From Design-Time to Runtime and Back Again with Liquid Models

Manuel Wimmer
manuel.wimmer@jku.at

Institute of Business Informatics - Software Engineering
https://www.se.jku.at

Christian Doppler Laboratory (CDL-MINT)
https://cdl-mint.se.jku.ac.at
JKU Linz
WE’RE HIRING!
Why Design-Time AND RUNTIME?
Software & Systems Engineering

Design-Time versus Runtime

Design Model

Usage

Implementation
Revisiting Model-Driven Engineering

A model is considered **prescriptive** when it serves the **creation** of an object based on the model.

A model is considered **descriptive** when it serves the **description** of an existing object.

In: Tagungsband der Modellierungskonferenz, pp. 7-22, 2002
Example: Cyber-Physical Systems

Real System

Digital Representation
Engineering Viewpoint

- Analysis
- Design
- Simulate, Test, Optimize
- Operations

Design-time & Runtime

observe & adapt

Models → Data
Operational Viewpoint

EDI & Web-Portale

ERP

MES

SCADA

SPS

I/O

Office Floor

Managed File Transfer

CAD

CAM

Enterprise Bus

Shop Floor
Liquid Models
Towards a Platform for Liquid Models

Christian Doppler Laboratory
Model-Integrated Smart Production (CDL-MINT)

www.cdl-mint.se.jku.at

Liquid Model Architecture

Model-driven Perspective

- DSL
- Model
- Code Generator
- Artifact
- Execution Platform

Data-driven Perspective

- Runtime Language
- Runtime Models
- Process Miner
- Sequence Pattern Miner
- (Hidden) Markov Model Miner
- Time Series Miner
- ... (Additional components)

Runtime includes: Past | Present | Future

Mazak, Wimmer, Patsuk-Bös: Execution-Based Model Profiling.
In: Proc. of Int. Symp. on Data-Driven Process Discovery and Analysis., pp. 37–52, 2018
A Tour on Selected Research Topics
Selected Research Topics

#1: Observing Ops with Models

#2: Representing Ops as Models

#3: Linking Ops with Design

#4: Inspecting Ops via Design
#1: Observing Ops with Models
+ #2: Representing Ops as Models
Problem Statement, Challenge, and Contribution

Problem: *Discrete* design vs. *continuous* operation
- Systems do not switch in a time discrete manner between states
- Variables are continuously evolving to intended values of next states

Challenge: *State identification*
- Transform specific value configurations to states
- Precision of system realization
- Measurement uncertainty

Contribution: Model-Driven Runtime State identification (MD-RISE)
- Transforms values streams into event streams
- Allows to explore a system by hypothesis testing
Example System

Design Model

**Structure**

<table>
<thead>
<tr>
<th>BDD System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grip-arm robot</td>
</tr>
<tr>
<td>BasePosition (BP): Float = 0.0 ± 0.1</td>
</tr>
<tr>
<td>MainArmPosition (MAP): Float = 0.0 ± 0.1</td>
</tr>
<tr>
<td>GripperPosition (GP): Float = 0.0 ± 0.1</td>
</tr>
</tbody>
</table>

**Behavior**

SM Grip-arm robot

- **DriveDown**
  - BP ← 0.0
  - MAP ← 1.50
  - GP ← 1.50

- **PickUp**
  - BP ← 0.0
  - MAP ← 1.50
  - GP ← -0.40

**Runtime Data**

Sensor Values

Graph showing sensor values for DriveDown and PickUp.
State Identification: IR Problem?
MD-RISE Architecture

Evaluation Results

**RQ1 – Precision**
- Correct recognition of states depends on
  - Tolerance range
  - Distinctness of states
  - Number of sensor

**RQ2 - Recall**
- Completeness of the states identification depends on
  - Tolerance range

**RQ3 – Performance**
- Influenced by
  - Number of data records
  - Number of sensors
Multi-View Digital Shadow Platform

View 1: Time-Series (Digital Shadow)

InfluxDB

View 2: Automation (Logical View)

Backend Kotlin

View 3: Animation (Physical View)

blender

MOTT

System

#3: Linking Ops with Design

#4: Inspecting Ops via Design
Problem Statement, Challenge, and Contribution

Problem: Text-based traces for collecting logging messages
  ◦ Long, difficult, often unstructured
  ◦ Reasoning and analyzing on isolated text files is limited

Challenge: Provide *semantically* enhanced logs
  ◦ Align runtime data to design models
  ◦ Define analysis through design models

Contribution: *Temporal* models
  ◦ Introduce runtime history viewpoints in modeling languages
  ◦ Define runtime analysis on model level as queries or derived properties
## Example Systems

### System 1: PiCar

#### Design Model

![PiCar Design Model](image)

#### Runtime Data

<table>
<thead>
<tr>
<th>caseID</th>
<th>timestamp</th>
<th>Sender</th>
<th>Receiver</th>
<th>Message</th>
<th>ParameterValue</th>
<th>kind</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.914</td>
<td>Car</td>
<td>Car</td>
<td>InitAll</td>
<td>none</td>
<td>REQ</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.917</td>
<td>Car</td>
<td>DistanceSensors</td>
<td>InitSensors</td>
<td>none</td>
<td>REQ</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.928</td>
<td>DistanceSensors</td>
<td>Car</td>
<td>InitReader</td>
<td>none</td>
<td>RES</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.934</td>
<td>Car</td>
<td>MotorControl</td>
<td>InitializeMotor</td>
<td>none</td>
<td>REQ</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.950</td>
<td>MotorControl</td>
<td>Car</td>
<td>InitializeMotor</td>
<td>none</td>
<td>RES</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.954</td>
<td>Car</td>
<td>ServoControl</td>
<td>InitializeServo</td>
<td>none</td>
<td>REQ</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.964</td>
<td>ServoControl</td>
<td>Car</td>
<td>InitializeServo</td>
<td>none</td>
<td>RES</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.991</td>
<td>Car</td>
<td>Car</td>
<td>SteerStraight</td>
<td>none</td>
<td>REQ</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:13.992</td>
<td>Car</td>
<td>ServoControl</td>
<td>SteerTo</td>
<td>direction=7</td>
<td>REQ</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:15.145</td>
<td>ServoControl</td>
<td>Car</td>
<td>SteerTo</td>
<td>direction=7</td>
<td>RES</td>
</tr>
<tr>
<td>1</td>
<td>2017-02-27 17:38:15.147</td>
<td>Car</td>
<td>Car</td>
<td>SteerStraight</td>
<td>none</td>
<td>RES</td>
</tr>
</tbody>
</table>

### System 2: IAF Plant

#### Design Model

![IAF Plant Design Model](image)

#### Runtime Data

```
#Fields: component, timestamp, item entered
- /IAF/a2/t1, 2017-02-08-23-28-51, 923b4ff191d5
- /IAF/a2/t1, 2017-02-08-23-28-54, 923b4ff191d5
- /IAF/a2/t1, 2017-02-08-23-28-57, 83e51507cf2f
- /IAF/a2/t2, 2017-02-08-23-28-61, 923b4ff191d5
- /IAF/a2/t1, 2017-02-08-23-28-63, 83e51507cf2f
- /IAF/a2/t1, 2017-02-08-23-28-69, 923b4ff191d5
- /IAF/a2/t1, 2017-02-08-23-28-74, 5b73647b66d4
```

---
Capturing Runtime in UML and SysML

- UML/SysML already provide runtime viewpoints
  - Object diagram, interaction diagrams, ...
- Injecting logs into UML/SysML models as linked elements
- Profiles for computing aggregated runtime information

EA Sequence Miner

• Execution logs as UML Sequence Diagrams
  ◦ Communication between components via messages
  ◦ Standard UML
  ◦ Play-in/play-out sequences
  ◦ Filters, queries, and transformations

• Realized episodes

<table>
<thead>
<tr>
<th>MotorControl</th>
<th>ServoControl</th>
</tr>
</thead>
<tbody>
<tr>
<td>drive_forward (f)</td>
<td>steer_left (l)</td>
</tr>
<tr>
<td>drive_backward (b)</td>
<td>steer_straight (s)</td>
</tr>
<tr>
<td>stop (s)</td>
<td>steer_right (r)</td>
</tr>
</tbody>
</table>
Capturing Runtime in AutomationML

- AutomationML lacks runtime viewpoints
- But allows language inherent extensions
- Maps to standardized operational frameworks

Mazak, Lüder, Wolny, Wimmer, Winkler, Kirchheim, Rosendahl, Bayanifar, Biffi: 
*Model-based generation of run-time data collection systems exploiting AutomationML.*
Capturing Runtime in any DSL

Find the components which had an item assigned at a particular point in time

 TObject::eGetAt(i:instant, f:feature)

Joint work with:

Conclusion & Outlook
A model is considered **prescriptive** when it serves the **creation** of an object based on the model.

A model is considered **descriptive** when it serves the **description** of an existing object.
... to Digital Twins

A model is considered **prescriptive** when it serves the **creation** of an object based on the model.

A model is considered **descriptive** when it serves the **description** of an existing object.

Engineers
Operators?
Commissioners?
Users?
Digital Twin Engineering

• **Digital Model**
  ◦ Documentation & communication
  ◦ Simulation & code generation
  ◦ Design-space exploration
  ◦ Commissioning

• **Digital Shadow**
  ◦ State inspection
  ◦ Runtime monitoring
  ◦ Predictive reasoning
  ◦ Conformance checking

• **Digital Twin**
  ◦ Runtime adaptation
  ◦ Live updates & rollbacks
  ◦ Decision making
  ◦ Autonomy
Emerging Digital Twin Domains (1/4)

Smart Production
Emerging Digital Twin Domains (2/4)

Smart Buildings

Projekt Virtuelle Flughafen Stadt

Object Information
Building Type: Technical Laboratory
ID: #456123

Max. Electrical Utilization xx % (Date, Time)
Max. Thermal Utilization xx % (Date, Time)

Schöberl & Pöll GmbH
Emerging Digital Twin Domains (3/4)

Smart Cities

Amaravati, the new capital of the Indian state of Andhra Pradesh, is thought to be the first entire city born with a digital twin.

Emerging Digital Twin Domains (4/4)

Smart Enterprise

https://industrie.de/top-list/software-ag-stellt-enterprise-digital-twin-framework-vor/

Thank you!
Comments? Questions? Feedback?

Manuel Wimmer
manuel.wimmer@jku.at

Institute of Business Informatics - Software Engineering
https://www.se.jku.at

Christian Doppler Laboratory (CDL-MINT)
https://cdl-mint.se.jku.ac.at
Publications (1/2)

- Wolny, Mazak, Wimmer: Automatic Reverse Engineering of Interaction Models from System Logs. ETFA 2019
- Wolny, Mazak, Wimmer, Huemer: Model-driven Runtime State Identification. EMISA 2019
Publications (2/2)