Preface

The term cyber-physical system (CPS) was introduced by Helen Gill at the NSF referring to the integration of computation and physical processes. In CPS, embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa. The principal challenges in system design lie in this perpetual interaction of software, hardware, and physics.

CPS safety is often critical for society in many applications such as transportations, whether automotive, trains or airplanes, power distribution, medical equipment, or tele-medicine. Whether or not life is threatened, failures may have huge economic impact. Developing reliable CPS has become a critical issue for the industry and society. Safety and security requirements must be ensured by means of strong validation tools. Satisfying such requirements including quality of service implies to have the required properties of the system proved formally before it is deployed.

In the past 15 years, technologies have moved towards Model Driven Engineering (MDE). With the MDE methodology, requirements are gathered with use cases, and then a model of the system is built that satisfies the requirements. Among several modeling formalisms appeared in these years, the most successful are *executable* models, models that can be exercised, tested, and validated. This approach can be used for both software and hardware.

A common feature of CPS is the predominance of concurrency and parallelism in models. Research on concurrent and parallel systems has been split into two different families. The first is based on synchronous models, primarily targeting design of hardware circuits and/or embedded and reactive systems. Esterel, Lustre, Signal, and SCADE are examples of existing technologies of this nature. Additionally, in many places, these have been connected with models of environments that are required for CPS modeling. The second family addresses loosely coupled systems, where communication between distributed entities is asynchronous by nature. Large systems are, indeed, mixing both families of concurrency. They are structured hierarchically and they comprise multiple synchronous devices connected by networks that communicate asynchronously.

In an architectural model, a CPS is represented by a distributed system. The model consists of components with well-defined interfaces and connections between ports of the component interfaces. Furthermore, it specifies component properties that can be used in analytical reasoning about the system. Models are hierarchically organized: each component can contain another sub-system with its own set of components and connections between them. An architecture description language for embedded systems, for which timing and resource availability are primary characteristics of requirements, must additionally describe resources of the system platform, such as processors, memories, and communication links. Several architectural modeling languages for embedded systems have emerged in recent years, including the SAE AADL, SysML, and UML MARTE.

An architectural specification serves several important purposes. First, it breaks down a system model into components of manageable size to establish clear interfaces between these components. In this way, complexity becomes manageable by hiding details that are not relevant at a given level of abstraction. Clear, formally defined, component interfaces allow us to avoid integration problems at the implementation phase. Connections between components, which specify how components affect each other, help propagate the effects of a change in one component to the linked components.

More importantly, an architectural model serves as a repository to share knowledge about the system being designed. This knowledge can be represented as requirements, design artifacts, and component implementations, all of which are held together by a structural backbone. Such a repository enables automatic generation of analytical models for different aspects of the system, such as timing, reliability, security, performance, or energy consumption.

In most cases, however, quantitative reasoning in architecture modeling and CPS is predominantly parameterized by the dimension of time. An architecture or CPS model refers to software, hardware, and physics. In each of these viewpoints, time takes a different form: continuous or discrete, event-based or time-triggered. It is, therefore, of prime importance to mitigate heterogeneous notions of time to support quantitative reasoning in system design, either using a tentatively unified model for it, or by formalizing abstraction/refinement relations from one to another in order to mitigate heterogeneity.

Despite recent research activities in the aim of formally defined or semantically interpreted architectural models, we observe a significant gap between the state of the art and practical needs to handle evolving complex models. In practice, most approaches cover a limited subset of the language and target a small number of modeling patterns. A more general approach would most likely require a semantic interpretation, an abstraction and a refinement, of the source architecture model by the target analytic tool, instead of hard-coding semantics and patterns into the model generator.

Duing March 21–24, 2016, we organized an NII Shonan Meeting on *Architecture-Centric Modeling, Analysis, and Verification of Cyber-Physical Systems*. The meeting invited 22 world-leading researchers from Asia, Europe, and North America and provided a unique opportunity for the participants to exchange ideas and develop a common understanding of the subject matter. Further information about this meeting can be found at the Web site (http://shonan.nii.ac.jp/seminar/073/) and in the meeting report, No. 2016-5 (http://shonan.nii.ac.jp/shonan/report/).

The meeting brought together contributions by research groups interested in defining precise semantics for architecture description languages, and of using this mathematical foundation to leverage tooled methodologies. Such new methodologies generate analytical models, for the purpose of simulation and formal verification, components integration, or requirements validation. Furthermore, they generate code, for the purpose of interfacing components properly, for the purpose of orchestrating execution of components with heterogeneous policies, and for the purpose of real-time scheduling execution of application thread components.

The book consists of chapters addressing these issues, written by the participants of the meeting, which offers snapshots on the state of the art in each of every viewpoint of the problem at hand, and means to put them through.

Tokyo, Japan — Shin Nakajima
Rennes, France — Jean-Pierre Talpin
Kariya, Japan — Masumi Toyoshima
Huntsville, USA — Huafeng Yu
January 2017

Springer

http://www.springer.com/978-981-10-4435-9

Cyber-Physical System Design from an Architecture
Analysis Viewpoint
Communications of NII Shonan Meetings
Nakajima, S.; Talpin, J.-P.; Toyoshima, M.; Yu, H. (Eds.)
2017, XIV, 159 p. 52 illus., 32 illus. in color., Hardcover
ISBN: 978-981-10-4435-9