An Architecture Approach to Heterogeneous Verification of CPS

A. Bhave¹, K. Butts², D. Caveney³, D. Garlan⁴, B. Krogh¹, S. Loos², A. Platzer², A. Rajhans¹, P. Ramachandra³, B. Schmerl²
¹ECE, ²CS) Department, Carnegie Mellon University, Pittsburgh, PA. ³Toyota Technical Center, Ann Arbor, MI.

Project Goals

Models as Architectural Views

Parameters to support Heterogeneous Verification

- Parameters as the first step towards adding semantics to the architectural framework
- Parameter constraints define the valuations of the parameters and affect the system/model behavior
- Auxiliary constraint captures parameter dependencies across the system and the models

Architectural Modeling of Cyber-Physical Systems

Structural Consistency using Graph Morphisms

Weak consistency (correctness)
- Each element in the view has a corresponding element in the base, i.e., graph monomorphism

Strong consistency (completeness)
- Each element in the base has a corresponding element in the view, i.e., graph isomorphism

References


Tool Support in AcmeStudio

- AcmeStudio is a semantically extensible framework for architectural design and analysis with built-in support for styles, system structure and constraints
- View consistency plugin under development. Uses maximum common sub-graph matching algorithm at back end
- Support for parameter constraints planned

Base Architecture (left) and view (center) modeling in AcmeStudio. View consistency plug-in under development (right).

How do we
- guarantee that the models represent the actual system?
- guarantee that the models are consistent with each other?
- infer system-level properties from heterogeneous analyses of these heterogeneous models?

Architectures
- Annotated graphs of components and connectors to represent system structure
- Standardized notations (architectural styles) provide a vocabulary of components and connectors as well as certain classes of properties

Proposed CPS Architectural Style [1]
- Cyber elements based on principal computational elements and pathways in the system, e.g., controller, estimator components, point-to-point, publish-subscribe connectors
- Physical elements based on effort-flow modeling, e.g., source and storage components, equal effort or power-flow connectors
- Interface elements, e.g., C2P and P2C transducers

Multiple modeling formalisms for CPS

How do we
• guarantee that the models represent the actual system?
• guarantee that the models are consistent with each other?
• infer system-level properties from heterogeneous analyses of these heterogeneous models?

Architectures
• Annotated graphs of components and connectors to represent system structure
• Standardized notations (architectural styles) provide a vocabulary of components and connectors as well as certain classes of properties

Proposed CPS Architectural Style [1]
• Cyber elements based on principal computational elements and pathways in the system, e.g., controller, estimator components, point-to-point, publish-subscribe connectors
• Physical elements based on effort-flow modeling, e.g., source and storage components, equal effort or power-flow connectors
• Interface elements, e.g., C2P and P2C transducers

Multiple modeling formalisms for CPS

Q. Heterogeneous models with their own parameters and specifications are verified independently. Can we guarantee that the underlying system satisfies its specification (without building a universal model)?

A. For safety specifications, if
1. each model abstracts the underlying system
2. each verification task succeeds, i.e., \(C_i(P_i), M_i \models S_i\)
3. model-level specifications \(S_i\) cover the system-level specification \(S_0\), i.e., \(S_0 \models S_i\)
4. models are external-constraint consistent, i.e.,
   \(C_i^{ext} := (C_i \wedge C_{max}) \downarrow_{P_i} \models C_i\)
then \(C_0(P_0), M_0 \models S_0\). [4]

Tool Support in AcmeStudio

- AcmeStudio is a semantically extensible framework for architectural design and analysis with built-in support for styles, system structure and constraints
- View consistency plugin under development. Uses maximum common sub-graph matching algorithm at back end
- Support for parameter constraints planned

Base Architecture (left) and view (center) modeling in AcmeStudio. View consistency plug-in under development (right).

References