Machine Learning In Model Based Engineering

Sunil Elanayar, Ph.D.
Phillip Austin
Juan Sanchez De Muniaín
Presentation outline

Global Product Data Interoperability Summit | 2018

• Model Based Engineering and Computational Design Synthesis (CDS)
  • Design Synthesis done manually
  • CDS using Graph Transformations
• Function Behavior Structure (FBS) role in CDS
  • Framework for CDS
  • FBS Levels in Modeling
  • Role of Knowledge Based Engineering (KBE) in CDS
  • Building Functional Structures using MOKA (Methodology and software tools Oriented to Knowledge-based engineering Applications)
• Behavior Modeling and Digital Twins
  • Auto Suspension Systems, Production Systems
  • Need for Black Box modeling
  • Radial Basis Function Networks for Dynamic Black Box Modeling
  • Learning Dynamic Models
• Proposed Workflow for CDS with Digital Twin
Authors’ bios

Global Product Data Interoperability Summit | 2018

• **Sunil Elanayar**
  Joined Boeing in 2007 from Dassault Systemes. Currently an IT Manager in Engineering Systems. Managing teams working in MBSE, Product Standards. In the past, he has worked in Visualization, New Wiring Systems, Aerodynamics, Knowledge Based Engineering (KBE), Machine Learning, and Optimization.

• **Phillip Austin**

• **Juan Sanchez De Muniaín**
  Software developer at Boeing since 2014, now on the 2CES PLM effort at Boeing; before that, developing computer-aided engineering and manufacturing applications primarily around product standards.
Enabling Model Based Engineering

Global Product Data Interoperability Summit | 2018

To Enable MBE capabilities, tools need the ability to:

- Model products at multiple levels of abstraction, i.e. Function, Behavior and Structure (FBS)
- Formalize Engineering Knowledge with a view to manipulate, transform, and reuse it.
- Enable the quick generation and adaptation of designs (especially in the conceptual phases.)
- Decrease the tedium in routine design tasks
- Support innovation in the design process
- Help keep behavioral models current in the light of IOT
- Support Computational Design Synthesis
- Support Digital Twin concepts

Benefits of Computational Design Synthesis (CDS):

1. Increases the efficiency of the design process and the creation of new solutions,
2. Facilitates design reuse during concept generation,
3. Enables the exploration of larger design solution spaces, and thereby removes psychological bias that may limit designers to previous solutions or to specific engineering domains.
The Process of Design Synthesis: Functional Decomposition

**STEP 1:** transform customer needs into an overall product function

**Clean Teeth**

- HE (Human Energy)
- Teeth
- Paste
- Hand

**Legend:**
- HE - Human Energy
- TME - Translational Mechanical Energy
- RME - Rotational Mechanical Energy
- EE - Electrical Energy
- CS - Control Signal

**STEP 2:** decompose function into elemental subfunctions using a repeatable functional representation

- Import HE
- Convert HE to CS
- ON/OFF

- Store EE
- Supply EE
- Transfer EE
- Actuate EE
- Convert EE to RME
- Transfer RME
- Convert RME to TME
- Transfer TME

- Import Mixture
- Mix Mixture & Solid
- Stop Mixture
- Export Mixture

- Import Human
- Guide Human
- Export Human

- Import Solid
- Separate solid
- Export Solid

- Mixture
- Export Mixture
- Mixture & Solid
- Hand

- Teeth
- Paste / Debris
The Process of Design Synthesis: Form Mapping

Legend:
HE - Human Energy
TME - Translational Mechanical Energy
RME – Rotational Mechanical Energy
EE – Electrical Energy
CS – Control Signal

STEP 3: seek solutions to sub-functions
Computational Design Synthesis: Graph Transformations

Function Structure of Electric Toothbrush

Configuration Flow Graph of Electric Toothbrush
Graph Transformations / Graph Grammars

Global Product Data Interoperability Summit | 2018

Function Structure
1. Store EE → EE → Supply EE → EE
2. RME → Transfer RME → RME
3. Mixture → Stop Mixture

Configuration Flow Graph
1. Battery → EE
2. RME → Driveshaft RME → Rotational Coupler RME
3. Mixture → Seal

Function Tree
Rule x

CFG Knowledge Object Tree
Rule xxx

Copyright © 2018 Boeing. All rights reserved.
FBS Framework for Computational Design Synthesis in MBE
Global Product Data Interoperability Summit | 2018

Translation of Requirements into System level specifications

System Level Specifications Derived from Requirements

Functional decomposition

Subsystems and component level specifications corresponding function hierarchy

Realization of structure and behavior

Structure and behavior described with Knowledge Objects, physical models, and parameter relations

Performance evaluation using performance indicators

Physical phenomenon (Behavior)

State Space Model of Behavior

\[ X = AX + U \]
FBS Levels and Process of Computational Design Synthesis

Ref: [7,9]
Knowledge Object Trees and CAD Interactions

CRUD

- Create (Objects, Features, Links, Rules etc.)
- Read (Attributes, Values, Measures)
- Update (Inputs, Objects, Features, Links)
- Delete (Objects, Features, Links)

KBE Side

Knowledge Object tree

Dispatcher Library

Mapping Table

V5

CAD feature tree
Enriching Computational Design Synthesis Using Knowledge Objects

Current KBE Focus Areas

Requirements

Functional

Logical

Physical

Allows Extension into Detailed Design Phases

Control

Usage, Needs Requirements

Architecture Interfaces

Information

MMI Virtual surrounding

Structure

Physical systems

Electronics Optronics

Model Integration and System Simulation

Models of physical objects

Models of information objects, needs, services

Current KBE Focus Areas

CDS with KBE

Systems Engg Focus Areas
Using MOKA ICARE forms to build Functional Structure Graphs

The informal model
Elaborating ICARE forms

Knowledge Configuration Flow Graph

The formal model
Elaborating Product model and Design Process model

Institution with Input Parameters

CAD/PDM Artifacts

- ICARE forms are templates that store the knowledge in five categories:
  - Illustrations – for describing any case studies or relevant examples
  - Constraints – limitations on Entities
  - Activities – the description of the elements of the design process
  - Rules – the means of regulating the Activities and providing the ‘know-how’ or strategy of the design process
  - Entities – the objects that describe the product (Entities may be further classified into E-structure and E-function)
Digital Twins: What are they good for?

Digital twins help manufacturers and OEMs by helping with:
- Visualizing products in use, by real users, in real-time
- Building a digital thread, connecting disparate systems and promoting traceability
- Refining assumptions with predictive analytics
- Troubleshooting far away equipment
- Managing complexities and linkage within systems-of-systems
- Shared Conceptualization, Comparison, and Collaboration

Ref: [16]
Behavior Modeling and The Digital Twin

State Space Equations

\[
\dot{Z} = AZ + BF_a + \dot{Z}_r
\]

\[
\begin{bmatrix}
\dot{Z}_1 \\
\dot{Z}_2 \\
\dot{Z}_3 \\
\dot{Z}_4
\end{bmatrix} =
\begin{bmatrix}
-\frac{K_S}{M_{us}} & -\frac{C_S}{M_{us}} & 0 & -1 \\
0 & -\frac{C_S}{M_{us}} & 0 & 0 \\
0 & 0 & -\frac{K_{us}}{M_S} & -\frac{C_S}{M_S} \\
-\frac{K_S}{M_S} & \frac{C_S}{M_S} & -\frac{K_{us}}{M_S} & -\frac{C_S}{M_S}
\end{bmatrix}
\begin{bmatrix}
Z_1 \\
Z_2 \\
Z_3 \\
Z_4
\end{bmatrix}
+ \begin{bmatrix}
0 \\
0 \\
0 \\
-1
\end{bmatrix} \dot{Z}_r
\]

Ref: [4]
Modeling Production Systems for Digital Twin

**State Space Equations**

Ref: [6]
Underwater Autonomous Vehicle Dynamics

State Space Equations

Physical model:

\[
M\ddot{z} = G(v)\dot{v} + D(v)v + \Gamma_g + \Gamma_u
\]

\[
\dot{\eta} = J_c(\eta_2)v
\]

where:
- \(M\): mass matrix: real mass of the vehicle augmented by the "water-added-mass" part,
- \(G(v)\): action of Coriolis and centrifugal forces,
- \(D(v)\): matrix of hydrodynamics damping coefficients,
- \(\Gamma_g\): gravity effort and hydrostatic forces,
- \(J_c(\eta_2)\): referential transform matrix,
- \(\Gamma_u\): forces and moments due to the vehicle’s actuators.

A 12 dimensional state vector: \(X = [\eta(6)\ v(6)]^T\).

- \(\eta(6)\): position in the inertial referential: \(\eta = [\eta_1\ \eta_2]^T\) with
  \(\eta_1 = [x\ y\ z]^T\) and \(\eta_2 = [\\phi\ \theta\ \psi]^T\). \(x, y, \) and \(z\) are the positions of the vehicle, and \(\\phi, \theta, \) and \(\\psi\) are respectively the roll, pitch and yaw angles.

- \(v(6)\): velocity vector, in the local referential (linked to the vehicle) describing the linear and angular velocities (first derivative of the position, considering the referential transform: \(v = [v_1\ v_2]^T\) with
  \(v_1 = [u\ v\ w]^T\) and \(v_2 = [p\ q\ r]^T\).

Axial propeller to control the velocity in Ox direction and 5 independent mobile fins:
- 2 horizontals fins in the front part of the vehicle (b1, b01).
- 1 vertical fin at the tail of the vehicle (d).
- 2 fins at the tail of the vehicle (b2, b02).
Need for Low fidelity Behavioral Models for Digital Twins

Models needed everywhere, but,

- What if you don’t have one from first principles?
- Are they the right fidelity?
- How do you address all disciplines?
- How do you keep them up to date?
- What about the data deluge with IOT?
- Is learning and adaptation built into the models over its lifecycle?
- Do the models apply uniformly to different levels of abstraction?

Ref: [1]
Generic Dynamic Models

State Equations

Black Box Model

Feedback / Learning

Sensor 1

Sensor 2

Sensor 3

Sensor 4

\[ x_{k+1} = f(x_k, u_k) + w_k \]

\[ y_k = h(x_k) + v_k \text{ for } k = 0, 1, 2, \ldots. \]
Black Box Neural Networks Models

Radial Basis Function Networks

\[ t_p(X) = \sum_{j=1}^{p} \lambda_j \Phi(||X - X_j^c||) + \lambda_0^T X \]

\[ |t_p(X) - F(X)| \to 0 \]

\[ \Phi(r) = \begin{cases} r^2 \log r, & \text{Duchon} \\ \sqrt{c + r^2}, & \text{Hardy} \\ r^l \exp(-r^2), & \text{Thin Plate Schagen} \end{cases} \]

\[ x_{k+1}^i = [\Lambda \Lambda_0] \begin{bmatrix} \Psi(X_{k}^i) \\ X_{k}^i \end{bmatrix} + w_k^i \]

\[ z_k^i = x_k^i + \zeta_k^i \]

Ref: [5]

Approximate System Equations
Training the Network

Global Product Data Interoperability Summit | 2018

Training / System Identification

\[ \hat{\Theta}_{N+1} = \hat{\Theta}_N + R_N \hat{\xi}_N^T \left[ I + \hat{\xi}_N R_N \hat{\xi}_N^T \right]^{-1} \]
\[ \times \left[ \gamma_{N+1} - \hat{\xi}_N \hat{\Theta}_N \right] \]

\[ R_{N+1} = R_N - R_N \hat{\xi}_N^T \left[ I + \hat{\xi}_N R_N \hat{\xi}_N^T \right]^{-1} \hat{\xi}_N R_N \]

Ref: [5]
Using the Learned system

State Estimation Using Learned System

\[
\hat{x}(k+1) = f'(\hat{x}_k, u_k) + F\hat{x}_k + bu_k + K_k[y_k - h'(\hat{x}_k) - H\hat{x}_k]
\]

\[
K_k = (2 + e_f)F\hat{P}_k H^T [(2 + e_f)H\hat{P}_k H^T + V]^{-1}
\]

\[
\hat{P}_{k+1} = l_1(F - K_k H)\hat{P}_k (F - K_k H)^T + l_2 I + l_3 Tr(\hat{P}_k) I + l_4 K_k K_k^T + l_5 Tr(\hat{P}_k) K_k K_k^T + W + K_k V K_k^T
\]

\[
\hat{P}_0 = \rho_0.
\]

Ref: [5]
Viability of Learned Network for Modeling Dynamic Systems

Inverted Pendulum

Ref: [5]

\[
\begin{bmatrix}
    x_{1,k+1} \\
    x_{2,k+1}
\end{bmatrix} =
\begin{bmatrix}
    x_{1k} - T x_{2k} \\
    x_{2k} - 3T x_{2k}^2 e^{-0.05 x_{1k}}
\end{bmatrix}
\]

\[
y_k = \sqrt{10000 + x_{1k}^2 + \nu_k}
\]
Learning Dynamic Black Box models

Global Product Data Interoperability Summit | 2018

Functional / Behavioral Model
- Propulsion Models
- Drive System Models
- Control Input
- Power Equations
- Vehicle Dynamics

Performance Model

Lifecycle Model
- As Designed Models
- As Built Models
- As Maintained Models
- IoT Data

Structural / Component Model
- Product
- Engine
- Body/Fuselage
- Wheels

Engineering/ Mfg./ Maint. Models
- Structural Models
- Mass Properties Models
- Safety Models
- Cost Models
- Maint. Models
- Mfg. Models

Ref: [1]
Proposed Workflow for CDS with Digital Twin

Global Product Data Interoperability Summit | 2018

User

Design Engineer

Knowledge Object Tree

CDS

CFG Knowledge Object Tree

PLM Repository

Digital Twin Repository

Sensor Data

Product(s)

Root Product Node

Sub-Assembly 1

Sub-Assembly 2

Sub-Assembly 3

Sub-Assembly 4

Function

Sub-Function 1

Sub-Function 2

Sub-Function 3

Sub-Function 4

 Requirement 1

 Requirement 2

 Requirement n

User

Design Engineer

Knowledge Object Tree

CDS

CFG Knowledge Object Tree

PLM Repository

Digital Twin Repository

Sensor Data

Product(s)

Root Product Node

Sub-Assembly 1

Sub-Assembly 2

Sub-Assembly 3

Sub-Assembly 4

Function

Sub-Function 1

Sub-Function 2

Sub-Function 3

Sub-Function 4

 Requirement 1

 Requirement 2

 Requirement n
Questions?
References

1. “Introduction to Model-Based Systems Engineering (MBSE) and SYSML,” Laura E. Hart, Delaware Valley INCOSE Chapter Meeting, July 30, 2015