Position Paper for NSF CPS Workshop:

Component-Based Cyber-Physical Systems

Fei Xie
Department of Computer Science
Portland State University
Portland, OR 97207
Email: xie@cs.pdx.edu
Tel: (503) 725-2403

1 Vision

Cyber-physical systems are globally virtual and locally physical in that manipulation of the physical world occurs locally but control and observability are enabled safely and securely across a virtual network. The architectures for cyber-physical systems must enable realization of this key characteristic. In addition, cyber-physical systems are inherently domain-specific due to their close ties to the physical world. Therefore, their architectures must also be flexible and support domain-specific adaptation. We envision a component-based paradigm for cyber-physical systems, under which physical entities being manipulated, hardware, and software are all represented as components of these systems. Composite components are constructed hierarchically from primitive components. Higher-level composite components encapsulate lower-level hardware and physical components. The level of abstraction rises as moving up in the component hierarchies. Locally, primitive physical, software, and hardware components exhibit different execution and interface semantics while globally, higher-level composite components exhibit uniform software semantics.

2 Roadmap for Component-Based Cyber-Physical Systems

The component-based paradigm has the promise of addressing fundamental limitations of today’s cyber-physical systems and providing new abstractions and terminology for specifying, developing, analyzing, and validating future cyber-physical systems.

2.1 Fundamental Limitations of Today’s Cyber-Physical Systems

Today’s cyber-physical system development is lack of a unified view of physical, hardware, and software entities involved in these systems. These entities are often developed in isolation and the systems are often put together in an ad-hoc fashion. As a result, properties of these systems such as correctness, real-time, and performance properties are difficult to establish.
2.2 Important Research Challenges

Important research challenges in realizing component-based cyber-physical systems include:

- **A unified component model is needed.** Cyber-physical systems are globally virtual and local physical. The component concept must reflect this characteristic and provides a unified view from local components: physical, hardware, or software, to global systems.

- **Significant semantic gaps need to be bridged.** The execution and interface semantics of physical, hardware, and software components are significantly different. To consider all these components in a system setting, their semantic gaps must be effectively bridged.

- **Scalable design, analysis and validation capabilities are needed.** To ensure correctness, real-time, and performance of component-based cyber-physical systems, design, analysis, and validation capabilities must scale from local components to global systems.

2.3 Promising Innovations and Abstractions

Key abstractions for component-based cyber-physical systems include components, properties, and platforms. The component concept and the composition mechanism provide the capability for scaling from local to global. By allowing components to have different execution and interface semantics and providing bridges between components of different semantics, an unified view of physical, hardware, and software components can be achieved. Correctness, real-time, and performance properties are treated as first-order elements of a component as its design and interface. Established properties of components are reused with the components. Platforms enable adaptation of the component-based paradigm to different application domains and implementation technologies. A platform dictates the semantics of physical, hardware, and software components for cyber-physical systems in a given domain.

2.4 Milestones for Next 5 to 10 Years

In the next five years, the research should be focused on development a robust component concept that is flexible and a composition mechanism that is scalable. At the same time, the design, analysis, and validation capabilities should be developed simultaneously. In the second five years, the research should be focused on development of the concept of platforms for building highly safe, secure, and reliable cyber-physical systems and on conducting comprehensive evaluations of the above concepts through tool development and case studies.

3 Unified Component Model for Embedded Systems

To facilitate a better understanding of the above concepts, we introduce a unified component model for hardware/software co-design and co-verification of embedded systems, which we have developed in our previous work. In defining this unified component model, we target embedded systems following an abstract but representative architecture as shown in Figure 1. Under this architecture, the software components of an embedded system execute on generic processors while the hardware components are implemented as Application Specific
Integrated Circuits (ASICs). The software components and hardware components interact through an embedded OS that also schedules the execution of software components.

From this abstract architecture, we can derive the unified component model as shown in Figure 2. Under this model, an embedded system is composed of a set of components. There are three types of primitive components: software components, hardware components, and bridge components. Bridge components interact with hardware (or software, respectively) components following hardware (or software) semantics and bridge the semantic gap between hardware and software components by propagating events across the hardware/software semantic boundary. The semantics of bridge components together with the hardware and software semantics abstract the processors, bus models, and embedded OS. Three types of composite components may also be defined: software components, hardware components, and hybrid components. Sub-components of a composite software (or hardware, respectively) components are all software (or hardware) components. A hybrid component contains both hardware and software sub-components, therefore, also bridge components.

**Composition.** A composite component, \( C = (E, I, P) \), is composed from a set of simpler components, \( C_0 = (E_0, I_0, P_0), \ldots, C_{n-1} = (E_{n-1}, I_{n-1}, P_{n-1}) \), as follows. \( E \) is constructed from \( E_0, \ldots, E_{n-1} \) by connecting \( E_0, \ldots, E_{n-1} \) through \( I_0, \ldots, I_{n-1} \). \( I \) may be a hardware interface, a software interface, or a hybrid hardware/software interface depending what types of components \( C_0, \ldots, C_{n-1} \) are. Essentially, \( I \) includes the semantic entities from \( I_0, \ldots, I_{n-1} \) that are needed for \( C \) to interact with its environment and/or for specification of properties of \( C \). Properties of primitive components are directly model-checked. Properties of a composite component, instead of being directly model-checked on the component, are verified on its abstractions constructed from verified properties of its sub-components.

**Platforms.** An embedded system platform for an application domain includes the following elements: (1) software specification language; (2) hardware specification language; (3) bridge specification language; (4) architectural patterns; (5) reusable hardware and software components. A platform essentially provides the semantic information needed for instantiating the unified component model for an application domain. The software, hardware, and bridge specification languages determine the interface semantics of software, hardware, and bridge components and are used to specify the designs or implementations of these components.

**Biography.** Fei Xie is an assistant professor in the Department of Computer Science at Portland State University. He received his Ph.D. in computer science from the University of Texas at Austin in 2004 under the supervision of Prof. James C. Browne. His research interests are primarily in the areas of software engineering, embedded systems, and formal methods. He is particularly interested in development of formal method based techniques and tools for building safe, secure, and reliable software and embedded systems. His research is currently funded by National Science Foundation and Semiconductor Research Corporation.