + ground.n.i = 0$

$step.p.v = resistor.p.v$

$resistor.p.v = inductor.p.v$

$inductor.p.v = capacitor.p.v$

$step.p.i + resistor.p.i + inductor.p.i + capacitor.p.i = 0$

$step.p.i + step.n.i = 0$

$step.p.i = f(t)$

$resistor.p.i + resistor.n.i = 0$

$resistor.p.i * resistor.R = resistor.p.v - resistor.n.v$

$inductor.p.i + inductor.n.i = 0$

$der(inductor.p.i) * inductor.L = inductor.p.v - industor.n.v$

$capacitor.p.i + capacitor.n.i = 0$

$capacitor.p.i * capacitor.C = der(capacitor.p.v) - der(capacitor.n.v)$

$\underline{ground.n.v} \Leftarrow \underline{0}$

Structure of Equations

$$\left[\begin{array}{cccccccccccccccccc}
&&& -1 & 1 &&&&& &&&&& &&& \\
&& -1 & 1 &&&&& &&&&& &&&& \\
& -1 & 1 &&&&& &&&&& &&&&& \\
-1 & 1 &&&&& &&&&& &&&&& & \\
&&&&& &&& -1 & 1 &&&&& &&& \\
&&&&& && -1 & 1 &&&&& &&&& \\
&&&&& && 1 &&&&& &&&&& \\
&&&&& 1 &&&&& && 1 && 1 &&& \\
&&&&& 1 & 1 &&&&& &&&&& & \\
&&&&& 1 &&&&& &&&&& && \\
&&&&& &&&&& && -1 & 1 &&&& \\
&&& -1 &&&&& 1 &&&& R &&&&& \\
&&&&& &&&&& & 1 &&&&& & \\
&& 1 &&&&& -1 &&& L &&&&& && \\
&&&&& &&&&& &&&& -1 & 1 && \\
&&&&& &&&&& &&&& -1/C && 1 & \\
&&&&& &&&&& & 1 && 1 && 1 && 1 \\
1 &&&&& &&&&& &&&&& &&
\end{array}\right]$$

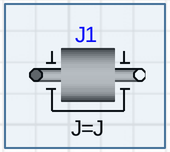
Sorted Structure

$$
\left\{\begin{array}{c}
ground.n.v \\
capacitor.n.v \\
inductor.n.v \\
resistor.n.v \\
step.n.v \\
step.p.i \\
step.n.i \\
inductor.p.v \\
resistor.p.v \\
step.p.v \\
der(inductor.p.i) \\
inductor.n.i \\
resistor.p.i \\
resistor.n.i \\
capacitor.p.i \\
capacitor.n.i \\
der(capacitor.p.v) \\
ground.n.i
\end{array}\right\}
:=
\left\{\begin{array}{c}
0 \\
ground.n.v \\
capacitor.n.v \\
inductor.n.v \\
resistor.n.v \\
f(t) \\
-step.p.i \\
capacitor.p.v \\
inductor.p.v \\
resistor.p.v \\
(inductor.p.i - inductor.n.v)/inductor.L \\
-inductor.p.i \\
(resistor.p.v - resistor.n.v)/resistor.R \\
-resistor.p.i \\
-step.p.i - resistor.p.i - inductor.p.i \\
-capacitor.p.i \\
capacitor.p.i/capacitor.C \\
-step.n.i - resistor.n.i - inductor.n.i - capacitor.n.i
\end{array}\right\}
$$

$$
\begin{bmatrix}
1 &&&&&&&&&&&&&&&&& \\
-1 & 1 &&&&&&&&&&&&&&&& \\
& -1 & 1 &&&&&&&&&&&&&&& \\
&& -1 & 1 &&&&&&&&&&&&&& \\
&&&& 1 &&&&&&&&&&&&& \\
&&&& 1 & 1 &&&&&&&&&&&& \\
&&&&&& 1 &&&&&&&&&&& \\
&&&&&& -1 & 1 &&&&&&&&&& \\
&&&&&&& -1 & 1 &&&&&&&&& \\
&&&&&&&& -1 & 1 &&&&&&&& \\
&& \frac{1}{L} &&&&&&&& 1 &&&&&&& \\
&&&&&&&&&& 1 & 1 &&&&&& \\
&&& \frac{1}{R} &&&&& -\frac{1}{R} &&&& 1 &&&&& \\
&&&&&&&&&&&& 1 & 1 &&&& \\
&&&&& 1 &&&&&&& 1 && 1 &&& \\
&&&&&&&&&&&&&& 1 & 1 && \\
&&&&&&&&&&&&&& -\frac{1}{C} && 1 & \\
&&&&&& 1 &&&&& 1 && 1 && 1 && 1
\end{bmatrix}
$$

Simple Inertia

(two states)



$$\dot{\theta}_1 = \omega_1$$

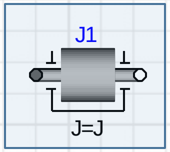
$$J_1 \dot{\omega}_1 = 0$$

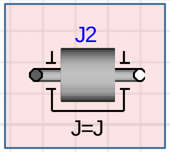
$$z = \{\dot{\theta}_1, \dot{\omega}_1\}$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 \\ 0 & J_1 \end{bmatrix}$$

Two Inertias

(four states)





$$\dot{\theta}_1 = \omega_1$$

$$J_1 \dot{\omega}_1 = 0$$

$$\dot{\theta}_2 = \omega_2$$

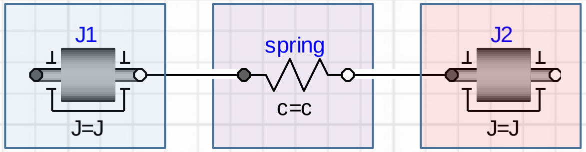
$$J_2 \dot{\omega}_2 = 0$$

$$z = \{\dot{\theta}_1, \dot{\omega}_1, \dot{\theta}_2, \dot{\omega}_2\}$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & J_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & J_2 \end{bmatrix}$$

Compliant Coupling

(four states)



$$z = \{\dot{\theta}_1, \dot{\omega}_1, \dot{\theta}_2, \dot{\omega}_2\}$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & J_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & J_2 \end{bmatrix}$$

$$\dot{\theta}_1 = \omega_1$$

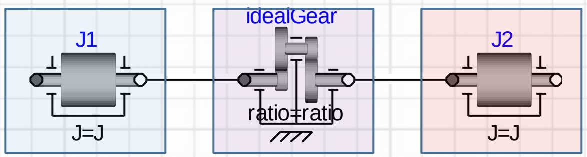
$$J_1 \dot{\omega}_1 = c(\theta_2 - \theta_1)$$

$$\dot{\theta}_2 = \omega_2$$

$$J_2 \dot{\omega}_2 = c(\theta_1 - \theta_2)$$

Kinematic Coupling

(two states)



?

$$z = \{\dot{\theta}_1, \dot{\omega}_1, \dot{\theta}_2, \dot{\omega}_2, \tau_1, \tau_2\}$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & J_1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & J_2 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & R & 1 \end{bmatrix}$$

$$\dot{\theta}_1 = \omega_1$$

$$J_1 \dot{\omega}_1 = \tau_1$$

$$\dot{\theta}_2 = \omega_2$$

$$J_2 \dot{\omega}_2 = \tau_2$$

$$\theta_1 = R\theta_2$$

$$\tau_2 = -R\tau_1$$

Index-Reduction

$$\dot{\theta}_1 = \omega_1$$
$$J_1\dot{\omega}_1 = \tau_1$$
$$\dot{\theta}_2 = \omega_2$$
$$J_2\dot{\omega}_2 = \tau_2$$
$$\theta_1 = R\theta_2$$
$$\tau_2 = R\tau_1$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & J_1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & J_2 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -R & 1 \end{bmatrix}$$

$$\frac{d}{dt}\left(\theta_1 = R\theta_2\right) \Rightarrow \dot{\theta}_1 = R\dot{\theta}_2$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & J_1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & J_2 & 0 & -1 \\ 1 & 0 & -R & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -R & 1 \end{bmatrix}$$

$$\dot{\theta}_1 = R\dot{\theta}_2 \Rightarrow \omega_1 = R\omega_2$$

$$\frac{d}{dt}\left(\omega_1 = R\omega_2\right) \Rightarrow \dot{\omega}_1 = R\dot{\omega}_2$$

$$\frac{\partial f}{\partial z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & J_1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & J_2 & 0 & -1 \\ 0 & 1 & 0 & -R & 0 & 0 \\ 0 & 0 & 0 & 0 & -R & 1 \end{bmatrix}$$

Models Are Software

» High quality IDEs
» Version Control
» Testing Frameworks
» Automatic Updates, Build Systems
» Open Source
» Purchasing Content/Tools Online

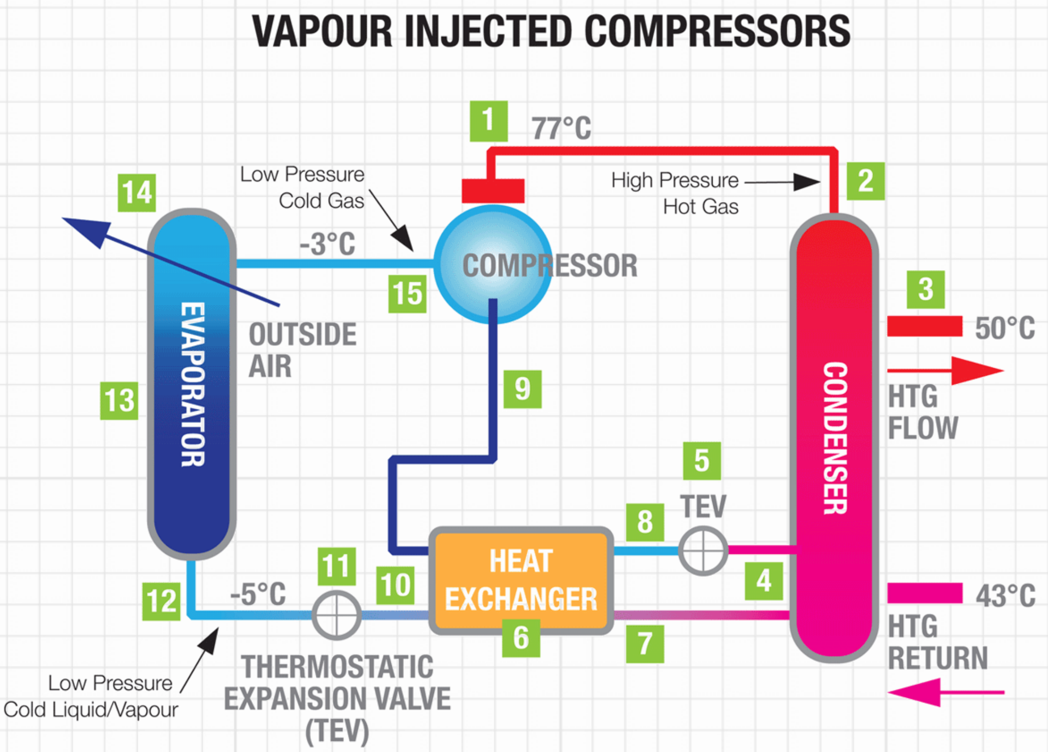
The Future

Fluid System Modeling

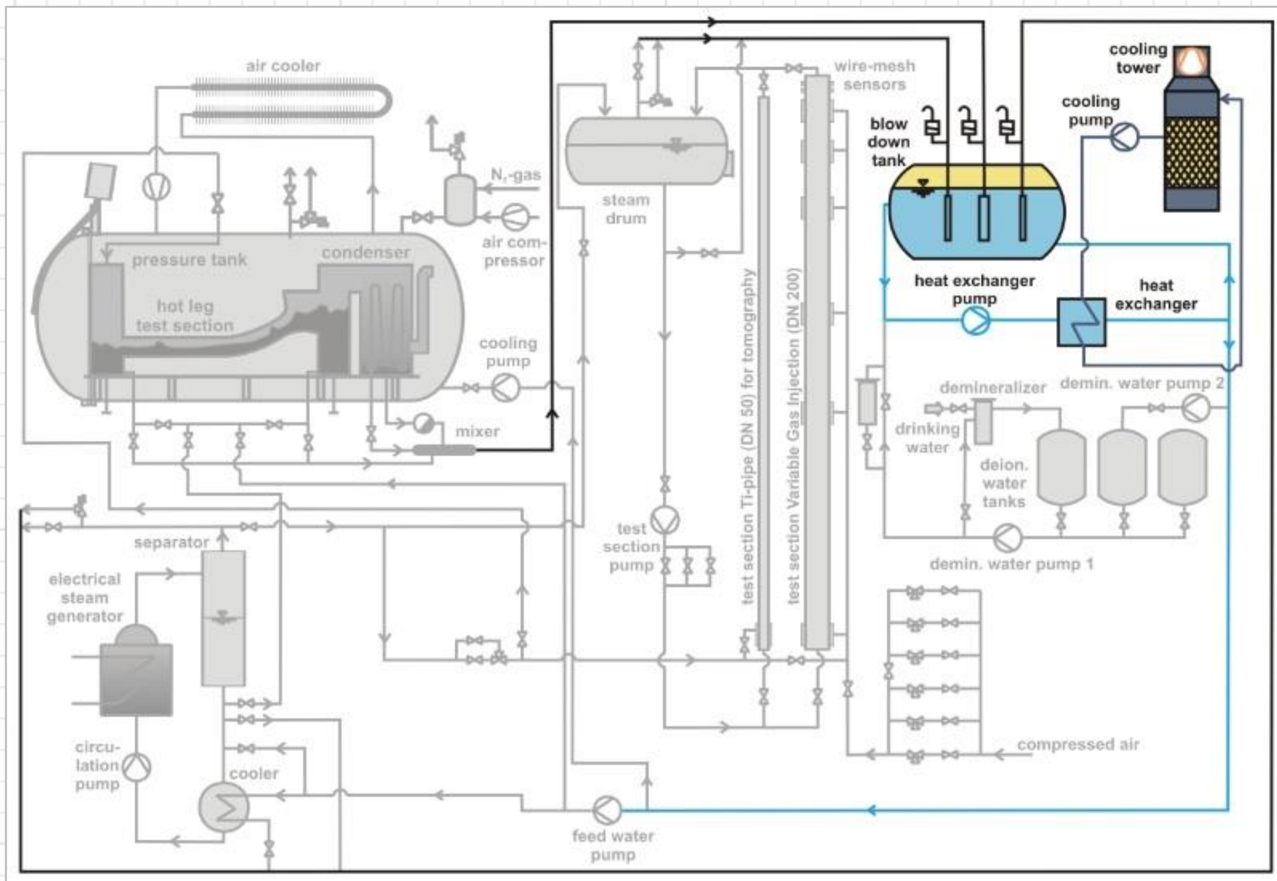
- » Fluid systems are hard
  - > Different fluid representations
    - + Compressible vs. incompressible
    - + Choice of states (pressure, density, mass, temperature, enthalpy)
  - > Different property models and equations of state
  - > Multi-phase
  - > Extremely non-linear
  - > No analogies to other domains
- » Fluid is a “component” by itself (that flows through other components!)
  - > Where is it defined?
  - > How is it “connected”?
  - > How to define modular boundaries?

Fluid System

- High pressure or high temperature channels might require more detailed models.
- Heat exchanger paths may or may not be the same fluid.
- Where (in a diagram) should these decisions be made?



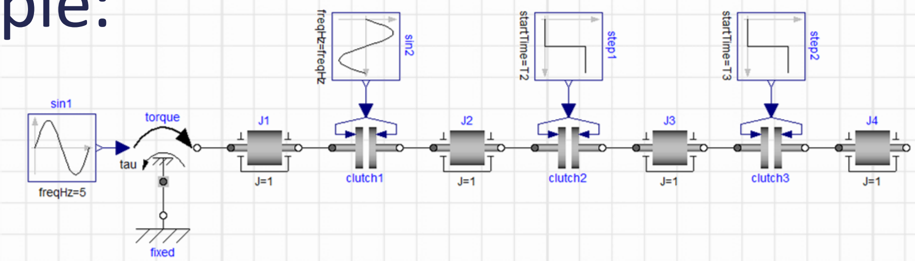
Fluid System



Variable Structure

» Modelica currently requires the number of equations and unknowns to be **static**.

» Example:



» Need to handle:

> Discontinuities (that generate impulses, e.g. collisions)
> Actors that enter and leave system

Model Reduction

» Biggest need in getting to the ideal iterative "V" process.

» Declarative approach helps here:

  > Exposes underlying structure of the problem
  > Avoids brute force numerical methods
  > Allows optimizations to be applied automatically and reliably

» Linear: Not so bad

» Non-linear: Quite challenging

Dynamic Programming

$$V(x) = \max_{a \in \Gamma(x)} \{F(x,a) + \beta V(T(x,a))\}$$

- Bellman's Principle applied to control systems
- Suffers from the "Curse of Dimensionality"
  - Cost goes up as x^n
- Previously impractical, but armed with "The Elephant", increasingly within reach

Scalability

» Try to imagine what it was like in 1990.
  > No “search engines”
  > No Wikipedia
  > Dial-up services (Compuserve, AOL, etc)

» What changed?
  > Internet: 1970s
  > TCP/IP: 1982
  > HTTP v0.9: 1991

» In order to develop a scalable eco-system you need standards to promote the development of tools and content.
  > Stability
  > Level playing field

Engineering 2.0

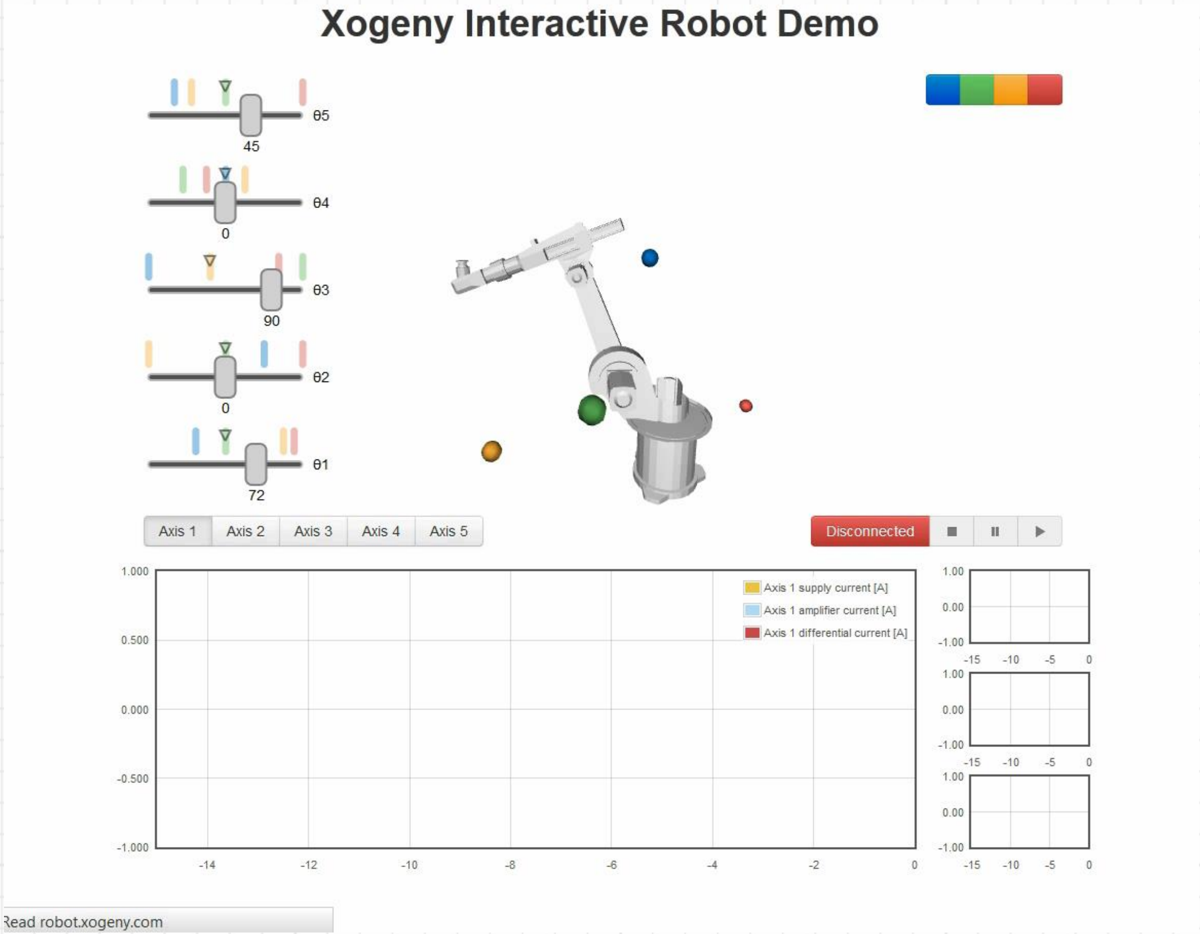
- Web 2.0
  - Where is that for engineering?
  - How has the web really changed the way industry designs products?
  - Still focused on monolithic eco-systems
- Collaboration
  - Facilitated by open, client-server architected systems
- Model deployment
  - Lack of vision and technology to help “deploy” models for greater impact
- Elastic Computing
  - FMI → FMQ & PyFMQ
- Resources and Talent

GitHub

Fork me on GitHub

- Releasing lots of stuff under Creative Commons license on GitHub
  - LotkaVolterra (Modelica package)
    - Companion models for Equations to Components blog post
  - Kinematics (Modelica package)
    - Companion models for Kinematic Transmissions blog posts
  - XogenyTest (Modelica package)
    - Companion models for future blog post on model testing
  - pyfmq (Python package)
    - Client module for forthcoming cloud simulation service
    - Looking for testers ;-)

Web-Based Simulation



www.xogeny.com

Conclusion

- In the beginning, engineering was ahead of software, now **software is ahead of engineering**.
- **Models are software**
- **Symbolic manipulation** will be a requirement in the future and, therefore, so will a **declarative** approach to modeling.
- More emphasis on the **model development process**
- Product development will be a **fusion** of CAD, lumped models, CFD/FEM, control laws and embedded systems.
  - What technologies are going to get us there in a scalable way?

Thanks for your Attention