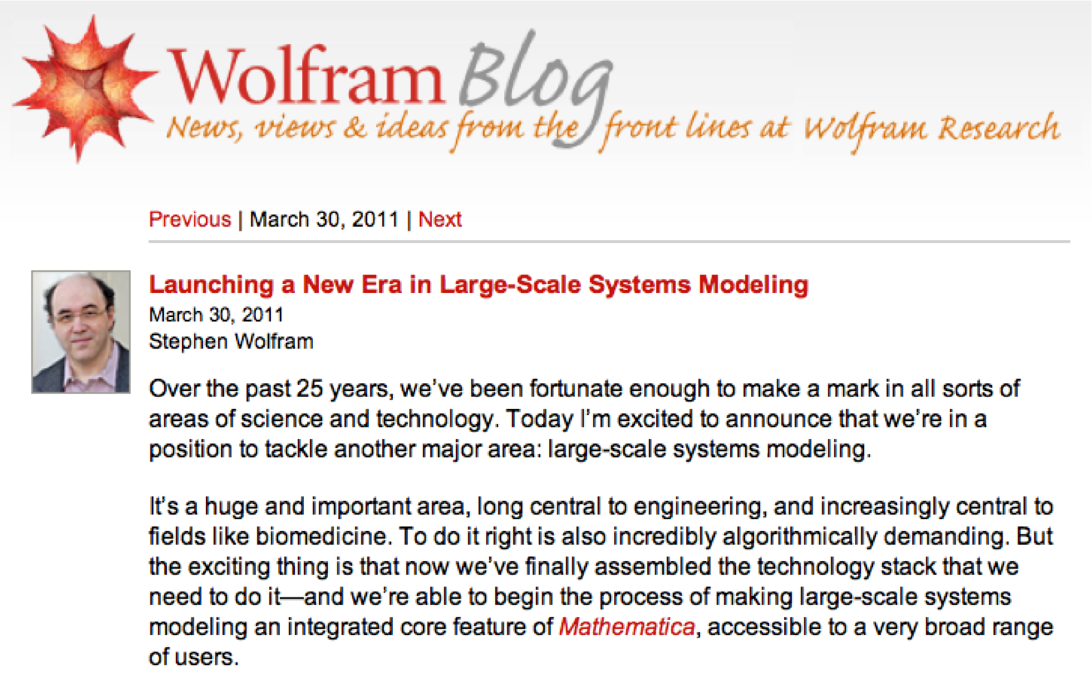
WOLFRAM
TECHNOLOGY CONFERENCE

A Formal Approach for Modeling and Simulation

Yves A. Papegay
INRIA/COPRIN

Motivation

A Communication in the Wolfram Blog



Wolfram Blog
News, views & ideas from the front lines at Wolfram Research

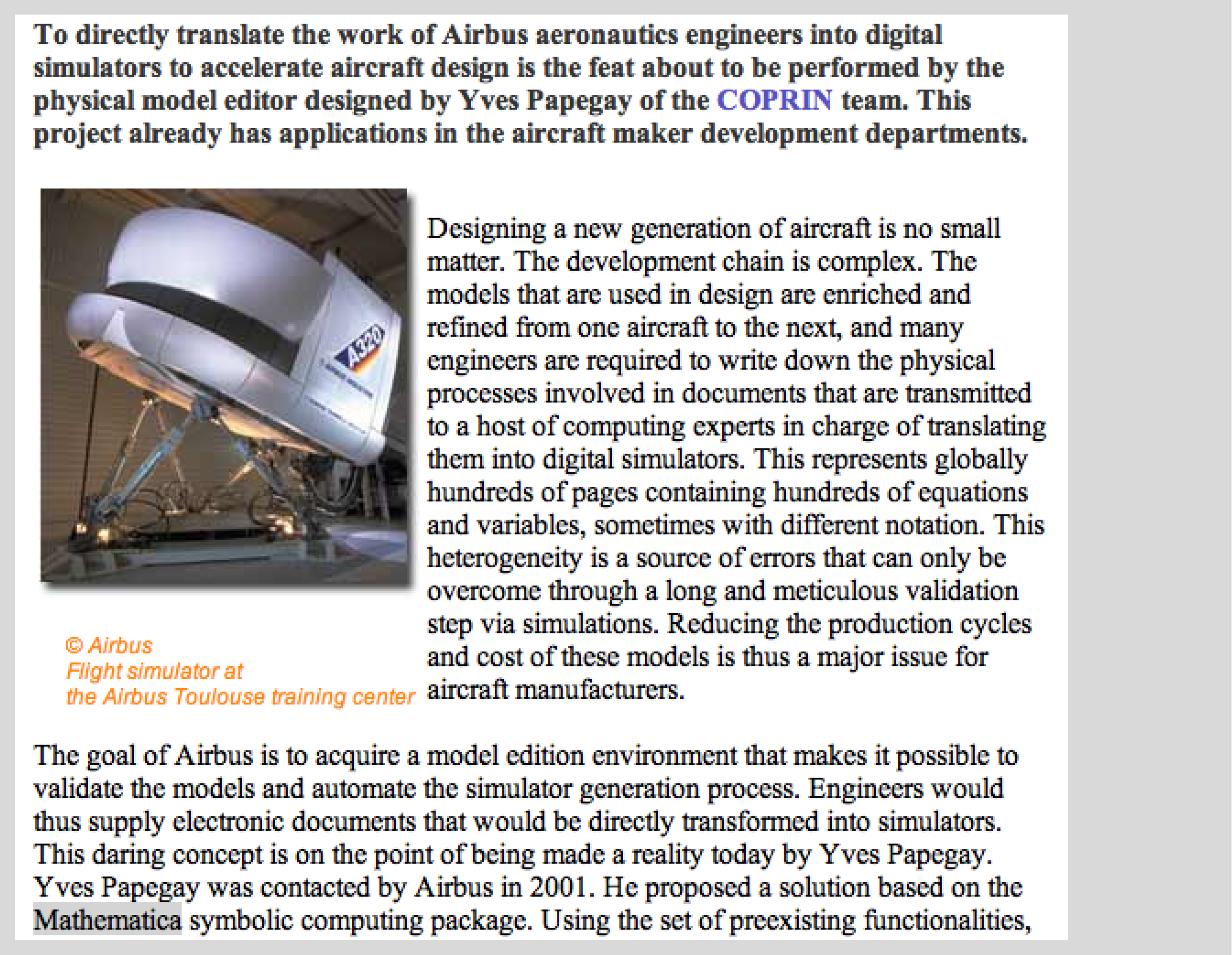
Previous | March 30, 2011 | Next

Launching a New Era in Large-Scale Systems Modeling
March 30, 2011
Stephen Wolfram

Over the past 25 years, we've been fortunate enough to make a mark in all sorts of areas of science and technology. Today I'm excited to announce that we're in a position to tackle another major area: large-scale systems modeling.

It's a huge and important area, long central to engineering, and increasingly central to fields like biomedicine. To do it right is also incredibly algorithmically demanding. But the exciting thing is that now we've finally assembled the technology stack that we need to do it—and we're able to begin the process of making large-scale systems modeling an integrated core feature of *Mathematica*, accessible to a very broad range of users.

PhD in 1991 on "Symbolic Tools for Dynamics Modeling".



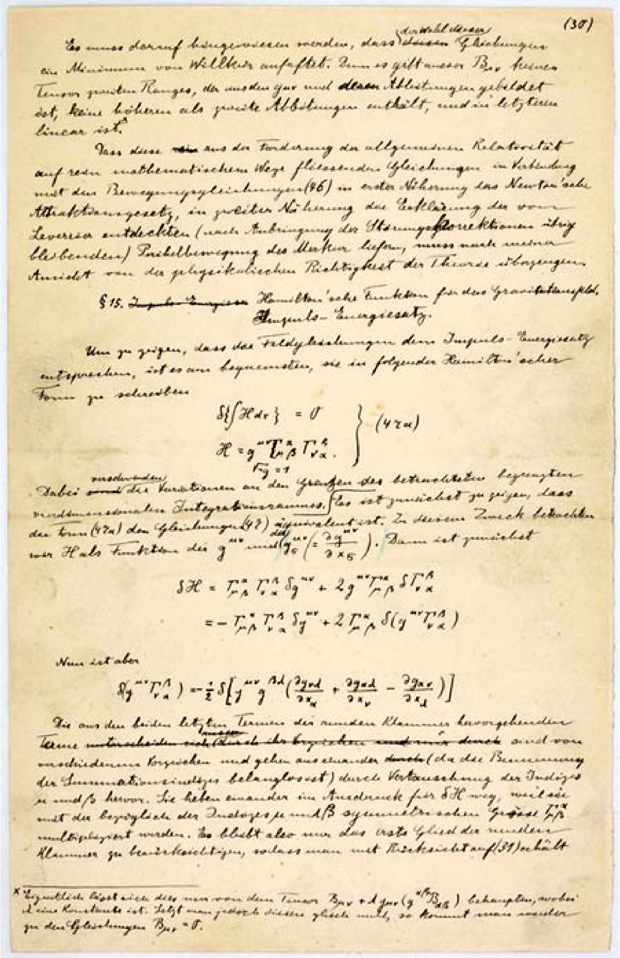
To directly translate the work of Airbus aeronautics engineers into digital simulators to accelerate aircraft design is the feat about to be performed by the physical model editor designed by Yves Papegay of the COPRIN team. This project already has applications in the aircraft maker development departments.

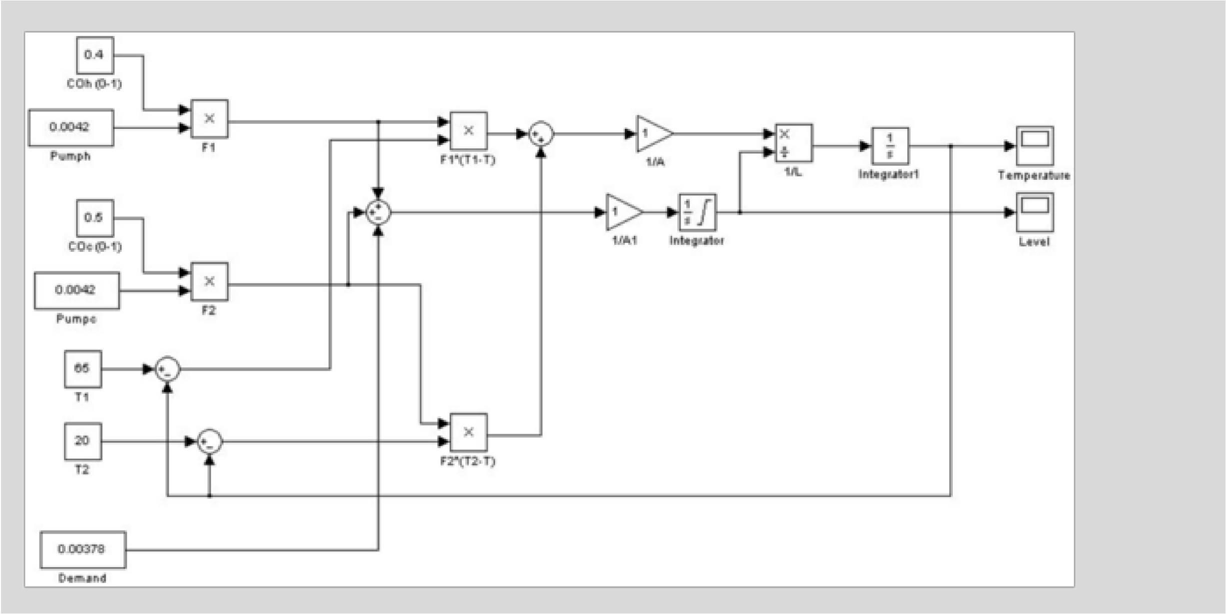
© Airbus
Flight simulator at
the Airbus Toulouse training center

Designing a new generation of aircraft is no small matter. The development chain is complex. The models that are used in design are enriched and refined from one aircraft to the next, and many engineers are required to write down the physical processes involved in documents that are transmitted to a host of computing experts in charge of translating them into digital simulators. This represents globally hundreds of pages containing hundreds of equations and variables, sometimes with different notation. This heterogeneity is a source of errors that can only be overcome through a long and meticulous validation step via simulations. Reducing the production cycles and cost of these models is thus a major issue for aircraft manufacturers.

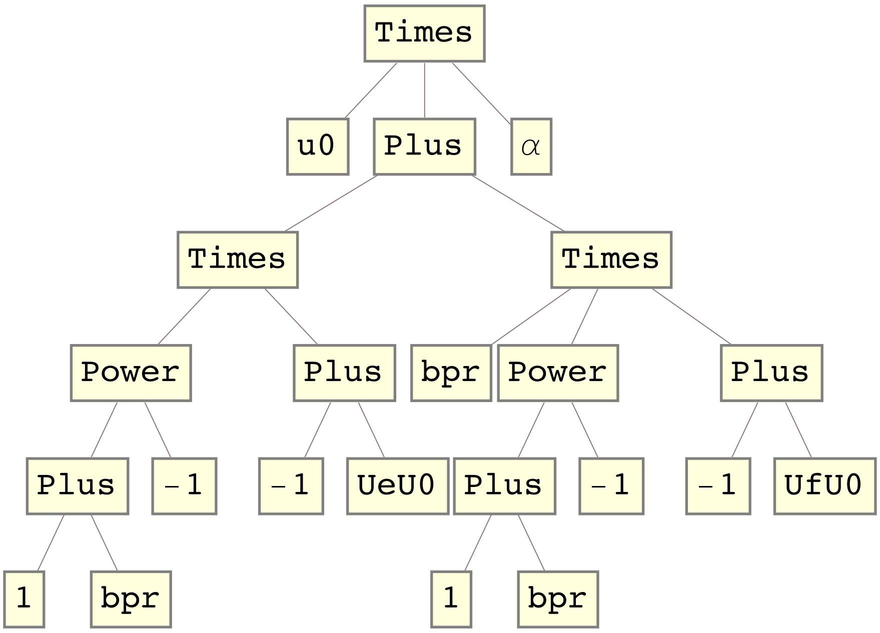
The goal of Airbus is to acquire a model edition environment that makes it possible to validate the models and automate the simulator generation process. Engineers would thus supply electronic documents that would be directly transformed into simulators. This daring concept is on the point of being made a reality today by Yves Papegay. Yves Papegay was contacted by Airbus in 2001. He proposed a solution based on the Mathematica symbolic computing package. Using the set of preexisting functionalities,

Modeling is a symbolic activity

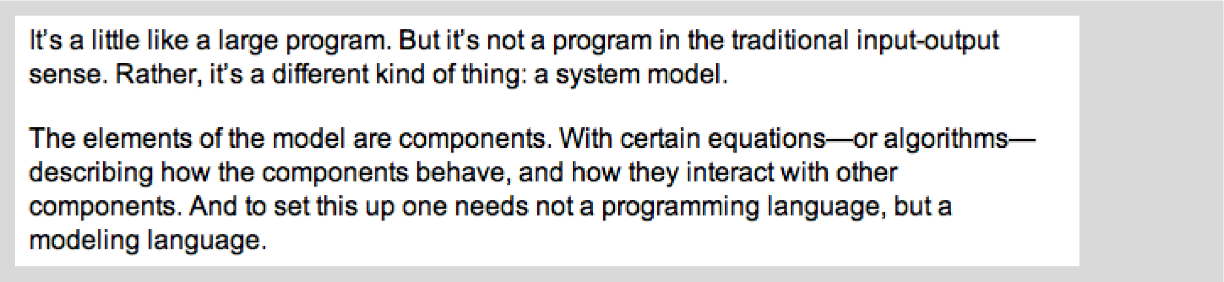




$$\text{u0}\left(\frac{-1+\text{UeU0}}{1+\text{bpr}}+\frac{\text{bpr}\,(-1+\text{UfU0})}{1+\text{bpr}}\right)\alpha$$

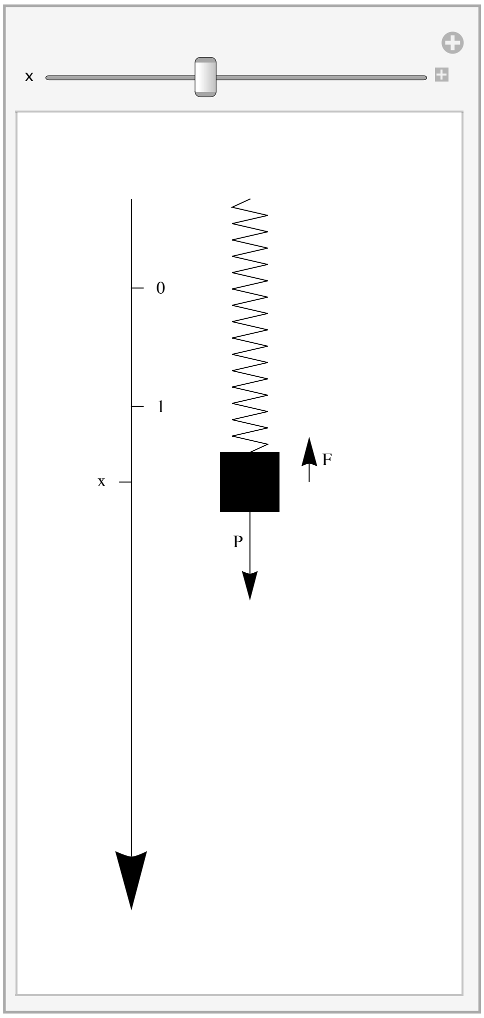


Modeling is descriptive



It's a little like a large program. But it's not a program in the traditional input-output sense. Rather, it's a different kind of thing: a system model.

The elements of the model are components. With certain equations—or algorithms—describing how the components behave, and how they interact with other components. And to set this up one needs not a programming language, but a modeling language.



Symbol	Description	Unit
l	Length at rest	m
k	stiffness	N.m^{-1}
b	damping	N.m^{-1}.s
m	Mass	kg
x	Position	m
v	Speed	m.s^{-1}
F	Force	N

$$g = 9.81$$
$$F = P + F_{spring} + F_{damp}$$
$$P = m\,g$$
$$F_{spring} = -k\,(x - l)$$
$$F_{damp} = -b\,v$$
$$F = m.\gamma$$
$$v = \frac{dx}{dt}$$
$$\gamma = \frac{dv}{dt}$$

Modeling is descriptive

Implementation requirement : managing a “context”

What are the variables involved in the model ?

What are the equations involved in the model ?

"Pendulum1"

```
variableList["Pendulum"]
equationList["Pendulum"]
```

"Pendulum2"

```
CurrentValue[nbo, {TaggingRules, listVariables}]
CurrentValue[nbo, {TaggingRules, listEquations}]
```

Modeling is based on variables

Implementation requirement : accessing extended symbols

```
x_y = 5
```

```
? x_y
```

```
y = 0
```

```
x_y
```

```
Clear[x_y, x_0]
```

Modeling is based on objects

Implementation requirement : mimicking object-oriented programming

```
Model["Pendulum", listVars → {Variable[l, Unit → m], Variable[k, N.m^-1], ...},
 listEqs → {Equation[Variable[g, Unit -> N.m^-1] == 9.81, Type → Constant] ...}]
```

```
model
```

```
ViewModel[model]
```

Modeling is based on programs

Implementation requirement : computable representation of Algorithms

```
algo // TableForm
```

```
expr = Fold[ReplaceAll, s, Reverse[algo]]
```

```
D[expr, bpr]
```

```
D[algo, bpr]
```

Implementation requirement : computable representation of Numerical Tables

Modeling led to simulation

Implementation requirement : switching between evaluated and held forms

```
nmodel = Model["Pendulum", listVars → {l, k, b, m, x, v, F, g, F_spring, F_damp, γ},
   listEqs → {g → 9.81`, F → P + F_damp + F_spring, P → g m, F_spring → -k (-l + x), F_damp → -b v, F → m.γ, v → D[x, t], γ → D[v, t]}]
```

```
Model[Pendulum, listVars → {l, k, b, m, x, v, F, g, F_spring, F_damp, γ},
 listEqs → {g → 9.81, F → P + F_damp + F_spring, P → g m, F_spring → -k (-l + x), F_damp → -b v, F → m.γ, v → 0, γ → 0}]
```

```
Attributes[Model] = {HoldRest}
```

```
{HoldRest}
```

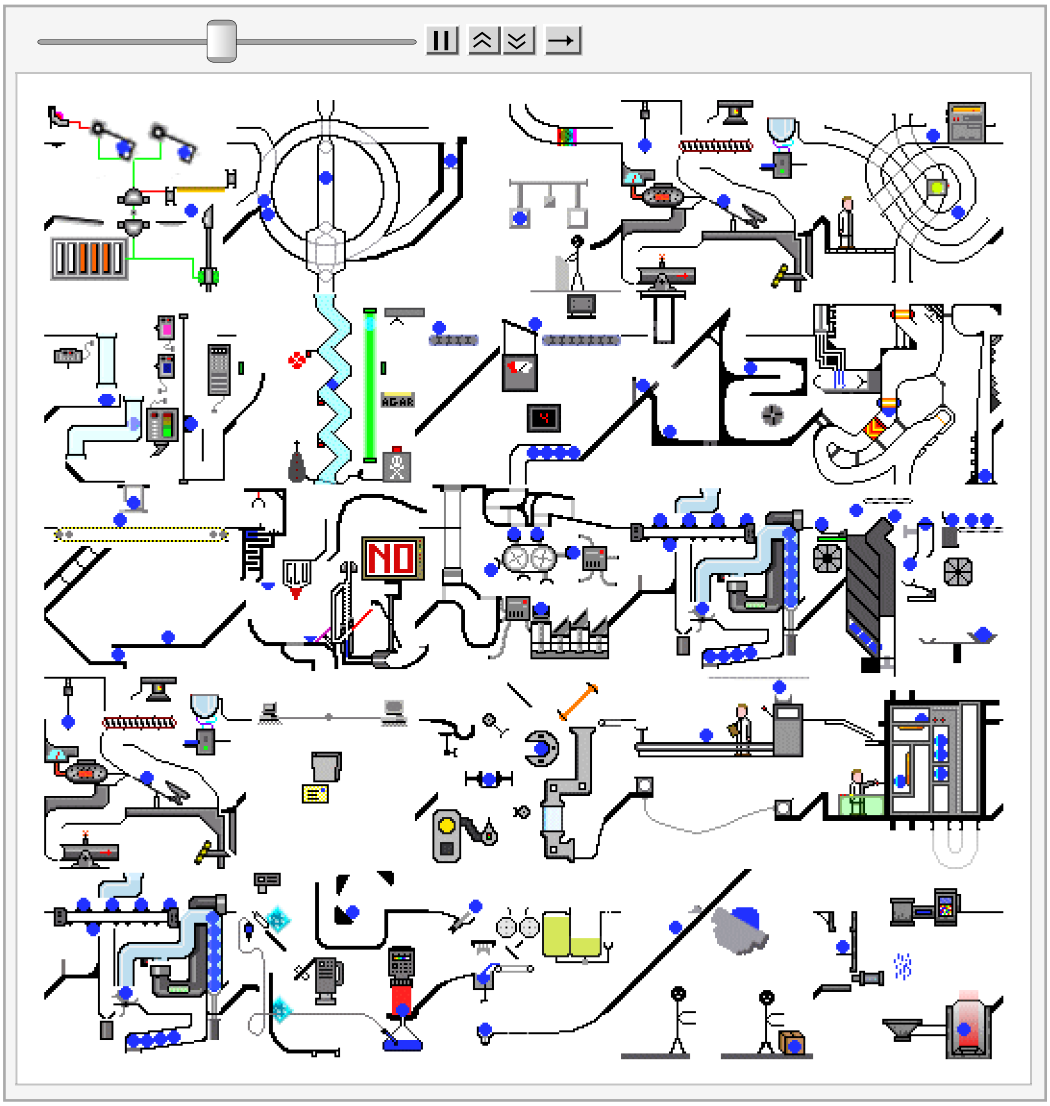
```
ClearAll[Model]
```

```
GetEquation[m_Model] := listEqs /. (Rest[m] /. Model → List)
```

```
GetEquation[nmodel]
```

```
{g → 9.81, F → P + Fdamp + Fspring, P → g m, Fspring → -k (-1 + x), Fdamp → -b v, F → m.γ, v → 0, γ → 0}
```

Conclusion



Good Luck !

Code

```
BeginModel[s_String] := (CurrentModel = s; variableList[s] = {}; equationList[s] = {};)
EndModel[] := (CurrentModel = None;)
DeclareVariable[s_] := AppendTo[variableList[CurrentModel], s]
DeclareEquation[s_] := AppendTo[equationList[CurrentModel], s]

ViewModel[Model[s_String, r__Rule]] := OpenerView[{Row[{Style["Model", Bold], s}, "    "], Column[Map[ViewModel, {r}]]}]
ViewModel[Rule[id_, l_]] := OpenerView[{Style[id, Italic], Column[Map[ViewModel, l]]}]
ViewModel[Variable[id_]] := Row[{Style["Variable", Bold], id}, "    "]
ViewModel[Equation[expr_]] := Row[{Style["Equation", Bold], expr /. Variable[any_] ⧴ any}, "    "]
```

$$\mathbf{model} = \mathbf{With}\Big[\{\mathbf{vars} = \{l, k, b, m, x, v, F, g, F_{\text{spring}}, F_{\text{damp}}, \gamma\}\}, \mathbf{Model}\Big[\text{"Pendulum"},\\ \mathbf{listVars} \to \mathbf{Map}[\mathbf{Variable}, \mathbf{vars}], \mathbf{listEqs} \to \mathbf{Map}\Big[\mathbf{Equation}[\#[[1]] == \#[[2]]]\ \&, \Big\{g \to 9.81\grave{}, F \to P + F_{\text{damp}} + F_{\text{spring}},\\ P \to g\, m, F_{\text{spring}} \to -k\ (-l + x), F_{\text{damp}} \to -b\, v, F \to m.\gamma, v \to \frac{d}{dt}[x], \gamma \to \frac{d}{dt}[v]\Big\}\Big]\ /.\ \mathbf{Map}[\# \to \mathbf{Variable}[\#]\ \&, \mathbf{vars}]\Big]\Big];$$

$$\mathbf{algo} = \Big\{\tau_\lambda \text{ -> } \mathbf{ttburn} / t_0, \tau_r \text{ -> } 1 + (\gamma - 1) / 2\, m_0{}^2, \tau_c \text{ -> } \pi_c\ (\gamma - 1)\ \eta_c / \gamma, \tau_\lambda \text{ -> } \mathbf{ttburn} / t_0,\\ f \text{ -> } \mathbf{cp}\, t_0 \frac{\tau_\lambda - \tau_r\, \tau_c}{\mathbf{hfuel}}, \tau_t \text{ -> } 1 - \frac{\tau_r}{\tau_\lambda}\ ((\tau_c - 1) + \mathbf{bpr}\ (\tau_f - 1)), \mathbf{UeU0} \text{ -> } \sqrt{\frac{\tau_r\, \tau_c\, \tau_t - 1}{\tau_r - 1}\ \frac{\tau_\lambda}{\tau_r\, \tau_c}}, \tau_f \text{ -> } \pi_f\ (\gamma - 1)\ \eta_f / \gamma,\\ \mathbf{UfU0} \text{ -> } \sqrt{\frac{\tau_r\, \tau_f - 1}{\tau_r - 1}}, \mathbf{Tdm}_0 \text{ -> } u_0 \left(\frac{1}{1 + \mathbf{bpr}}\ (\mathbf{UeU0} - 1) + \frac{\mathbf{bpr}}{1 + \mathbf{bpr}}\ (\mathbf{UfU0} - 1)\right) \alpha_{\text{thrust}}, s \text{ -> } \frac{f}{10^6\ (1 + \mathbf{bpr})\ \mathbf{Tdm}_0}\Big\};$$