Modeling for Everyone

developing high-end production print systems

Eugen Schindler & Bart Theelen
Systems Modeling Architecture Team

STAF Industry Day 2019
Agenda

• Applications & Products
• Software: Controller & Embedded
• Variability of Print Systems
• General Examples MBD at Océ/Canon
• Illustrate scaling-up of MBD at Océ/Canon
• Detailed examples
• Next steps & application
• Challenges in applying MBD at Océ/Canon & Conclusions
“Typical” print applications?
“Typical” Print Applications
Also print applications
Newest Océ Product Families

ColorWave 700
- 500 posters per day

Arizona 6170 XTS
- 200 signs per day

ProStream 1000
- 425,000 direct mail pieces per day

Colorado 1640
- Up to 600 banners per day

VarioPrint i-series
- 100,000 brochures per day
Breaking down a print system

VarioPrint i300

Engine

Controller
Breaking down a print system

Controller features

- UI
- Job management
- Workflows, planning
- Fleet management
- Color management
- Media management

VarioPrint i300
Breaking down a print system

VarioPrint i300

Engine features (software)

- Brings engine alive with behavior (control, measurement)
- Systems level multi-disciplinary KPIs:
  - Productivity
  - Data trends/analysis
  - Variability
  - Error handling
Variability

Same engine, different configuration (input and finishing/output)

Comparable configuration, different engines

Variability within the engine:
1. Paper input module
2. Paper transport module
3. Print module
4. Fixation module
5. Finishing equipment
model-based development is of strategic importance to realize continuous improvement ambition
Model Based Development examples

Integral productivity
(job, SW, HW, operator behavior, floor plan)

Impositions
(Finishing vs Applications)

State Machines
(Component & Interface)

- Interface specifications
- System capabilities
- Features

Performance
Model Based Development examples

- Software-in-the-loop co-simulations

- CAE/CFD/Chemistry simulations

- Multi-disciplinary system models

- Model-based control

- Virtual Environments (e.g. service training)
We don’t make software, we make models!

Model-based development is of strategic importance to realize continuous improvement ambition.
Replacing Documents by Models → Modelling Isles

- During development, a lot of knowledge is captured in many different models.

Variety in languages, models & tools

- Knowledge comes together in physical (prototype) implementations.

Variety in purposes
model-based development is of strategic importance to realize continuous improvement ambition.

we (R&D) don’t make documents/machines, we make models (i.e. production specs)!
Isles → Maintainable & reusable modeling

- Results

- Effort

- Work / functionality complexity

- Document-based “in your head” way of working

- Combining models → exponential added value!

- MBD

- Want to be here!

- Throwaway models...

- Reusable model
There are already a lot of MBD success stories and it is not enough!

“faster, better, cheaper”

But we need more...
Many times a bit = a lot!

- Instead of being too specialistic... (e.g. solving complex problems in an optimal way that affect only a few people on the floor)
- Focus on issues that affect everyone.

Everyone understands that
Make the modeling users into developers

• Instead of being too narrow...
  (e.g. just a selected group of people is able and willing to do it)

• They come with improvements and new ideas

Supports the current way of working

model simulation
test generation
support other languages
...
model interfaces
capabilities
Next steps in modeling

• “Modeling islands” established
• Challenge to scale up modeling, make it “boring”:
  • Accessibility of models & tools for “everyone”
  • Proper interfacing and lifecycle management of models
  • Connecting models leads to exponential value
Example: Controller Software Modeling

- Instead of being too isolated...
  (e.g. models are developed in isolation)

"faster, better, cheaper"

- State Machines
- Software Architecture
- Interfaces
- Specs

- Impact analysis
- Less errors
- Better quality
- Consistency
- Correctness by specification
- Faster development
- Separation of concerns
- Verification
- Better communication
Example: Engine Specification Modeling
Example: Engine System-simulation

Image

Print Mode

Integration of models for simulation

Virtual Print

Print Quality
software discipline enables model-based development for all disciplines

develop your development environment
Detailed Example for Engine Specification Modeling
Mechanics/Software interface
Mechanics/Software interface and media handling
What’s it for?
Mechanics/Software interface (layout)
Layout + routes

Diagram:

- **Route Simplex =** [Input - process - output]
- **Sectors**
  - **Sector input:** C1, C2
  - **Sector process:** C2 (C3_C2_1), C3, C4
  - **Sector return:** C4, C6, C11, C7, C11, C8, C2
  - **Sector output:** C4, C10
- **Entries**
  - **entry c1 connector:** C1
- **Destinations**
  - **destination c10 connector:** C10
- **First turn point:**
  - **Turn point tolerance (amount of extra segment after a turn point):** 0 mm

```
Sector input : C1, C2

Route Simplex = (Input - process - output)

Sectors:
- Sector input: C1, C2
- Sector process: C2 (C3_C2_1), C3, C4
- Sector return: C4, C6, C11, C7, C11, C8, C2
- Sector output: C4, C10

Entries:
- entry c1 connector: C1

Destinations:
- destination c10 connector: C10

First turn point:
- Turn point tolerance (amount of extra segment after a turn point): 0 mm
```
Job specifications, media catalog, timing behavior

Job Specification
Varioprint-i300-Schedule for DL908490Mitsubishi x 70:
Orient sheet with LongEdgeFirst
Follow route Duplex
Adhere to timing behavior Varioprint-i300-TimingBehavior
Use scheduler MONOTONIC with period 0.9 s (period random: 回, then period is the max. period)
Use profile recipe: VP-i300-ProfileRecipe

Timing Behavior
Varioprint-i300-TimingBehavior
/*Transport (<< ... >>)*/
BeltCrossing (speed vNom1 = 0 mm/s, speed vNom2 = 0 mm/s)
XCorrection (speed vIn = 0 mm/s, speed vCorr = 0 mm/s, speed vOut = 0 mm/s)
BufferInterweave (<< ... >>)
Turn (speed vInitial = 0 mm/s, speed vMax = 1 mm/s)

Action
Turn (speed vInitial, speed vMax) using <choose optional parameter calculation model>
{
Entered with speed vInitial
Speed is vMax with LeadingEdge starting at TMJOIN_InterfacePoint2598_25_SENSHMSRSE - 5 mm
Stopped with TrailingEdge at InterfacePoint1225_InterfacePoint1364_FIXED2URUSRPI + 10 mm and waited for 0.03 s
Flip Edges
Accelerated to speed vMax
Speed is vInitial with LeadingEdge starting at SCOOLSCEUERRORFLAP_InterfacePoint1588_16_SCOOLSCEUFLIP1 mm during 0 s
}

Media Catalog
A4
Width: 2100 thenths of mm
Height: 2970 thenths of mm
Weight: 80 g/mm²
Color: White
DL908490Mitsubishi
Width: 2100 thenths of mm
Height: 2970 thenths of mm
Weight: 90 g/mm²
Color: White
MagnoMatt15Sappi
Width: 2100 thenths of mm
Height: 2970 thenths of mm
Weight: 115 g/mm²
Color: White
KernelF with math notation, unit type tags, interpreter

```
library OceUnits

unit mm := mm for millimeter
conversion m -> mm {
    val as number -> val / 1000
}
unit mm/s^2 := mm*s^-2 for acceleration

```

timing behavior VPi300Timing

```
import: OceUnits

job: Varioprint-1300-Schedule

transportfunction xportToPrintBelt(velocity: number, velocity2: number, acceleration: number | start: FIXDEXUSXLSE end: FIXHTURSE3) {
    val z = BYPI4 + 50 * \sqrt{\frac{velocity^2 - velocity2^2}{2 * acceleration}}
    accelerate (vStart: z vEnd: 200 mm/s acc: 10 mm/s^2 t: 20 s s: 22 mm): get sheet to printing speed
}
transportfunction xportToCooling( | start: POMIN - 1 end: POMOUT + 10) {
    xportToPrintBelt (10, 20, 30)
```
Job Specification Varioprint-1300-Schedule for DL508489Mitsubishi x 70:
Orient sheet with LongEdgeFirst
Follow route Duplex
Adhere to timing behavior Varioprint-1300-TimingBehavior
Use scheduler MONOTONIC with period 0.9 s (period random: 0, then period is the max. period)

Use profile recipe: VP-1300-ProfileRecipe

double decelerationDistance(double vPrev, double vGoal, double a) {
  return vPrev * vGoal / (2 * a);
}

decelerationDistance (function)

Action Turn (speed vInitial, speed vMin) using <choose optimal parameter calculation model>

Enters with speed vInitial
Speed is vMin with leadingedge starting at InterfacePoint1107 InterfacePoint1155 InterfacePoint1103 InterfacePoint1157 InterfacePoint1109 InterfacePoint1153 InterfacePoint1109 InterfacePoint1153 InterfacePoint1109
Stopped with trailingedge at InterfacePoint1229 InterfacePoint1229 InterfacePoint1229 InterfacePoint1229 InterfacePoint1229 InterfacePoint1229 InterfacePoint1229
Anceled to speed vMin

Route list RouteSpec for ExamplePath:

Route Simple = ( [input - process - output] )

Sectors
Sector input : C1, C2
Sector process : C2 (C1_C2), C3, C4
Sector return : C4, C6, C11, C7, C11, C8, C2
Sector output : C4, C12

Entities
entry c1 connector : C1

Destinations
destination c10 connector : C10

First turn point:
Turn point tolerance (amount of extra segment after a turn point)
0 mm

Layout ExamplePath:

R4 - Public Timing DSLs
Variant management
Variants & modular design
Variants & functional decomposition
# Variants & cost price

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Code Number</th>
<th>Cost</th>
<th>Currency</th>
<th>Minimum Quantity</th>
<th>Nr of Circuits</th>
<th>Nr of Wires</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Optical sensor</td>
<td>Sensor</td>
<td>133773311</td>
<td>10</td>
<td>USD</td>
<td>250</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2 Temperature sensor</td>
<td>Sensor</td>
<td>133773312</td>
<td>5</td>
<td>USD</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3 Pressure sensor</td>
<td>Sensor</td>
<td>133773313</td>
<td>100</td>
<td>USD</td>
<td>28</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>4 Distance sensor</td>
<td>Sensor</td>
<td>133773314</td>
<td>80</td>
<td>USD</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5 Solenoid</td>
<td>Actuator</td>
<td>133773315</td>
<td>5</td>
<td>USD</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>6 Stepper motor</td>
<td>Actuator</td>
<td>133773316</td>
<td>10</td>
<td>USD</td>
<td>3480</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>7 BLDC motor A</td>
<td>Actuator</td>
<td>133773317</td>
<td>25</td>
<td>USD</td>
<td>500</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>8 BLDC motor B</td>
<td>Actuator</td>
<td>133773318</td>
<td>20</td>
<td>USD</td>
<td>5</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>9 BLDC motor C</td>
<td>Actuator</td>
<td>133773319</td>
<td>50</td>
<td>USD</td>
<td>10</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>10 Heater</td>
<td>Actuator</td>
<td>133773310</td>
<td>15</td>
<td>USD</td>
<td>0</td>
<td>2</td>
<td>10</td>
</tr>
</tbody>
</table>
Challenges

• Cross-disciplinary consistency and model lifecycle management
• Consistent co-evolution
• Instantiation of Virtual/Hybrid Prototypes
• Model Governance
• Automated Abstraction
• Horizontal & Vertical Traceability
• Model-Based Product-Line Test & Accreditation
• Product lines and variants
• Design space exploration
• Functional & System KPI Interactions
• Semantics & Thinking Framework for Model Interoperability & Lifecycle Management
• Usability Challenges
  ► Modelling for many roles (engineers, designers, architects)
• Democratization Challenges
  ► Teaching, Selling, Attitude, Value Propositions
• Practical (re-)use of simulation results
  ► Loading/caching, EDGE, HPC
• Analysis of System KPIs
  ► Formal analysis, simulation, test

• Model Lifecycle Management, Way of working, and Organization
• Requires (partial) domain-specific interpretation of models

• Parameter Extraction
  ► Simulations/Experiments Conclusions for Higher-Abstraction Calculations/Simulations
• Domain modelling
  ► Analysis, method, process
• Model-based development process
  ► When to use what models for which purpose?
• Requirements modelling
  ► Focus on ‘What’ not on ‘How’
• Design rationale tracking
• Connecting Models with Data-Science and (Machine) Learning
  ► (Domain) Models for Rich Interpretation of Data
• Model life-cycle management as link between requirements, virtual prototyping and multi-disciplinary verification / validation / test
• Structural Organizational Embedding
  ► Bottom-up: Unobtrusive, Effective, and Scalable Tools
  ► Top-down: Structure, Shared Values, Skills, Staff, Style
Conclusions

• Océ is heavily applying MBD: we don’t make software, we make models!
• Complex, holistic field: many open challenges still exist
• Complete (commercial/open source) solutions are not available
• Research is direly needed