Dealing with viewpoint interrelations - a necessary means for efficient systems engineering

Martin Törngren
Professor in Embedded Control Systems, Mechatronics at KTH – Royal Institute of Technology

Joint work with Jad Elkhoury, Frederic Loiret, Magnus Persson, Ahsan Qamar, Mattias Biehl, KTH, and Stavros Tripakis, Patricia Derler and Edward Lee, UC Berkeley

Support from ARTEMIS/VINNOVA is acknowledged
The ”4th industrial revolution”
Connectivity and Convergence

Social media
Internet
Telecom
Wired
Wireless
Computation/Software
Analog & digital


WWW
Mobile internet
Facebook
GSM
GPS
Google
iPhone
RFID
NFC
WLAN
Zigbee
WirelessHART
Bluetooth

Wired
Wireless
Internet

Social
media
Computation/Software
Analog & digital
Telecom

CAN
Ethernet

Distributed computing
Ubiquitous computing
Smart dust
Internet of Things
Cyber-physical system
Cloud computing
The swarm

Martin Törngren; May 26, Stockholm

Viewpoints and relationships

Sensing, estimation and control
Vehicle dynamics
Project Management
Quality
Maintenance, After-market
Production/parameterization
Diagnostics, Support tools
Mechanical components
Software
ECU/electronics

Architecture
Networking
CAN database
Integration tests

Safety
Power Propulsion
• Thermal

Martin Törngren; May 26, Stockholm
Main messages

CPS come with growing functionality, extent and concerns
• More advanced engineering environments:
  – viewpoints and relations between them
• We need to engineer and deal with viewpoint systems
  – Viewpoint contracts
  – Dependency modeling
  – Systematic and efficient tool integration

Outline

Cyber-Physical Systems
  – Implications - complexity and engineering environments

Viewpoint interrelations and MBSE
  – Techniques for dealing with viewpoint interrelations
  – Perspectives and discussion

Conclusions
Mechanics vs. Mechatronics; adding flexible information processing and flow.

"Purely" mechanical vehicle

<table>
<thead>
<tr>
<th></th>
<th>Sus/C</th>
<th>Brake</th>
<th>Steer</th>
<th>Wheel</th>
<th>Diff</th>
<th>Trans</th>
<th>Clutch</th>
<th>Eng</th>
<th>Driver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sus</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brake</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Steer</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wheel</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diff</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trans</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clutch</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eng</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Driver</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

X – Mechanical relations
Fully programmable vehicle!

<table>
<thead>
<tr>
<th></th>
<th>Sus/C</th>
<th>Brake</th>
<th>Steer</th>
<th>Wheel</th>
<th>Diff</th>
<th>Trans</th>
<th>Clutch</th>
<th>Eng</th>
<th>Driver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Susp</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
</tr>
<tr>
<td>Brake</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
</tr>
<tr>
<td>Steer</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
</tr>
<tr>
<td>Wheel</td>
<td>X</td>
<td>X</td>
<td>X+P</td>
<td>X</td>
<td>P</td>
<td>X</td>
<td>P</td>
<td>P</td>
<td>X</td>
</tr>
<tr>
<td>Diff</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Trans</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Clutch</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Eng</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Driver</td>
<td>X+P</td>
<td>X+P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>X+P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
</tbody>
</table>

P – Programmable relations  
X – Possible change

State space complexity  
- illustration

Assume an embedded system with
• 8 bit variables
• 5 key variables per ECU (embedded computer)
• 5 ECUs
⇒ 10^60 states  
⇒ ≈ 3 milion billion billion billion billion years!  
for complete testing
(assuming 0.1 nanosecond per test, billion = 10^9)
### Stakeholder illustration

<table>
<thead>
<tr>
<th>Stakeholder/role</th>
<th>Concerns</th>
<th>Analysis/Synthesis</th>
<th>Model characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical engineer</td>
<td>ECU interfaces, EMC</td>
<td>Electrical load, tests</td>
<td>Logic, continuous, FEM</td>
</tr>
<tr>
<td>Software engineer</td>
<td>Logics of functionality</td>
<td>Simulation of behaviour</td>
<td>Discrete-event</td>
</tr>
<tr>
<td>Quality engineer</td>
<td>Reliability, availability</td>
<td>Life-time, FMEA</td>
<td>Stochastic, logic</td>
</tr>
<tr>
<td>Mechanical Engineer</td>
<td>Geometry, fitting</td>
<td>Cable length, alignment</td>
<td>2D &amp; 3D mechanical</td>
</tr>
<tr>
<td>Thermo analyst</td>
<td>Temperature</td>
<td>Heat transfer/dissipation</td>
<td>FEM</td>
</tr>
<tr>
<td>Safety engineer</td>
<td>System safety</td>
<td>FTA, FMEA</td>
<td>Logic, DE, stochastic</td>
</tr>
<tr>
<td>Control engineer</td>
<td>Noise, disturbances,</td>
<td>Simulation, stability,</td>
<td>Continuous-time,</td>
</tr>
<tr>
<td></td>
<td>performance, robustness</td>
<td>reachability, synthesis</td>
<td>discrete-time and DE</td>
</tr>
<tr>
<td>Integration engineer</td>
<td>Communication and</td>
<td>Testing, automation, report generation</td>
<td>DE, test cases, structure</td>
</tr>
<tr>
<td></td>
<td>distributed functions</td>
<td></td>
<td>(configurations)</td>
</tr>
</tbody>
</table>

**Implications:** More interrelations!

---

Martin Törngren; May 26, Stockholm
Need to deal with viewpoint interrelations

A prerequisite for efficient and effective concurrent engineering:

– Understanding when "synchronization" is necessary
  • "inconsistency management"
– Impact, trade-offs, optimization
– Suitable information management and access
– Automation
– Better planned and designed engineering environments

Outline

Part I: Cyber-Physical Systems
  – The 4th industrial "revolution" and the connected society
  – Implications - complexity and engineering environments

Part II: Viewpoint interrelations and MBSE
  – Techniques for viewpoint interrelations
  – Perspectives and discussion

Conclusions
Viewpoints and views

Terminology partly from the ISO/IEEE 42010 standard: Systems and software engineering — Architecture description

“Viewpoints”
- Stakeholder concerns
- Concepts, theories, frameworks, languages

Models/Artefacts – “Views”
Product properties

Tools – using models and providing services

The viewpoint is where you look from

Multi-level approach for dealing with viewpoint interrelations

Viewpoint 1
- People Level
- Models Level
- Tools Level
- Process
- Product

Viewpoint 2
- People Level
- Models Level
- Tools Level
- Process
- Product

Martin Törngren; May 26, Stockholm
Example:
Controller vs. Embedded system design

- Little emphasis on effects due to implementation
- Delays, quantization, partial failures, resource sharing

Concepts, theory and tools mismatches, e.g. Discrete-time control theory vs. Fixed priority scheduling

Controller reuse with "violated contract"

DC motor feedback control:
Assume controller is designed to compensate for nominal one sampling period delay
Example – view dependencies

Variety of tools
Challenges of Tool Integration

Tools provide limited native integration
Difficult to connect fragmented “islands”
  - Data, tracing, scripts/control, presentation
  - Standardization challenge

Model-based approaches for dealing with viewpoint interrelations

- **Viewpoint contracts**
  - Stakeholder, concept and theory level
  - Establishing shared concepts among two or more viewpoints, with assumptions and constraints

- **Dependency models**
  - Model level
  - Captures dependencies among properties of models

- **Tool integration models**
  - Describes tools and their interactions

Scenario basis (design stages, aspects, models, tools, standards)
Example contracts:
Control-Embedded SW with timing constraints

Agreement and obligations regarding functionalities and timing properties
• SW engineers: execute functions; meet timing requirements
• Control engineers: ensure correct closed-loop behavior

Example contracts:
- 'ZET' ~ the synchronous approach
- 'LET' ~ the PLC / Giotto approach
- 'BET' ~ interpretation of FPS
- 'DET' ~ deadline monotonic scheduling
- 'TOL' ~ Tolerances on time variations

Basis for communication and agreements
Targets specific scenarios
Support for modeling and simulation

Example: Control-Embedded SW contracts

Mealy machine $M$ (functional part)

Sampling function: read sensors and assign values to inputs

Inputs (set of input variables)

State (set of state variables)

Update function: compute new state from input and state

Outputs (set of output variables)

Output function: compute output from input and state

Actuation function: write outputs to actuators

Courtesy of P. Dehter, Design contracts presentation at CPS conference 2013
Example contracts – Control vs. Embedded SW with timing constraints

Support for Contract based design-methodology outline
Two Degree-Of-Freedom Robot

- Control the position of the robot with accuracy (CPA) and avoid obstacle within the specified workspace (WS)
- Two design domains
  - Mechanical design
  - Control design
  - Hw/Sw design
- Design variables
  - L_A, L_B
  - W_A, W_B
  - θ_A, θ_B
  - g
  - S_A, S_B
  - M_A, M_B
  - Origin ‘O’
  - PE

Dependency model example
Snapshot of the dependency graph for Mechanical, Control and Hw/Sw design of the robot

Highlighting selected interrelations
Analysis made possible

Change management
Consistency checking

Goal: Tool integration

Tool Integration Framework
- OSLC

iFEST ARTEMIS project

Martin Törngren; May 26, Stockholm
Approach

Development Support:
- Design/Refinement
- Analysis
- Synthesis

Stakeholders

Modeling Language

Tool Chain Implementation

TIL Language Concepts

<table>
<thead>
<tr>
<th>No</th>
<th>Concept</th>
<th>No</th>
<th>Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Tool Chain</td>
<td>4</td>
<td>User</td>
</tr>
<tr>
<td>1</td>
<td>Tool Adapter</td>
<td>5</td>
<td>DataChannel</td>
</tr>
<tr>
<td>2</td>
<td>ControlChannel</td>
<td>6</td>
<td>TraceChannel</td>
</tr>
<tr>
<td>3</td>
<td>Sequencer</td>
<td>7</td>
<td>Repository</td>
</tr>
</tbody>
</table>

Martin Törngren; May 26, Stockholm
Simple example tool chain model

Case Study: A TIL Model for an industrial embedded systems tool chain
Perspectives and discussion

- Experiences
- Complexity management
- Model-based engineering
- Other integration techniques

Status

- Several papers, case studies and PhD thesis on the presented techniques (references enclosed)
- Dependency modeling and Tool integration implemented as DSLs
- Viewpoint contracts applied to Control-Embedded systems and Control-Mechanical design
How do we manage complexity when developing technical systems?

• Divide and conquer
  – Separation of concerns
  – Applied to products and systems (supporting artefacts, organizations)

• Abstraction
  – Modeling, at the “right” level of abstraction using the “right” formalism

• Ensuring integration / composability
  – Interfaces, interrelationships, assumptions

• By use of automation ("tools")

---

Model-based engineering

• Risk reduction technique

• Communication, analysis, synthesis
  – Modeling and simulation
  – Multiple aspects/questions → Lots of data & models
Discussion

• Applicability of “viewpoint integration” techniques:
  – Informally / Formally
  – Individually / Together
  – Top-down / Bottom-up

• Complement other techniques such as
  – Co-simulation
  – Integration specific views (e.g. ADLs, function models)

• Relevance beyond CPS for complex systems

Contrasting techniques

• Integration specific views
• (Co-) simulation and optimization
• Multiview frameworks
• Model-driven engineering
  • Consistency in multiview modeling
• Tool integration
• Other techniques
  – Organizational integration
  – Processes synchronization
Conclusions

Part I: Cyber-Physical Systems - connectivity
– Implications on complexity and engineering environments

Part II: Viewpoint interrelations and MBSE
– Multiview modeling and interrelations!
– Model based approaches for dealing with interrelations
  • Making assumptions, dependencies and interactions explicit
  • Viewpoint contracts; Dependency modeling; Tool integration
  • Complementary to existing techniques

Future:
– 2nd generation DLS’s, methodology and industrial evaluation

References


Martin Törngren; May 26, Stockholm