An Industrial Model Based Development Systems Engineering Strategy

American Modelica Conference 2018, October 9-10 2018, Cambridge, MA

Presenters:

Johan Åkesson, Modelon
Kristian Tuszyński, CCS
Clas Jacobson, CCS
AGENDA

• Team
• Key Points
• Drivers
• Tools chain
• Application examples
• Summary
TEAM

Modelon
Johan Åkesson, Johan Andreasson, Magnus Gäfvert, Magda Axelsson, Katrin Prolss, Iakov Nakhimovski, Fredrik Magnusson, Bryan Eisenhower

Carrier
Trevor Bailey, Degang Fu, Qingfan Zeng, Clas Jacobson, Kristian Tuszynski, Rui Huang, Chen Zhang, Lishan Wang, Dongzhi Guo

External
John Cassidy (UTC, retired), Larry Biegler (CMU), Karl Åström (LTH), Carl Laird (Sandia), Kevin Otto (RSS)
KEY POINTS

Modelica is a modeling language that (1) captures physics and is useful for modeling at the (2) system level and for modeling (3) heterogeneous systems:

- Need for system models of different scope, complexity and domains
- One modelling language

The use of Modelica is on simulation but goes beyond in “systems engineering” the (re)use of models for variability and robustness analysis, optimization and analysis of design freedom, and control design and analysis:

- Started with control design
- Goal: Unification of model development to Modelica

CCS is using Modelica for system level modeling and the Modelon tool chain to capture system level modeling and to deploy widely using library architectures, GUI and Python infrastructure:

- Support for a tools set that allows unification of models
- Steady-state/Dynamic/Cluster execution/Optimization/Variability
DRIVERS

Systems Engineering Needs

Engineering effectiveness – **drive designs by models** – validate requirements and drive efficient testing

Need to **deal with increased system integration complexity** – Components > Chiller > Chiller plant > Building

**Regulatory environment demands design efficiencies** (new technologies, refrigerant changes) and energy efficiency (whole building level)

---

**Energy Savings in Commercial Buildings**

- 45-80% Demand Reduction
  - 5-20% ↓
  - 10-15% ↓
  - 15-15% ↓
  - 20-30% ↓

**Baseline Footprint (ASHRAE 90.1)**

- **Information Systems & Networks**
  - Occupancy, utility rate, …

- **Comfort & Ventilation Systems**
  - Ventilation, air movement…

- **Thermal Systems**
  - Thermal recovery, storage…

- **Green Buildings**
  - Architecture, envelope…

**Technology Risk & Readiness**

- Reduce risk/enhance maturity
- Make solutions scalable & robust
- Drive commercial adoption

---

NEED TO EXPOSE DIFFERENT VIEWS…
Heterogeneous Modeling, Different Fidelities…

Figure 1: Proposed Model Based Development Process
TOOLCHAIN ECO-SYSTEM

Control

Design

Optimization

Variability

MATLAB & SIMULINK

IPM BOLT

python

COIN-OR

IPOPT

IBM. Rational software

Jazz

Dymola

Modelon

fmi

Modelon WAMS

MODELICA

Version 9 OCT

OCT

Control

Design

Optimization

Variability

MATLAB & SIMULINK

IPM BOLT

python

COIN-OR

IPOPT

IBM. Rational software

Jazz

Dymola

Modelon

fmi

Modelon WAMS

MODELICA

Version 9 OCT

OCT
MODELON PLATFORM FOR MBD

WAMS

Create

Customizations

Deploy

OPTIMICA Compiler Toolkit

Compiler for Modelica and FMI

Numerical solvers for simulation and optimization

Compile

Compute
VARIABILITY – ENGINEERING METHODS

Analysis methods
- Model-based manufacturing analysis
- Test variation: Gage R&R, internal audits, extra testing

Design methods
- Adjust the relationship between mean (designed), deviation, tails of KPI’s
- Feasibility analysis
- Sensitivity/variability reduction

Component variability

Performance Indicators

Reduced variation

Supply chain variability improvements ($)

System Design Changes
MODEL BASED CONTROL DESIGN

Software tools to support automatic code generation and testing

Dymola
Dynamic model

FMI Toolbox
Run model in Simulink

Matlab/Simulink/Stateflow
Create controller
Automatic code generation
Tool boxes: EmbeddedCoder

Legacy code

Generated code
System level considerations... heterogeneous system...

Examine limits of performance using MPC control design...

Advantages:
- Predictive
- Systematic: no if-then-else and extensive trial and error tuning
- Multivariable, Model Based
- Guarantees: Performance and Constraint satisfaction
- Large success in the process industry

Implementation done in cloud environment...

Thanks to F. Borrelli (Berkeley) – joint work with UTRC
Modelica is a modeling language that captures physics and is useful for modeling at the system level and for modeling heterogeneous systems (and both steady state & dynamics).

CCS is using Modelica for system level modeling and the Modelon tool chain to capture system level models and to deploy widely using library architectures, GUI and Python infrastructure.

The use of Modelica is on simulation but goes beyond in “systems engineering” the (re)use of models for variability and robustness analysis, optimization and analysis of design freedom, and control design and analysis.