Addressing Accidental Complexities in Modelling Languages

Jason Balaci, Dr. Jacques Carette, Dr. Spencer Smith
Department of Computing and Software, McMaster University

Background

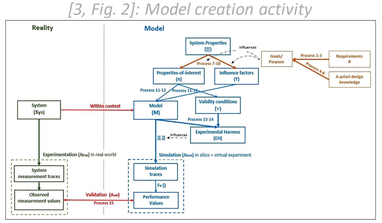
Model-Driven Engineering (MDE): Industry-adopted, successful software development paradigm that emphasizes *models* (domain-specific knowledge) as the primary object of development efforts [1].

Context

More **features**!

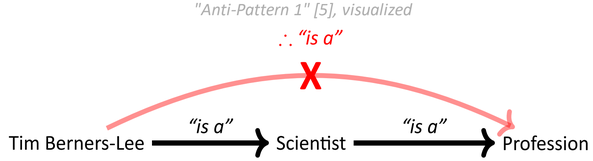
1. Validity Frames [2]: model-validity conditions for improved model sharing and appropriate application, and capturing model influence factors, validity-checking procedures, experimental frames, and assisted calibration procedures. Good for Cyber-Physical System (CPS) modelling.

[3, Fig. 2]: Model creation activity



2. Multi-level-supported Modelling [4]: permit *n*-levels (instead of just 2) to avoid inconsistencies; add deeper abstractions without incurring runtime performance hits, code duplication, complicated maintenance, or delicate features.

"Anti-Pattern 1" [5], visualized



3. *TODO: Diagram types, model views, etc.*

Problem

Accidental complexities!

Extrinsic issues (i.e., from tool rather than task), e.g., excessive code boilerplate, "edge cases," non-intuitive syntax, problematic semantics (inconsistencies, ambiguities, etc.), etc. [6].

Discussion

What is the state of practice for implementing them?

1. Validity Frames
[3] demonstrate a proof-of-concept, where:

$$VF = \langle M, S_M, \pi_M, \gamma_M, map_{\gamma \to S_M}, map_{\gamma \to S_M}, SPEC_{exe}, Val_M, \alpha_{val} \rangle$$

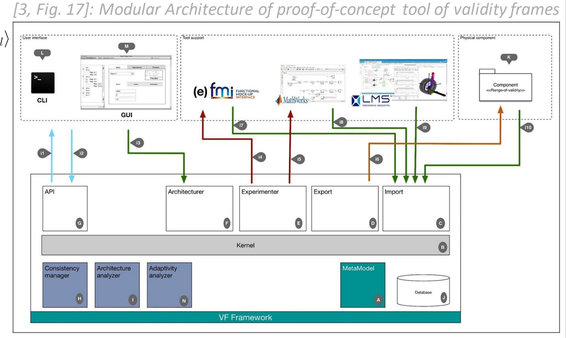
M, contained model; S_M, structural elements; π_M, properties of interest; γ_M, influence factors; $map_{\gamma \to S_M}$, property-to-structure map; $map_{\gamma \to S_M}$, influence-to-structure map; $SPEC_{exe}$, execution framework; Val_M, model fidelity measure; α_{val}, validation activities.

[3, Tab. 6]: Set of properties for the ODrive BLDC Motor

Type	ID	Name	Description	Unit	Min	Max
Property	p_201	R_p	Resistance of a single phase	Ohm	0.039	0.039
Property	p_202	L_p	Inductance of a single phase	μ F	20.2	20.2
Property	p_203	p	Number of pole pairs	–	7.0	7.0
Property	p_204	kV	Back EMF constant	RPM/V	270.0	270.0
Property	p_205	J	Inertia of the rotor	kg * m²	0.0000926	0.0000926
Property	p_206	RPM_{motor}	RPM of the BLDC motor (unidirectional)	rpm	−8000.0	8000.0
Property	p_207	d_1	Diameter of the primary shaft	mm	8.0	8.0
Property	p_208	d_2	Diameter of the secondary shaft	mm	8.0	8.0
Property	p_209	t_{motor}	Operation temperature of the BLDC motor in housing	°C	0.0	125.0
Property	p_210	I_{motor}	Actual phase current of the motor	A	−120	120

BLDC: Brushless DC; EMF: Electromagnetic field; RPM: Rotations Per Minute.

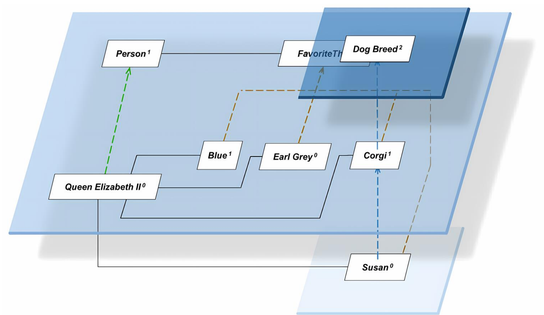
[3, Fig. 17]: Modular Architecture of proof-of-concept tool of validity frames



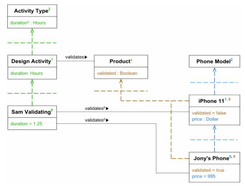
2. Multi-level-supported modelling
Several possible ways, each with their own difficulties [7]:

1. Strict meta-modelling
2. Relaxed strictness
3. Orderless typing
4. Transitive levels
5. Level blindness
6. Multi-dimensional (orthogonal ontologies), multi-level

[7, Fig. 9]: Recognizing orthogonal classification dimension



[7, Fig. 7]: Overlapping and Disjoint class. concerns



Where do we go from here?

Semantics of Modelling Languages [8]: Often neglected, leading to confusion, reasoning difficulties, and complications in extension. Need to recognize "language" as a coupling of syntax and "semantics" (semantic mapping and domain), highlighting the importance of recognizing the nature and purpose of each component [8].

Syntax Definition, Extended Backus-Naur Form

$$\begin{aligned}
\langle \text{expression} \rangle &\to \langle \text{term} \rangle \{\langle \text{term-op} \rangle \langle \text{term} \rangle\} \\
\langle \text{term-op} \rangle &\to \text{“+”} \mid \text{“−”} \\
\langle \text{term} \rangle &\to \langle \text{factor} \rangle \{\langle \text{factor-op} \rangle \langle \text{factor} \rangle\} \\
\langle \text{factor-op} \rangle &\to \text{“}*\text{”} \mid \text{“/”} \\
\langle \text{factor} \rangle &\to \langle \text{integer} \rangle \mid \text{“(”} \langle \text{expression} \rangle \text{“)”} \\
\langle \text{integer} \rangle &\to \langle \text{non-zero-digit} \rangle \{\langle \text{digit} \rangle\} \\
\langle \text{non-zero-digit} \rangle &\to \text{“1”} \mid \text{“2”} \mid \ldots \mid \text{“9”} \\
\langle \text{digit} \rangle &\to \text{“0”} \mid \langle \text{non-zero-digit} \rangle
\end{aligned}$$

Syntax Definition, Inductively

1. $\mathsf{Z} \in \mathcal{L}$
2. If $n \in \mathcal{L}$, then $\mathsf{S}\ n \in \mathcal{L}$
3. If $l, r \in \mathcal{L}$, then $\{l + r, l * r\} \subseteq \mathcal{L}$

Attribute Grammar Semantics

$$\begin{aligned}
\langle \text{expression}(w) \rangle &\to \langle \text{term}(t_h) \rangle \ll w \leftarrow t_h \gg \{\langle \text{term-op}(w_{op}) \rangle \langle \text{term}(t_t) \rangle \ll w \leftarrow \text{“(}w_{op}\ \ (w)\ \ (t_t)\text{)”} \gg\} \\
\langle \text{term-op}(w) \rangle &\to \text{“+”} \ll w \leftarrow \text{“i64.add”} \gg \mid \text{“−”} \ll w \leftarrow \text{“i64.sub”} \gg \\
\langle \text{term}(w) \rangle &\to \langle \text{factor}(f_h) \rangle \ll w \leftarrow f_h \gg \{\langle \text{factor-op}(f_{op}) \rangle \langle \text{factor}(f_t) \rangle \ll w \leftarrow \text{“(}f_{op}\ \ (w)\ \ (f_t)\text{)”} \gg\} \\
\langle \text{factor-op}(w) \rangle &\to \text{“}*\text{”} \ll w \leftarrow \text{“i64.mul”} \gg \mid \text{“/”} \ll w \leftarrow \text{“i64.div_u”} \gg \\
\langle \text{factor}(w) \rangle &\to \langle \text{integer}(i) \rangle \ll w \leftarrow \text{“}i\text{”} \gg \mid \text{“(”} \langle \text{expression}(e) \rangle \text{“)”} \ll w \leftarrow e \gg \\
\langle \text{integer}(w) \rangle &\to \langle \text{non-zero-digit}(n) \rangle \ll w \leftarrow \text{“}n\text{”} \gg \{\langle \text{digit}(d) \rangle \ll w \leftarrow \text{“}wd\text{”} \gg\} \\
\langle \text{non-zero-digit}(w) \rangle &\to \text{“1”} \ll w \leftarrow \text{“1”} \gg \mid \text{“2”} \ll w \leftarrow \text{“2”} \gg \mid \ldots \mid \text{“9”} \ll w \leftarrow \text{“9”} \gg \\
\langle \text{digit}(w) \rangle &\to \text{“0”} \ll w \leftarrow \text{“0”} \gg \mid \langle \text{non-zero-digit}(d) \rangle \ll w \leftarrow d \gg
\end{aligned}$$

Axiomatic Semantics/Hoare Logic

$$\{\text{true}\}\ \mathsf{Z}\ \{\text{result} = 0\}\ \text{z} \quad \{\text{result} = n'\}\ \mathsf{S}\ n\ \{\text{result} = n' + 1\}\ \text{s}$$

$$\frac{\{P\}\ l\ \{\text{result} = l'\} \qquad \{Q\}\ r\ \{\text{result} = r'\}}{\{R\}\ l + r\ \{\text{result} = l' + r'\}}\ \textsc{Add}.$$

$$\frac{\{P\}\ l\ \{\text{result} = l'\} \qquad \{Q\}\ r\ \{\text{result} = r'\}}{\{R\}\ l * r\ \{\text{result} = l' \cdot r'\}}\ \textsc{Mul}.$$

Denotational Semantics

$$\begin{aligned}
[\![\mathsf{Z}]\!] &:= 0 \\
[\![\mathsf{S}\ n]\!] &:= 1 + [\![n]\!] \\
[\![l + r]\!] &:= [\![l]\!] + [\![r]\!] \\
[\![l * r]\!] &:= [\![l]\!] \cdot [\![r]\!]
\end{aligned}$$

Big-Step Operational Semantics

$$\frac{}{\mathsf{Z} \Downarrow 0}\ \text{Z} \qquad \frac{n \Downarrow n'}{\mathsf{S}\ n \Downarrow 1 + n'}\ \text{S}$$

$$\frac{l \Downarrow l' \qquad r \Downarrow r'}{l + r \Downarrow l' + r'}\ \textsc{Add}. \qquad \frac{l \Downarrow l' \qquad r \Downarrow r'}{l * r \Downarrow l' \cdot r'}\ \textsc{Mul}.$$

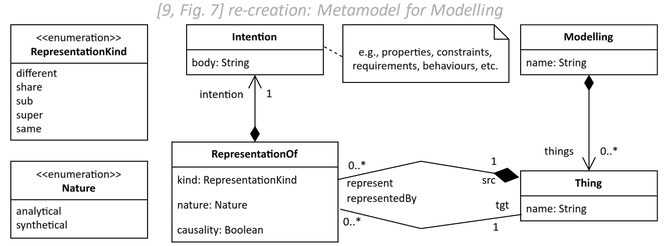
"Pertinent questions" when developing languages [8]:

1. Does the given formalization capture the users' intuition?
2. Are the context conditions sufficient to ensure that language expressions are consistent and meaningful?
3. Does the notation permit the specification of important semantic domain properties?
4. If analysis techniques or transformations for the language exist, are they sound with respect to the semantics?

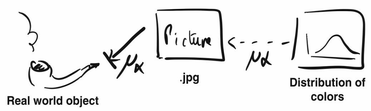
What does "modelling" really entail?

Modelling community lacks a concrete, conventional definition of "modelling" [9]. [9] reduce modelling to building representation relationships between "things" with essential components: *intention* and *nature.*

[9, Fig. 7] re-creation: Metamodel for Modelling



[9, Fig. 9], Example of Representation Relationships: This Is Not a Pipe



Open Research Questions

1. What are the requirements of (meta-/)modelling languages known today?
2. Can we apply knowledge from programming language research on formal semantics definition styles so that we can build and reason about modelling languages with clear semantics?
3. How can we develop a simplified approach to incorporate the essential features of validity frames without modelling languages (or at least CPS modelling languages) and frameworks whilst minimizing accidental complexities?

References

[1]Douglas C. Schmidt. “Guest Editor’s Introduction: Model-Driven Engineering”. In: *Computer* 39.2 (2006-02), pp. 25–31. DOI: 10.1109/MC.2006.58.

[2]Simon Van Mierlo, Bentley James Oakes, Bert Van Acker, Raheleh Eslampanah, Joachim Denil, and Hans Vangheluwe. “Exploring validity frames in practice”. In: *International Conference on Systems Modelling and Management*. Springer. 2020, pp. 131–148.

[3]Bert Van Acker, Paul De Meulenaere, Hans Vangheluwe, and Joachim Denil. “Validity Frame-enabled model-based engineering processes”. In: *SIMULATION* 100.2 (2024), pp. 185–226. DOI: 10.1177/00375497231205035.

[4]Colin Atkinson and Thomas Kühne. “Reducing accidental complexity in domain models”. In: *Software & Systems Modeling* 7.3 (2008-07), pp. 345–359. DOI: 10.1007/s10270-007-0061-0.

[5]Freddy Brasileiro, João Paulo A. Almeida, Victorio A. Carvalho, and Giancarlo Guizzardi. “Applying a Multi-Level Modeling Theory to Assess Taxonomic Hierarchies in Wikidata”. In: *Proceedings of the 25th International Conference Companion on World Wide Web*. WWW ’16 Companion. Montréal, Québec, Canada: International World Wide Web Conferences Steering Committee, 2016, pp. 975–980. DOI: 10.1145/2872518.2891117.

[6]Brooks. “No Silver Bullet Essence and Accidents of Software Engineering”. In: *Computer* 20.4 (1987), pp. 10–19. DOI: 10.1109/MC.1987.1663532.

[7]Thomas Kühne. “Multi-dimensional multi-level modeling”. In: *Software and Systems Modeling* 21 (2022-04), pp. 1–17. DOI: 10.1007/s10270-021-00951-5.

[8]David Harel and Bernhard Rumpe. “Meaningful modeling: What’s the semantics of “semantics”?” In: *Computer* 37.10 (2004), pp. 64–72.

[9]Pierre-Alain Muller, Frédéric Fondement, Benoit Baudry, and Benoit Combemale. “Modeling modeling modeling”. In: *Journal of Software and Systems Modeling* 11.3 (2012), pp. 347–359. DOI: 10.1007/s10270-010-0172-x.