Scaling CPS Understanding

From Foundations and Models to Practical Engineering

Manfred Broy (TU Munich), **Harald Ruess** (SRI)

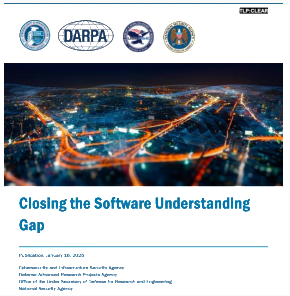
HCSS'26

Annapolis, 13th May 2026

From Software to CPS Understanding

The Software Understanding Gap: We currently lack the capacity to fully construct and assess software across all operational conditions. This gap results in critical failures:

- Inability to create secure-by-design software.
- Difficulty fixing defects once discovered.
- Failing to change SW at mission-relevant speed and scale.



The Evolution to CPS Understanding:

- Fusing digital logic directly with physical dynamics.
- Shifting the baseline to rigorously establish essential security and resilience properties.
- Enforcing stability of real-time feedback loops.

Four Pillars of CPS Understanding

1. Semantic Clarity: From Ambiguity to Certainty

Well-defined behavior for informal box-and-arrow diagrams.

2. Compositionality: The Whole is the Sum of its Parts

Architectural operations with well-defined feedback loops and modular reasoning.

3. Architectural Viewpoints: Separating Concerns to Synthesize Clarity

Orthogonal projections of complexity (Functional, Logical, Technical).

4. Semantic Traceability: Preserving Intent from Blueprint to Binary

Maintaining the unbroken chain of evidence across development steps and the lifecycle

"The tools we use have a profound and devious influence on our thinking habits, and therefore on our thinking abilities". *Dijkstra, EWD496, 1975.*

1. Semantic Clarity: Specifications as Deterministic Stream Transformers

Deterministic specifications are modeled by

$$f : \mathrm{Stream}(I) \to \mathrm{Stream}(O)$$

Example: Reactor Cooler Function

$$H_{\mathrm{off}}(x \cdot s) = \text{if } x \geq 20^\circ \text{ then ON} \cdot H_{\mathrm{on}}(s) \text{ else OFF} \cdot H_{\mathrm{off}}(s)$$
$$H_{\mathrm{on}}(x \cdot s) = \text{if } x \leq 15^\circ \text{ then OFF} \cdot H_{\mathrm{off}}(s) \text{ else ON} \cdot H_{\mathrm{on}}(s)$$

No Chattering: If $\Delta x \leq 2^\circ$ then no $\mathrm{ON} \cdot \mathrm{OFF} \cdot \mathrm{ON}$ subsequence.

Strong Causality: $x \downarrow t = z \downarrow t \implies f(x) \downarrow t+1 = f(z) \downarrow t+1$

Strongly causal functions $\simeq$ Deterministic Moore machines

1. Semantic Clarity: Specifications as Nondeterministic Stream Transformers

Non-determinism captures uncertainty, tolerances, design freedom, or requirements:

$$F : \mathrm{Stream}(I) \to \mathcal{P}(\mathrm{Stream}(O))$$

Example: Non-Deterministic Reactor Cooler

$$H_{\mathsf{off}}(x{\cdot}s) = \begin{cases} \mathsf{ON} \cdot H_{\mathsf{on}}(s) & x \geq 20 \\ \mathsf{OFF} \cdot H_{\mathsf{off}}(s) & x < 19 \\ \mathsf{ON} \cdot H_{\mathsf{on}}(s) \cup \mathsf{OFF} \cdot H_{\mathsf{off}}(s) & 19 \leq x < 20 \end{cases} \qquad H_{\mathsf{on}}(x{\cdot}s) = \begin{cases} \mathsf{OFF} \cdot H_{\mathsf{off}}(s) & x \leq 15 \\ \mathsf{ON} \cdot H_{\mathsf{on}}(s) & x > 16 \\ \mathsf{OFF} \cdot H_{\mathsf{off}}(s) \cup \mathsf{ON} \cdot H_{\mathsf{on}}(s) & 15 < x \leq 16 \end{cases}$$

No Chattering: No $\mathsf{ON} \cdot \mathsf{OFF} \cdot \mathsf{ON}$ subsequence if $\Delta x \leq 2^\circ$.

Strong Causality: $x \downarrow t = z \downarrow t \implies F(x) \downarrow t+1 = F(z) \downarrow t+1$

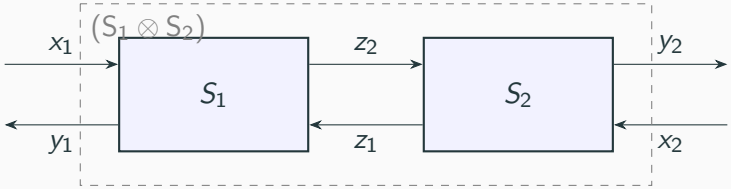
Full Realizability = Strong Causality + Deterministic Strategy for Every Valid Output

Specification-Machine Correspondence: Fully Realizable Specs $\simeq$ Gen. Moore Machines

Refinement $G \sqsubseteq F$: IO-behaviors of G are a subset of the IO-behaviors of F; e.g:

$$\mathrm{No\_Chatter} \sqsupseteq \mathrm{Cooler}^{\mathrm{nd}} \sqsupseteq \mathrm{Cooler}^{\mathrm{det}} \sqsupseteq \mathrm{Cooler}^{\mathrm{mm}}$$

2. Compositionality: Concurrent Composition with Feedback Loops



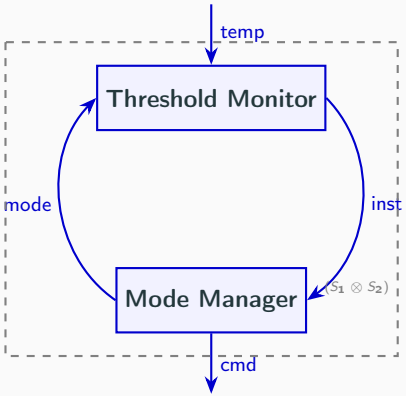
Concurrent composition $(S_1 \otimes S_2)$ is defined by conjoining the interface assertions and hiding the internal feedback channels:

$$(S_1 \otimes S_2) \triangleq \exists \text{state}, \text{inst} : S_1(x_1, z_1; y_1, z_2) \wedge S_2(x_2, z_2; y_2, z_1)$$

2. Compositionality: Implementing the Non-Deterministic Cooler

Mutually Dependent Architecture

- **Threshhold Monitor:** Takes physical temperature and current mode. Outputs a discrete instruction ($\text{inst} \in \{\text{TurnOn}, \text{TurnOff}, \text{Stay}, \text{NonDet}\}$).
- **Mode Manager:** Takes instruction (inst) to output physical command and loops back its new mode.



Concurrent Composition:

$\text{Cooler}_{impl} \triangleq \exists z_1, z_2 : (\text{Monitor} \otimes \text{Manager})$

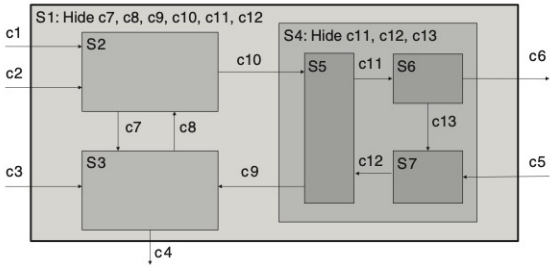
Semantic Guarantee

As long as the loop maintains *strong causality*:

$$\text{Cooler}_{impl} \sqsubseteq \text{Cooler}_{nd} \sqsubseteq \text{No\_chatter}$$

2. Compositionality: Architectures from Specifications

An architecture is composed of specifications (and architectures) in terms of $\otimes$.



$$S_1 = S_2 \otimes S_3 \otimes S_4$$

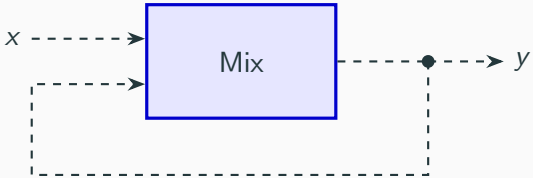
$$S_4 = S_5 \otimes S_6 \otimes S_7$$

Independent development: refining an individual component ($S_i' \sqsubseteq S_i$) maintains that refinement relationship on the architectural level.

2. Compositionality: The Nondeterministic Feedback Anomaly

The Mix Specification:

- Mix merges input streams x and z into y.
- **Logical Specification:**
  $(\forall m)\, count(m, y) = count(m, x) + count(m, z)$



Fixed-Points in Feedback:

- $count(m, y) = count(m, x) + count(m, y)$
- If $count(m, x) > 0$ then $count(m, y) = \infty$.
- There are infinitely many, computationally infeasible solutions.
- Traditional *prefix-monotonicity* is failing:
  1. $(1^\omega, \langle\rangle) \leq (1^\omega, 2^\omega)$
  2. Assume monotonicity $Mix(1^\omega, \langle\rangle) \leq Mix(1^\omega, 2^\omega)$
  3. This implies $1^\omega \leq Mix(1^\omega, 2^\omega)$, contradiction.

2. Compositionality: Unique Fixpoints and Virtual Integration

- **Strongly Causal Specifications:** We constrain the Mix model to physical reality.

$$count(m, x)|_t + count(m, z)|_t \geq count(m, y)|_{t+1}$$

- **The Baire Metric:** Streams form a Cauchy-complete ultrametric space where distance is determined by the longest common prefix.

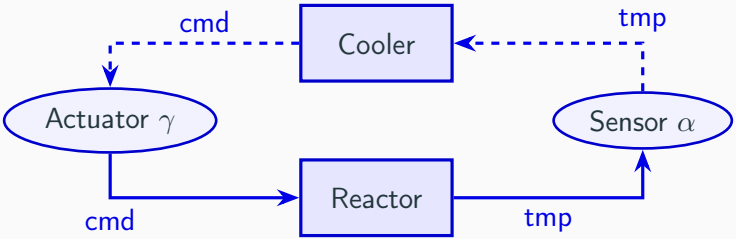
$$d(x, y) = 2^{-\inf\{t \in \mathbb{N} | x(t) \neq y(t)\}}$$

- **Strong Causality $\equiv$ Contraction:** A stream transformer is strongly causal *if and only if* it is a contractive mapping in the Baire metric space.

- **Banach's Fixed-Point Theorem:** Because the space is complete and the function is contractive, Banach's theorem implies a **unique, computationally feasible fixed-point**.

- **Concurrent composition**: if S_1 and S_2 are fully realizable, their concurrent composition $S_1 \otimes S_2$ yields a unique global behavior solely from interface specifications

2. Compositionality: Abstract Model-Checking



- **Hybrid System:** $S_{\text{sys}} \triangleq \exists \text{tmp}, \text{cmd} : \text{Cooler} \otimes (\gamma; \text{Reactor}; \alpha)$
- **Relational abstraction (HybridSAL):** $\gamma; \text{Reactor}; \alpha \sqsubseteq \text{Reactor}^{\text{abst}}$
- **Model check:** $\text{Reactor}^{\text{abst}} \otimes \text{Cooler} \models \varphi$
- **Refinement:** S_{sys} inherits safety property φ of the abstracted discrete system.

2. Compositionality: Modular Reasoning

- A reactive contract $C = (A, G)$ consists of
  - Weakly causal environment assumption $A(x, y)$
  - Strongly causal (and strongly realizable) system guarantee $G(x, y)$
- Assumptions may depend on output y because an environment's input x might only be valid or reliable for certain system responses (e.g. previous *ready* signal)
- The restriction to weakly causal $A(x, y)$ rules out that this assumption restricts outputs.
- Strict semantics: the system upholds G at step n only if A was upheld strictly *prior* to n:

$$S_C(x, y) \triangleq \forall n \in \mathbb{N} : ((\forall k < n : A(x|_k, y|_k)) \Rightarrow G(x|_n, y|_n))$$

Assume-Guarantee Reasoning (for Safety Properties):

$$\frac{\begin{array}{ll} 1. & \forall t \in \mathbb{N} : (P \downarrow t \wedge G_2 \downarrow t - 1 \sqsubseteq A_1 \downarrow t \\ 2. & \forall t \in \mathbb{N} : (P \downarrow t \wedge G_1 \downarrow t - 1 \sqsubseteq A_2 \downarrow t \\ 3. & (G_1 \wedge G_2) \sqsubseteq Q \end{array}}{P \sqsubseteq C_1 \otimes C_2 \sqsubseteq Q}$$

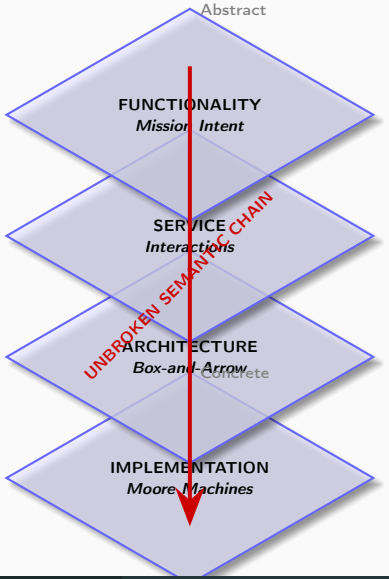
3. Viewpoints: Integrated Refinement Layers

Partitioning Cross-Domain Complexity

- **Functionality View**: The logical contract with the physical environment.
- **Service View**: Analysis of concurrent feature interactions and modes.
- **Architectural View**: The box-and-arrow blueprint of components.
- **Implementation View**: Technical evidence via Moore Machines.

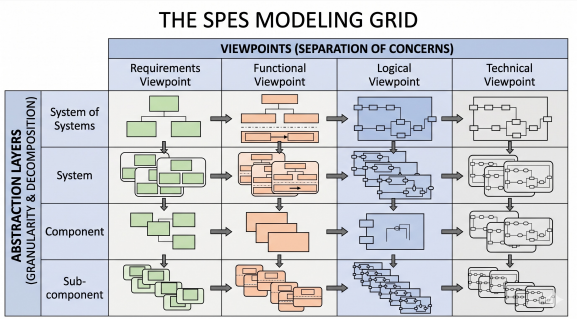
The Semantic Guarantee

Every transition is a formal **refinement** ($\sqsubseteq$), ensuring safety properties are inherited from top to bottom.



4. Semantic Traceability: Unbroken Chain of Evidence

- The SPES grid is a foundational structure used to manage the multi-domain complexity of CPS by cross-referencing four distinct mathematical viewpoints with varying levels of abstraction.
- This ensures that a change made at any level—from a high-level requirement to a sub-component's implementation—can be precisely traced and validated across the entire system.

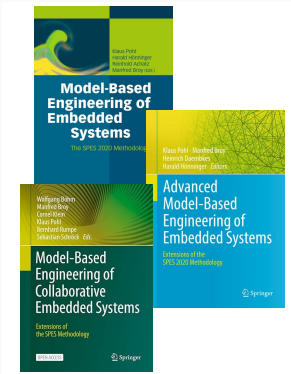


System Assurance

A formal basis for **assurance cases** where overarching system properties are backed by localized mathematical evidence of precisely defined verification conditions.

Stream Calculus in Industry (The SPES Experience)

- **Conquering Complexity:** The SPES Grid (Viewpoints vs. Granularity) successfully eliminated **spaghetti architectures** by enforcing a strict separation of concerns.
- **Cross-Domain Co-engineering:** Explicit semantic interfaces between logical and technical viewpoints provided a common modeling language for mechanical, electrical, and software teams.
- **Front-Loading V&V:** Formalizing logical architectures allowed engineers to perform virtual simulations, catching critical design flaws long before physical prototyping.
- **Practical Legacy Integration:** Industrial adoption requires accommodating existing IP; providing guidelines to wrap legacy code and documents was essential for transition.



Summary: Achieving Scalable CPS Understanding

The Shift to Rigorous Evidence

- Stream calculus is comprehensive and based on just a few integrated concepts: interface specifications, generalized Moore machines, refinement, and concurrent composition.
- It enables verifiable compositionality of CPS, coherent architectural viewpoints, and end-to-end, semantically traceable assurance across the entire engineering lifecycle.

Industrial-Scale Assurance

- An unbroken chain of evidence provides a formal basis for assurance cases, backing overarching system properties with traceable localized evidence.
- **Verify First, Generate Later:** System-of-systems behavior is predictable solely from interface contracts, catching issues well before implementation.

The AI Horizon

Stream calculus provides the **formal guardrail** for agentic AI design, enabling LLMs to generate high-assurance CPS artifacts that are formally checkable.

Literature

- Broy, M., & Ruess, H. (2026). Principled Cyber-Physical System Design: From Foundations and Models to Practical Engineering. *IEEE Computer*.
- Broy, M., Ruess, H., & Shankar, N. (2025). It's the Specification, Stupid! *Communications of the ACM*.
- Pohl, K., Hönninger, H., Achatz, R., & Broy, M. (Eds.). (2012). *Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology*. Springer Berlin, Heidelberg.
- Pohl, K., Broy, M., Daembkes, H., & Vogelsang, A. (Eds.). (2016). *Advanced Model-Based Engineering of Embedded Systems: Extensions of the SPES 2020 Methodology (SPES XT)*. Springer International Publishing.
- Böhm, W., Broy, M., Daun, M., et al. (Eds.). (2021). *Model-Based Engineering of Collaborative Embedded Systems: Extensions of the SPES Methodology*. Springer Open.