INVITED PAPER

Cyber–Physical Systems: A Perspective at the Centennial

This paper surveys cyber–physical systems and the potential benefits of the convergence of computing, communications, and control technologies for developing next-generation engineered systems.

By Kyoung-Dae Kim and P. R. Kumar, *Fellow IEEE*

ABSTRACT | Cyber–physical systems (CPSs) are the next generation of engineered systems in which computing, communication, and control technologies are tightly integrated. Research on CPSs is fundamentally important for engineered systems in many important application domains such as transportation, energy, and medical systems. We overview CPS research from both a historical point of view in terms of technologies developed for early generations of control systems, as well as recent results on CPSs in many relevant research domains such as networked control, hybrid systems, real-time computing, real-time networking, wireless sensor networks, security, and model-driven development. We outline the potential for CPSs in many societally important application domains.

KEYWORDS | Cyber–physical systems (CPSs); hybrid systems; model-driven development; networked control systems; real-time systems; security; verification and validation; wireless sensor networks

I. INTRODUCTION

The engineering research field of cyber–physical systems (CPSs) has drawn a great deal of attention from academia, industry, and the government due to its potential benefits to society, economy, and the environment [1]. As a whole, CPSs refer to the next generation of engineered systems that require tight integration of computing, communication, and control technologies to achieve stability, performance, reliability, robustness, and efficiency in dealing with physical systems of many application domains [2].

Even though the specific context of problems and challenges of today's CPSs is different from those in the past, the basic goal of developing control systems through integration of technologies from computing and communication has roots that go back nearly a century. For example, at the time of World War II, the development of automatic antiaircraft guns was one of the most important and challenging problems that required tight integration of technologies from the mechanical, electrical, electronics, and communication fields [3], [4]. In a much broader sense, we may also interpret CPSs as physical systems controlled or manipulated in a principled manner through engineering technologies. With such an interpretation, the history of CPSs can easily be traced back to the Industrial Revolution sparked by the development of the steam engine governor in the 18th century. Hence, we can view and understand the emergence of today's CPSs as a continuation of technological evolution that started from the early uses of feedback control technologies.

Over the last several decades, the advancements in computing and communication technologies have been so significant that we now refer to them as having collectively given rise to an information technology (IT) revolution. In fact, every aspect of today's individual, social, industrial, and economic activities are highly dependent on such cyber–system technologies. In particular, the Internet has changed the way we interact and communicate with each other and also how we create, distribute, and consume information. Continuing this trend, the advent of ubiquitous embedded computing, sensing, and wireless networking technologies are becoming the key enabling technologies for how we interact, control, and build physical engineered systems such as automobiles, aircrafts, power grids,

Manuscript received January 14, 2012; revised February 11, 2012; accepted February 11, 2012. Date of publication April 3, 2012; date of current version May 10, 2012. This work was supported in part by the National Science Foundation (NSF) under Contracts CNS-1035378, CNS-0905397, CNS-1035340, and CCF-0939370, by the United States Army Research Office (USARO) under Contracts W911NF-08-1-0238 and W-911-NF-0710287, and by the U.S. Air Force Office of Scientific Research (AFOSR) under Contract FA9550-09-0121.

The authors are with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843-3128 USA (e-mail: kdkim@tamu.edu; prk@tamu.edu).

Digital Object Identifier: 10.1109/JPROC.2012.2189792

manufacturing plants, medical systems, and building systems, on which our modern society and economy are becoming highly dependent.

The potential benefits of the convergence of computing, communication, and control technologies for developing next-generation engineered systems that can be called CPSs are transformative and wide ranging. Through real-time embedded systems for distributed sensing, computation, and control over wired or wireless communication networks, multiobjective optimization, high-level decision-making algorithms, and formal verification technologies, engineered systems in many societally critical application domains such as transportation, energy, and medical systems can be designed and developed to be much more smart, reliable, secure, efficient, and robust. Needless to say, there are many challenges ahead that need to be addressed in the future. These efforts will have to span all the constituent fields.

The spectrum of research fields relevant to CPSs is very broad. This overview paper is not an exhaustive survey that covers every aspect of CPS research, and is necessarily limited by the knowledge of the authors. In Section II, we review the history of control, communication, and computing technologies leading to CPSs. Then, we review recent achievements in many research disciplines. In Section III, we review research advances in selected areas, networked control systems and hybrid systems, which constitute some of the theoretical foundations for design and analysis of the dynamical behavior of CPSs. In Section IV, we discuss theories and technologies vis-a-vis real-time computing and networking. In Section V, we review fundamental theoretical results and implementation platforms for wireless sensor networks. In Section VII, we discuss the design and development of CPSs from the software engineering point of view. In Section VIII, we conclude by envisioning opportunities and challenges in some domains.

II. THE COMING OF AGE OF CYBER–PHYSICAL SYSTEMS

Computers were originally invented to perform computation. The first computer ENIAC [5] was constructed in 1946 to perform ballistic calculations. However, computers subsequently began to be used to close control loops around physical systems. This motivated the development in 1973 of real-time computation [6], [7], which involved the problem of how to schedule computational tasks so that every job in every task was completed before its deadline. This constituted a significant shift in the usage of computers. If performing calculations correctly was the only purpose, then all one needed was to ensure the *order* of computations. There is no need to deal with physical time. However, if one is interfacing a computer with a physical plant, then the *time* by which computations are performed is important. So, already by this time, there was interest in CPSs, though the name itself was to be invented much later.

In the 1990s, there began to appear much greater interest in the interaction between computational and physical systems [8]. Specifically, with the physical plant modeled by differential equations, and the computational systems modeled by finite state machines or other discrete models of computation, the interest centered on how the interaction of the two evolved. This field was called hybrid systems, reflecting the composite nature of the overall system.

Around 2006, researchers, predominantly in real-time systems, hybrid systems, and control systems, coined the name "cyber–physical systems" to describe this increasingly important area at the interface of the cyber and physical worlds.

There are several other paths also leading to this area of interest. From its origins as ARPANET [9] in 1969, the Internet developed into a worldwide network connecting computers. Around 1973 was the beginning of the cellular telephony revolution. Also around 1971 the ALOHA network was developed to interconnect users across the Hawaiian islands with a mainframe computer in Oahu [10]. Its pioneering ideas, concerning how to resolve contention of the shared wireless medium, were used in Ethernet as well as packet radio networks. In 1977, the U.S. Defense Advanced Research Projects Agency (DARPA) tested the PRNET packet radio network [11]. In 1978, the U.S. Army deployed the Single Channel Ground and Airborne Radio System (SINCGARS) packet radio system [12]. Subsequently, in 1997, the IEEE 802.11 WiFi standard was developed and proliferated across offices and homes after the introduction of IEEE 802.11b [13]. All this, including the landline telephone network, have led to a communication revolution. The goal of interconnecting computers to form a communication network has played a central role in ALOHA, the Internet, and WiFi. Thus, we see here the convergence of communication and computation.

Around 1998, a new element was added to the mix—sensing, with the development by the Smart Dust project [14] of a mote, a tiny device capable of sensing, communication, and computation. These motes allowed the attachment of sensors to the nodes, bringing information about the physical environment into the interconnected wireless communication network of computational nodes.

When nodes in a communication network are connected to both sensors and actuators, one obtains a networked control system. Thus, again, we see an evolutionary path from communication and computing to, in this case, *networked* CPSs.

There is yet another path that one can trace to the present interest in CPSs. In the modern electronic era, the first generation of control systems, analog control, was based on the operational amplifier [15]. To use this technology, a theoretical framework was needed. The

appropriate framework was the frequency domain approach, developed by Nyquist [16], Bode [17], Evans [18], and others. This also led to CPSs—though based on analog computation. One can regard Ziegler–Nichols tuning rules [19], for example, as methods to adjust the overall CPS to achieve desired behavior. Already, by 1954, there was beginning to emerge the second generation of control—digital control [20]. This was spawned by the development of the digital computer. Now simple calculations on algorithms could be performed on the measured signals before closing the loops. This too required a theoretical framework; the appropriate one in this case was the state–space approach. This was developed by Bellman [21], Pontryagin [22], Kalman [23]–[27], and others under the leadership of Solomon Lefschetz at the Martin Company's Research Institute for Advanced Study in Baltimore which was founded in 1955. This led to a very strong foundation of systems theory, with a thorough investigation of optimal control [28], stability [29], linear systems [30], nonlinear control [31], stochastic systems [32], adaptive control [33], robust control [34], infinite-dimensional systems [35], decentralized control of complex systems [36], discrete event systems [37], and even attempts at integrating automata theory and control [38].

Digital control is more than 50 years old, and in the intervening years there have been dramatic advancements in the power of computers as well as the proliferation of embedded computers. There has also been enormous growth in the complexity of software and in the programming abstractions that have been developed for building them. Finally, wireline and wireless data networks were nonexistent 50 years ago. Thus, the emergence of networked CPSs is leading to a third generation of control systems. There has been evolution in the technology of control system implementations on distributed systems. In process control, the controller area network (CANBus) [39] has been used to provide the underlying communication network for distributed control systems. There has also been developed the Field Bus system [40] for interconnection. There is also interest in the "Internet of Things," where physical objects are assigned addresses and interconnected with each other, with interest therefore focused on the communication–physical system interface. All this, together, constitutes yet another platform revolution. At such a time of platform revolution, it is necessary to examine both mechanisms as well as policies. By "mechanisms," we mean how to implement a system, while by "policies," we mean what to implement, for example, which control law.

There is also a great impetus from the viewpoint of applications of societal interest to develop more complex control systems featuring sensing, actuation, and computation capabilities connected by a communication network. There is an increasing demand for more and better transportation systems, energy systems, healthcare systems, and water systems, across all segments of the planet. Due to these demands, as well as the increasing awareness of the resource limitations of the planet, the 21st century could well be the age of building large systems. Many if not most or all of these systems will be composed of complex CPSs.

All these trends—the convergence of several disciplines, the evolution of technology in various fields, and the increasing need to build large scale systems to meet the burgeoning societal needs in an environment of resource frugality—have led to great research interest in the issues sought to be captured by the phrase of CPSs [1].

III. MODELING AND ANALYSIS OF CPS DYNAMICS

The dynamics of CPSs is complex, involving the stochastic nature of communication systems, discrete dynamics of computing systems, and continuous dynamics of control systems. In this section, we review recent theoretical results on modeling and analysis of dynamical behavior of CPSs from different points of view.

A. Networked Control Systems (NCS)

One of the fundamental characteristics of today's CPSs is the existence of a communication network mediating between and among computing and physical entities as shown in Fig. 1. The interactions between controller and the physical system can therefore experience network-induced delay. Packets can even occasionally be lost. The network's links can be regarded as communication channels that are subject to data rate constraints. Hence, some of the fundamental questions that are of importance for networked control systems are as follows. 1) How do the network-induced delay, packet loss, and communication channel affect the stability of the system? 2) Under what conditions is an NCS stabilizable, and how does one stabilize it?

The first issue is when to sample a physical system. The traditional approach is to sample it periodically or at predetermined instants. An alternative is to sample it when specific events occur, e.g., when a signal crosses a level. These have been called Riemann and Lebesgue sampling [41]. The latter approach requires continuous monitoring of the system to detect when to sample it. An alternative is to decide a safe interval for which the system can be left unsampled and an appropriate time to sample it

Fig. 1. *Structure of networked control systems.*

next. This is called self-triggering and can lead to more efficient monitoring as well as usage of resources, and even be used to guarantee stability based on some knowledge of the plant [42]–[44].

To study the effect of network-induced delay, consider an NCS modeled as consisting of a linear continuous time plant and a controller exchanging data packets over a lossless communication network that is shared with other unrelated nodes [45], [46]. Define the network-induced error $e(t) := [\hat{y}(t)\,\hat{u}(t)]^T - [y(t)\,u(t)]^T$ where $y(t)$ is the output of a plant, $u(t)$ is the output of a controller, and $\hat{y}(t)$ and $\hat{u}(t)$ are the most recently received versions of $y(t)$ and $u(t)$, respectively. If there is no network-induced delay between plant and controller, then $\hat{y}(t) = y(t)$ and $\hat{u}(t) = u(t)$ and so $e(t) = 0$ for all t. A network scheduling strategy, called maximum-error-first with try-once-discard (MEF-TOD), which dynamically assigns the packet transmission order among nodes to share the network is proposed in [45] and [46]. The notion of maximum allowable transfer interval (MATI) is introduced to bound the amount of time between transmission events and derive a sufficient condition in terms of MATI for stability of the NCS.

Another approach to the stability analysis of an NCS [47] is by using hybrid systems analysis techniques [48]. As a model of an NCS, consider a plant $\dot{x}(t) = Ax(t) + Bu(t)$ for $t \in [kh + \tau, (k+1)h + \tau)$, and a state feedback controller

$$u(t^+) = -Kx(t - \tau), t \in \{kh + \tau : k = 0, 1, \ldots\} \quad (1)$$

where h is the sampling period, τ is the fixed network-induced delay that is the sum of the delays from sensor to controller and controller to actuator, and $u(t^+)$ is piecewise continuous changing values only at $kh + \tau$. Stability is guaranteed if the following matrix has all its eigenvalues inside the unit disk:

$$H = \begin{bmatrix} e^{Ah} & -E(h)BK \\ e^{A(h-\tau)} & -e^{A\tau}(E(h) - E(\tau)BK) \end{bmatrix} \quad (2)$$

where $E(a) := \int_0^a e^{A(a-s)}ds$. Instead of (1), one can consider a state feedback controller that uses an estimated plant state $\hat{x}(kh + \tau)$ to compute a control input at $kh + \tau$ [49].

A more general framework for stability analysis of the NCS is to consider a nonlinear NCS with disturbance and also a general class of network scheduling protocols, called Lyapunov uniformly globally exponentially stable (UGES) protocols [50]. Both the round-robin (RR) static scheduling protocol and the MEF-TOD dynamic scheduling protocol considered in [45] turn out to be Lyapunov UGES protocols. Moreover, the input–output $\mathcal{L}_p$ stability of the NCS for Lyapunov UGES protocols is shown in [50] based on the small gain theorem.

A data packet that is transmitted, especially over wireless, can be dropped. One way to model packet loss [51] is as an asynchronous sample and hold switch which closes with a certain rate r. The NCS with packet loss can then be modeled as an asynchronous dynamical system (ADS) incorporating both discrete and continuous dynamics, and its stability analyzed through Lyapunov-based analysis. Lower bounds on the transmission rate r needed for stability can be obtained [52].

The stabilization of an NCS over a channel that is prone to packet drops can be addressed through robust control analysis and synthesis techniques [53]. An NCS can be viewed as a feedback interconnection of a deterministic nominal system, denoted as G, and a zero-mean stochastic structured model uncertainty, denoted as Δ. The stability problem can then be formulated as a linear matrix inequality (LMI) feasibility problem. Using the notion of mean square structured norm of G, denoted by $\mu_{\mathrm{MS}}(G, \Delta)$, the controller design problem for stabilizing an NCS can be posed as an optimization problem

$$\begin{aligned} \mu^*_{\mathrm{MS}}(G, \Delta) &= \inf_{K-\mathrm{stab,LTI}} \mu_{\mathrm{MS}}(G, \Delta) \\ &= \inf_{\theta > 0, \mathrm{Diag.}} \inf_{K-\mathrm{stab,LTI}} \|\theta^{-1} G \theta\|^2_{\mathrm{MS}} \end{aligned} \quad (3)$$

where the infimum is taken over all stabilizing LTI controller K for the given feedback interconnection of G and Δ. However, it turns out that the search for the controller K^* with the largest stability margin $\mu^*_{\mathrm{MS}}(G, \Delta)$ is nonconvex with respect to the parameter θ. Hence, the optimization problem (3) is intractable in general. However, it is shown in [53] that, for any fixed $\theta > 0$, the optimization problem (3) can be converted into an equivalent LMI optimization problem and the optimal controller K^* can be determined through it.

The problem of state estimation over a lossy communication link corresponds to a filtering problem with intermittent observations [54]. More explicitly, the plant can be modeled by a discrete time linear Gaussian system $x_{t+1} = Ax_t + w_t$, where packets $y_t = Cx_t + v_t$ arrive with probability $(1 - \alpha)$ as a Bernoulli process, and w_t and v_t are independent identically distributed (i.i.d.) Gaussian random vectors. If the matrix C is invertible, then for a stable Kalman filter, it is necessary that the packet drop probability $\alpha < (1/(\max_i |\lambda_i(A)|)^2)$, where $\lambda_i(A)$ are the eigenvalues of A.

One can formulate the control of NCSs as an optimal control problem for LTI systems over a lossy communication link, with an uplink from sensor to controller, and a downlink from controller to an actuator that is collocated with the plant [55], [56]. A fundamental problem that arises when there are packet drops between the controller

that computes a potential value of control to be applied, and an actuator that actually applies control inputs, is the resulting nonclassical information pattern which renders very difficult the computation of the optimal control law under a linear quadratic control framework [57]. This difficulty disappears when the network protocol is a TCP-like protocol, i.e., a notification of successful reception is available. Then, there is indeed separation of estimation and control [58]. A sufficient condition for the stabilizability of an NCS is $\max\{\alpha, \beta\} < (1/(\max_i |\lambda_i(A)|)^2)$, where α and β are critical values of drop probabilities for uplink and down link.

In the NCS, it is also important to determine where in the network to perform calculations required by the control law. Under some conditions, the optimal placement of a controller in the NCS is to collocate it with the actuator [59]. Also, the above condition is then a necessary and sufficient condition for the existence of a stabilizing controller in the presence of packet drops, even when the matrix C is not invertible.

Another important issue is how the presence of the data-rate limited communication channels affects the stabilizability of the system. An early precursor [60] considers the problem of optimal control with respect to a long-term average quadratic cost criterion of a linear Gaussian system. The channel is modeled as one that appropriately delays finite length codewords. It is shown that when the encoder codes the innovations process of the state estimate rather than the state itself, then there is a separation theorem and the optimal control is linear in the state estimate. More recently, there has been increasing attention paid to the problem of stabilizing a linear system when some of the feedback loop has to be closed over a communication channel of limited data rate. In an early work [61], the plant is modeled as a linear deterministic continuous system. The communication channel's limited data rate is modeled by assigning a long time delay, proportional to the number of bits that are sought to be communicated. An instantaneous output measurement taken at a certain time is simply quantized by a symbol from a finite alphabet. The decoder can however choose an appropriate control, from a finite set, based on the past history of all encoded measurements received. An unstable system is not asymptotically stabilizable, and an appropriate notion of containability related to the ability to keep the system in an open sphere around the origin when it is started close enough to the origin is introduced. It is shown that an inequality relating the rate of change of the system and the data rate is sufficient for containability. Another early paper [62] considers a scalar plant with a channel capable of transmitting R bits per second without noise, and shows that in order to keep the trajectory bounded when sampling uniformly it is necessary for the rate R to exceed a multiple of the logarithm of the absolute value of the unstable eigenvalue. In [63], the problem of quantization is studied where the sensitivity (i.e., fineness) of the quantizer is varied within a bounded neighborhood of the origin. In [64], it is shown that for quadratic stabilizability, the optimal sampling time depends on the sum of the logarithms of the magnitude of the unstable eigenvalues, and that the optimal quantization levels are logarithmic. In [65], a discrete linear system is considered, and the channel is modeled as being able to transmit R bits perfectly in each second, i.e., as a bit pipe. It is shown that for asymptotic stabilizability it is necessary that the data rate exceeds the sum of the logarithms of the magnitudes of the unstable eigenvalues of the system matrix. When the encoder has access to past control inputs that were applied, some of the complications caused by information patterns, as in [57], do not arise, and it is shown that such a rate is also sufficient for asymptotic stabilizability. A companion paper [66] considers the case of noisy channels, and a similar necessary condition is shown on the rate, defined in a Shannon-theoretic sense, for almost sure asymptotic stabilizability. For certain channels with erasures, and when past control inputs are available to the encoder, this rate is also shown to be sufficient. In [67], a deterministic scalar autoregressive-moving-average (ARMA) system with a random initial condition is considered. The channel is modeled as a bit pipe, and a necessary and sufficient condition is obtained on the rate to ensure that the mth moment of the state is driven to zero. The minimum data rate needed for mean-square stabilizability of a system with both state and observation noises in treated in [68]. A dynamic quantizer is used to account for possible unbounded values of the state. There has also been attention to the case of bit pipes where the data rate varies randomly. In fact, the case of dropped packets can be regarded as a special case where the rate can be zero. The case of i.i.d. rate variation is considered in [69] and [70]. The case of a channel which changes between 0 and a certain rate as a two-state Markov chain is considered in [71]. The case of a more general Markovian channel rate evolution is examined in [72]. The concept of anytime capacity is introduced in [73] to capture a noisy communication channel when it is used as part of a feedback loop to stabilize an unstable linear system. Again, for scalar systems, the required rate is larger than the logarithm of the unstable systems' gain. The issue of coding for noisy communication channels when they are used to close control loops is examined in [74]–[77]. The case when the channel noise is Gaussian is simpler because uncoded transmission can be used [78]; this problem is connected to the problem of communication in the presence of feedback.

Further results on the NCS can be found in [79]–[81] on optimal control over a communication channel, in [82] and [83] on NCS with sampling and delay, in [84] and [85] on stability and control analysis of the NCS through delayed differential equation framework, in [86] on wireless control network where the entire network itself acts as the controller, in [87] and [88] on decentralized control problems, and the references in [72] and in [89] and [90] for a survey of this field.

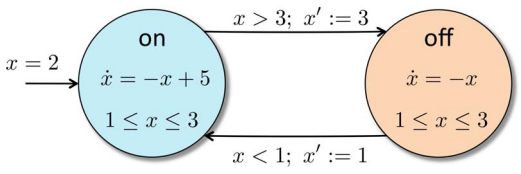

Fig. 2. *An example of a hybrid automaton [99].*

B. Hybrid Systems

CPSs are typically required to adapt to various changes in internal and external factors. One way of adaptation is through "switching" between different operation "modes" which results in a switched system. The class of systems with switching can be described by $\dot{x} = f_\sigma(x)$, where $\sigma : [0, \infty) \to \mathcal{P}$ is a piecewise constant function of time, called a switching signal, and $\mathcal{P}$ is some index set [91]. The stability of such systems has been studied, and recent results can be found in [91] and [92] and the references therein.

A more general modeling framework for CPSs is hybrid automaton (HA), which can be used to model complex dynamics of CPSs through various mathematical formalisms [93]–[98] that can capture both the transition between discrete states and also the evolution of continuous states over time. One useful HA model developed for algorithmic verification of CPS has the following components [94].

- A finite directed graph $\langle V, E\rangle$ where each $v \in V$ is called a control mode or a location, and each edge $e \in E$ is called a control switch.
- A finite set of continuous real-valued variables $X = \{x_1, \ldots, x_n\}$. The first derivative of X is written as $\dot{X} = \{\dot{x}_1, \ldots, \dot{x}_n\}$ and $X' = \{x'_1, \ldots, x'_n\}$ is used for the value of X at the conclusion of a discrete change.
- Two edge labeling functions *guard* and *reset* that assign to each $e \in E$ predicates of variables from $X \cup X'$ to indicate a discrete transition condition and a reinitialization of a continuous variables.
- Three vertex labeling functions *init*, *invariant*, *flow* to indicate an initial, invariant, and flow condition for each $v \in V$. Both *init(v)* and *invariant(v)* are predicates with variables from X, while *flow(v)* is a predicate with variables from $X \cup \dot{X}$ to describe the dynamics of X within a mode.

A simple example of an HA is Fig. 2 in which $V = \{\text{on}, \text{off}\}$ and $X = \{x\}$. For the mode *on*, three vertex labeling functions are $x = 2$ for *init(on)*, $x \in [1, 3]$ for *invariant(on)*, and a differential equation $\dot{x} = -x + 5$ for *flow(on)*. The discrete transition, or control switch, from *on* to *off* occurs based on an edge labeling function *guard* $x > 3$ for the mode *on*, and the variable x is reset to a value by a *reset* map $x' := 3$ during the transition.

Safety verification of a given hybrid automaton $\mathcal{A}$ can be addressed by determining whether the set $\overline{\text{Reach}}(\mathcal{A}, \mathcal{I}) \cap \mathcal{U}$ is nonempty, where $\mathcal{I}$ denotes a given initial set of states, $\mathcal{U}$ denotes a set of unsafe states, $\text{Reach}(\mathcal{A}, \mathcal{I})$ represents the set of states reached by executions of $\mathcal{A}$ starting from $\mathcal{I}$, and $\overline{\text{Reach}}(\mathcal{A}, \mathcal{I})$ is an overapproximation of $\text{Reach}(\mathcal{A}, \mathcal{I})$.

$\text{Reach}(\mathcal{A}, \mathcal{I})$ can be computed through the iteration $\varphi_{k+1} = \text{Post}(\varphi_k)$, where φ_k is the set of reached states at the kth step, and $\text{Post}(\varphi_k)$ is the set of states that is the union of φ_k and the set of states reached from φ_k through a discrete transition and continuous flow. If φ_k and φ_{k+1} coincide for some finite number k, then the algorithm terminates, returning $\text{Reach}(\mathcal{A}, \mathcal{I})$. However, it is well known that the exact computation of $Reach(\mathcal{A}, \mathcal{I})$ is undecidable in general [100], [101]. Hence, in such cases, computing $\overline{\text{Reach}}(\mathcal{A}, \mathcal{I})$ is also an important research issue as we will discuss later.

The first subclass of hybrid automata for which reachability was shown to be decidable is the class of timed automata [102]. Roughly, timed automata are those where 1) the vertex labeling function *flow(v)* is of the form of $\dot{x}_i = 1$; 2) the edge labeling function *reset(e)* either does not change the value of x_i or resets x_i to zero during a discrete transition; and 3) the sets associated with *init*, *inv*, *guard* are all in rectangular form, i.e., a finite boolean combination of the form $x_i \oplus c$ for some $c \in \mathbb{Q}$ and $\oplus \in \{<, \le, =, \ge, >\}$. It is important to note that even though the continuous dynamics of timed automata is very simple from a control perspective, introducing time in a model of computation was a significant conceptual advance in the area of algorithm verification, and a precursor to a lot of work on hybrid systems.

The notions of simulation and bisimulation relations were established in the area of formal methods and used successfully for complexity reduction in discrete systems [103]–[105]. It turns out that they are also very useful for complexity reduction of hybrid systems to address reachability. In [102], the reachability problem for timed automata is shown to be decidable since there exists a finite quotient transition system $\mathcal{R}(\mathcal{A})$ which is bisimilar to the original timed automaton. A quotient transition system is one that is constructed by the partition of the continuous state space. Transition systems $\mathcal{T}_1$ and $\mathcal{T}_2$ are said to be bisimilar if there exists a bisimulation relation $\mathcal{B}$ between $\mathcal{T}_1$ and $\mathcal{T}_2$. Definitions of transition system, simulation, and bisimulation relation are as follows [106].

Definition 1 (Transition Systems): A (labeled) transition system with observations is a tuple $\mathcal{T} = (Q, \Sigma, \to, Q^0, \Pi, \langle\langle\cdot\rangle\rangle)$ where 1) Q is a set of states; 2) $Q^0 \subseteq Q$ is a set of initial states; 3) Σ is a set of labels; 4) Π is a set of observations; 5) $\langle\langle\cdot\rangle\rangle$ is an observation map from Q to Π; and 6) $\to$ is a transition relation such that $\to \subseteq Q \times \Sigma \times Q$ and a transition from q to q' with a label σ is denoted by $q \xrightarrow{\sigma} q'$.

Definition 2 (Simulation): A relation $\mathcal{S} \subseteq Q_1 \times Q_2$ is called a simulation of $\mathcal{T}_1$ by $\mathcal{T}_2$ if for all $(q_1, q_2) \in \mathcal{S}$: 1) $\langle\langle q_1 \rangle\rangle_1 = \langle\langle q_2 \rangle\rangle_2$; and 2) $\forall q_1 \xrightarrow[1]{\sigma} q_1'$, $\exists q_2 \xrightarrow[2]{\sigma} q_2'$ such that $(q_1', q_2') \in \mathcal{S}$.

Definition 3 (Bisimulation): A relation $\mathcal{B} \subseteq Q_1 \times Q_2$ is called a bisimulation between $\mathcal{T}_1$ and $\mathcal{T}_2$ if $\mathcal{B}$ is a simulation relation from $\mathcal{T}_1$ to $\mathcal{T}_2$ and $\mathcal{B}^{-1}$ is a simulation relation from $\mathcal{T}_2$ to $\mathcal{T}_1$.

A result on the decidability of the class of initialized rectangular hybrid automata (IRHA) is shown in [101]. Two important factors for decidability are: 1) rectangularity, that is, if we denote the set of all rectangular regions in $\mathbb{R}^n$ by $\mathcal{R}^n$, then the three vertex labeling functions *init, inv, flow* are all mapping functions from V to $\mathcal{R}^n$, and the two edge labeling functions *guard, reset* are mapping functions from E to $\mathcal{R}^n$; and 2) initialization, that is, a continuous variable has to be reinitialized whenever its flow changes during a discrete transition. In [101], it is shown that slight generalizations from IRHA lead to undecidability.

The o-minimal hybrid systems are defined in [107] as initialized hybrid systems whose relevant sets such as *guard, reset*, etc., and *flow* are definable in an o-minimal (or order-minimal) theory [108], [109]. This class captures hybrid systems with relatively complex continuous dynamics including linear, polynomial, and exponential *flow* dynamics. In [107], it is shown that every o-minimal hybrid system admits a finite bisimulation, and furthermore, the computation of such finite bisimulation terminates. Hence, o-minimal hybrid systems comprise a decidable class of hybrid system.

An interesting class of hybrid systems, called linear hybrid automata (LHA) [99], [110], are those for which, for each $v \in V$ and $e \in E$: 1) the vertex labeling functions *flow(v), inv(v), init(v)*, and edge labeling functions *guard(e), reset(e)* are finite conjunctions of linear inequalities; and 2) more importantly, the flow function *flow(v)* is finite conjunction of linear inequalities over the variables in $\dot{X}$ only. An important result is that if a given HA $\mathcal{A}$ is an LHA, then $Reach(\mathcal{A}, \mathcal{I})$ can be computed exactly [110]. However, it is not guaranteed that the iterative reach set computation terminates.

One of the most common class of hybrid systems of interest has vertex labeling functions *flow(v)* in the form of a differential equation $\dot{x} = f(x)$. An example of an HA with linear differential equations is shown in Fig. 2. For such HAs and other classes of HAs more general than LHA, there is no known algorithm that can compute $\mathrm{Reach}(\mathcal{A}, \mathcal{I})$ exactly even without termination guarantee. Hence, the safety verification problem for this class of HAs can only be addressed through an overapproximation of $\mathrm{Reach}(\mathcal{A}, \mathcal{I})$.

In [99], an approximation technique, called linear phase portrait approximation, is proposed. The basic idea of this technique is to replace the dynamics of $f(x)$ for each $v \in V$ by a corresponding rectangular region that upper and lower bounds the function $f(x)$ over the invariant set for the mode v. As an example, the dynamics $\dot{x} = -x + 5$ for the mode *on* in Fig. 2 can be over-approximated by $\dot{x} \in [2, 4]$ over the range of $x \in [1, 3]$. Then, it is easy to see that the HA in Fig. 2 can be overapproximated by an LHA through this technique.

Another useful technique is to overapproximate the evolution of continuous variables using polyhedral representation [111], [112]. Given a dynamical system $\dot{x} = f(x)$, let $\mathcal{R}_{[t_{k-1}, t_k]}(\mathcal{I})$, called a flow-pipe segment, be the set of states over the time interval $[t_{k-1}, t_k]$ reachable from the initial set $\mathcal{I}$ at time t_0, and let (C, d) be a matrix–vector pair that defines a polyhedron to approximate the flow-pipe segment such that $x \in \{x | Cx \leq d\}$ for any $x \in \mathcal{R}_{[t_{k-1}, t_k]}(\mathcal{I})$. Then, for a given C, the optimal d^* that minimizes the overapproximation error can be determined as the solution to the optimization problem

$$\begin{aligned} \max_{x_0, t} \quad & c_i^T x(t, x_0) \\ \text{s.t.} \quad & x_0 \in \mathcal{I} \quad \text{and} \quad t \in [t_{k-1}, t_k] \end{aligned} \tag{4}$$

where c_i^T is the ith row vector of C that is the unit normal vector to the ith face of the polytope $Cx \leq d$, and $x(t, x_0)$ is the solution of $\dot{x} = f(x)$ at time t from the initial state x_0. Then, from the optimal solution (t^*, x_0^*) of (4), the optimal value d_i^* for the given c_i is determined as $d_i^* = c_i^T x(t^*, x_0^*)$. Now, the question is how to determine C. A heuristic approach is also proposed in [111] based on a convex hull computation from a set of vertices. Assuming that $\mathcal{I}$ is a polyhedron, let $\mathcal{V}(\mathcal{I})$ be the set of vertices of $\mathcal{I}$, and let $\mathcal{V}_t(\mathcal{I}) = \{x(t, v) | v \in \mathcal{V}(\mathcal{I})\}$. Then, the matrix C can be determined by the set of outward pointing normal vectors of the convex hull that is obtained from the set of vertices $\mathcal{V}_{t_{k-1}}(\mathcal{I}) \cup \mathcal{V}_t(\mathcal{I})$.

As noted earlier, the notion of a bisimulation relation between transition systems is crucial for the decidability result of several classes of HAs that have fairly simple continuous dynamics, such as timed automata. This notion can be extended to explicitly include the observation error in its definition so that a larger class of continuous dynamics $\dot{x} = f(x)$ can be abstracted as a finite state transition system that is *approximately* bisimilar to the original continuous dynamics. If we let $\mathcal{T}_M(\Sigma, \Pi)$, called a metric transition system, be the set of transition systems associated with a set of labels Σ and a set of observations Π where (Q, d_Q) and (Π, d_Π) are metric spaces, then, for $\mathcal{T}_1, \mathcal{T}_2 \in \mathcal{T}_M(\Sigma, \Pi)$, an approximate bisimulation relation is defined as follows [106].

Definition 4 (Approximate Bisimulation): A relation $\mathcal{B}_\delta \subseteq Q_1 \times Q_2$ is a δ-approximate bisimulation relation between $\mathcal{T}_1$ and $\mathcal{T}_2$ if for all $(q_1, q_2) \in \mathcal{B}_\delta$: 1) $d_\Pi(\langle\langle q_1 \rangle\rangle_1, \langle\langle q_2 \rangle\rangle_2) \leq \delta$; 2) $\forall q_1 \xrightarrow[1]{\sigma} q_1'$, $\exists q_2 \xrightarrow[2]{\sigma} q_2'$ such that

$(q_1', q_2') \in \mathcal{B}_\delta$; and 3) $\forall q_2 \xrightarrow[2]{\sigma} q_2', \exists q_1 \xrightarrow[1]{\sigma} q_1'$ such that $(q_1', q_2') \in \mathcal{B}_\delta$.

Concerning an approximate bisimulation relation, if a nonlinear control system is incrementally asymptotically stable [113], then it is δ-approximately bisimilar [114] to a symbolic model of the original continuous system that can be constructed by aggregating states and control inputs using several parameters such as $\tau \in \mathbb{R}^+$ for time domain quantization, $\eta \in \mathbb{R}^+$ for state space quantization, and $\mu \in \mathbb{R}^+$ for input space quantization satisfying the following inequality:

$$\beta(\delta, \tau) + \mu + \eta/2 \leq \delta \tag{5}$$

where β is a $\mathcal{KL}$ function [31]. Once we have such a symbolic model of a continuous control system, a controller satisfying a given specification can be synthesized automatically using techniques developed in supervisory control of discrete event systems or algorithmic game theory [114], [115]. Other results relevant to automatic controller synthesis for hybrid systems can be found in [116] on algorithmic controller synthesis through finite bisimulation to satisfy LTL specifications of discrete-time linear systems and in [117] on the synthesis of control laws for piecewise-affine hybrid systems on simplices.

It is important to note that, based on these theoretical results, many software tools have been developed for formal verification and automatic controller synthesis of hybrid systems. Some examples are UPPAAL [118], a verification tool for real-time systems based on timed automata, HyTech [99] and PHAVer [119] for LHA, SpaceEx [120], which is based on the LeGuernic–Girard (LGG) algorithm [121] that can efficiently handle HAs with linear differential equations with a larger number of system states compared to other approximation techniques, and PESSOA [122], which is a tool for controller synthesis based on [114]. More details and results can be found in [123]–[127] for various approaches for reachability, in [128] for other classes of systems for which some verification/synthesis problems are decidable, and in [129]–[132] for abstractions of hybrid systems.

C. Distributed Hybrid Systems

A major goal in the design of CPSs is to have formal proofs of correctness of the overall system design. This overall system can however be quite complex, involving not only differential-equation-based dynamics of the physical system, but also discrete models of the physical system, as well as interaction with real-time computation and communication. The system is the composition of several systems. Thus, proofs of correctness will have to be holistic and transcend domains. An example is a proof of correctness of an automobile traffic control system in [133] and [134]. It involves not only differential equation models of automobiles, but also a balls-and-bins model of the positions of all cars, which is necessary to prove properties such as deadlock avoidance [135]. Also involved is real-time scheduling of the computational tasks. Similarly, the design of CPSs can involve several choices such as the extent of centralization, and the extent of robustness, both of which may have to be made, keeping in mind the provable correctness of the design. An example is a design accompanied with a proof of correctness for an automated traffic intersection [136]. So far, verification for such systems has mostly involved pencil–paper proofs and interactive theorem prover-based verification [137]–[140]. For future systems, it would be valuable to have compositional frameworks, and more systematic or automated methods for proving correctness.

IV. REAL-TIME COMPUTING AND NETWORKING

Computing and networking are key driving forces and key components of new highly connected, distributed, and reliable CPSs. We review classical and recent results in these areas.

A. Real-Time Scheduling Theory

In real-time systems, the correctness of a system depends not only on the logical results of the computation but also on the time at which the results are produced [141]. One of the primary design objectives of a real-time computing system is to support *temporally predictable* execution of a set of computing tasks so that it is guaranteed that there will be timely interaction between computing tasks and the physical environment. More precisely, for a given set of computing tasks $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ with timing constraints, a set of processors $P = \{P_1, P_2, \ldots, P_m\}$, and a set of resources $R = \{R_1, R_2, \ldots, R_r\}$, a real-time computing system has to make an appropriate scheduling decision on Γ to meet all the timing constraints. If some tasks cannot meet their timing constraints, then the system should be able to determine this in advance. However, it is known to be computationally intractable to solve such scheduling problems in general [142].

One of the most influential results in real-time scheduling theory is based on a task set model, which is simple enough to be computationally tractable and also practical enough to be useful in many applications [6]. Consider the scheduling problem for a set of preemptible and periodic tasks under both fixed (or static) and dynamic priority assignment based on the following assumptions: 1) all tasks $\tau_i \in \Gamma$ are independent, i.e., there is no shared resource or precedence relationship between tasks; 2) all instances of a periodic task τ_i have the same relative deadline D_i and it is equal to its period T_i; 3) all instances of a periodic task τ_i have the same worst case execution time C_i; and 4) there is only one processor.

The rate monotonic (RM) policy is a static priority scheduling policy that assigns priorities to tasks based on the rate of arrivals of jobs in the task. The shorter the period of a task, the higher is the priority assigned to the task. It is optimal among all static priority scheduling policies in the sense that if a periodic task set can be scheduled by some static priority policy, then it can be scheduled by RM [6]. Moreover, there is a simple sufficient condition for schedulability: $\sum_{i=1}^{n} U_i \leq n(2^{1/n} - 1)$, where $U_i = C_i/T_i$ is the processor utilization of τ_i. There is also a less conservative schedulability condition for RM, called a hyperbolic bound [143]. For a set of periodic tasks whose relative deadlines are less than their periods, the deadline monotonic (DM) scheduling policy is an extension of RM. For the exact schedulability analysis for a given periodic task set, there is an iterative algorithm, called response-time analysis [144].

The earliest deadline first (EDF) policy assigns priorities to tasks according to the absolute deadline of their instances. Hence, EDF is a dynamic scheduling policy. It is optimal in that if there is any schedule that can meet all deadlines, then EDF will too. For task sets with deadlines less than periods, a necessary and sufficient condition for schedulability under EDF is derived in [145].

Many scheduling algorithms have been proposed to simultaneously handle both *hard* periodic tasks and *soft* aperiodic tasks. The primary objective is to minimize the response time for aperiodic tasks without compromising the schedulability of the periodic tasks. In fixed-priority scheduling, a basic idea is to create a periodic task $\tau_s = (T_s, C_s)$ for serving aperiodic tasks where T_s is the period and C_s is the computation time for τ_s, called server capacity, in addition to the hard periodic task set. Some examples of scheduling algorithms based on this idea are polling server [146], deferrable server [147], and priority exchange server [148].

For dynamic scheduling, especially under EDF, one useful idea, called the total bandwidth server (TBS) [149], to handle aperiodic requests along with a set of periodic tasks, is to assign each aperiodic request a deadline such that the overall aperiodic load never exceeds a specified value of processor utilization U_s that is called the *bandwidth* of an aperiodic server. When the kth aperiodic request which requires C_k amount of execution time arrives at time r_k, then the deadline assigned to this aperiodic task is $d_k = \max\{r_k, d_{k-1}\} + (C_k/U_s)$, where d_{k-1} is the deadline assigned previously for the $(k-1)$th aperiodic request. However, TBS cannot be used when C_k is unknown. For such cases, a bandwidth reservation mechanism, called the constant bandwidth server (CBS), is proposed in [150]. The basic idea of CBS is that when a new aperiodic request arrives, it is assigned a suitable deadline that is determined by the currently available bandwidth resource for the request. If the request cannot be completed before its deadline, then its deadline is postponed. Notice that, under EDF this implies that its priority is decreased and thus the interference to the other tasks is reduced.

For a task set that consists of aperiodic tasks with arbitrary arrival times, execution times, and deadlines, a utilization bound for schedulability is derived in [151]. In particular, the notion of *synthetic utilization*, denoted as $U(t)$, is introduced, which is roughly defined as the sum of utilization values of all arrived aperiodic requests whose deadlines have not yet expired at time t. A set of n aperiodic tasks is schedulable under the deadline monotonic scheduling policy if, for all t, $U(t) \leq UB(n)$ where $UB(1) = 1$, $UB(2) = 0.75$, and $UB(n) = 1/1 + \sqrt{(1/2)(1 - (1/n - 1))}$ for $n \geq 3$. Deadline monotonic scheduling policy is optimal among all time-independent scheduling policies, a generalized notion of fixed-priority scheduling that applies to aperiodic tasks, since no other time-independent scheduling policy can have a higher upper bound for $U(t)$. For the case of dynamic aperiodic task scheduling, EDF is optimal and its utilization bound is 1.

In reality, tasks are not generally independent since they typically share resources such as memory, files, communication network, etc., for their execution in a mutually exclusive manner. In such cases, a higher priority task can be blocked by a lower priority task due to resource sharing. This is called *priority inversion* and its duration can be arbitrarily long. In [152], a simple solution is proposed that is called the priority inheritance protocol (PIP). The basic idea of PIP is to let a lower priority task which currently holds the shared resource to inherit temporarily the highest priority among the blocked tasks, until it releases the resource. After releasing the resource, it recovers its original priority. However, it is known that priority inheritance does not prevent deadlocks. The priority ceiling protocol (PCP) is also proposed in [152] as an extension of the PIP to resolve this issue. Under EDF, the PIP and the PCP are not applicable since they are based on the fixed-priority scheduling system. For such cases, the stack resource policy (SRP) [153], extended from the PCP to support dynamic priority scheduling and to allow the sharing of runtime stack-based resources, is a useful mechanism that is applicable to both fixed-priority and dynamic scheduling.

It is of interest to determine guarantees for jobs that have to be processed by a sequence of processors, i.e., as they move through a network. The work on stability of reentrant lines provides bounds on end-to-end delays that are of potential interest because it establishes a pipeline property where the delay is related to the bottleneck node [154]. Other important results can be found in [155] and [156] on real-time queueing theory for stochastic analysis of soft real-time systems, in [157] and [158] on real-time scheduling analysis in a resource partitioned computing environment, in [159] on resource kernels as an approach for operating system resource management based on resource reservation for real-time applications, in [160]–[162] on a control theoretical approach to performance and throughput management of computing systems, in [163] on real-time scheduling algorithms for power management in

embedded real-time systems, and in [7] for more on real-time system theories.

B. Real-Time Systems for CPSs

There are several important characteristics which make today's CPSs different from earlier generation control systems: 1) the scale of a CPS is much larger; 2) entities comprising a CPS typically run over heterogeneous environments; and 3) entities interact with each other in a very complex manner. It is also expected that CPSs should be highly extensible for new functionalities, and flexible for runtime adaptation. Due to such structural and behavioral complexities, it is more challenging to design and implement a CPS. To overcome such complex issues, it is becoming increasingly important to develop a software platform, called middleware, based on an appropriate abstraction of such complex systems, and a well-designed architecture for rapid implementation of reliable and evolvable CPS applications [134].

The Common Object Request Broker Architecture (CORBA) [164] is a well-known industry standard specification for middleware developed by the Object Management Group (OMG). It is designed primarily for interoperability between software objects running on different machines in a heterogeneous distributed environment. Thus, it is not designed for control system applications in which temporal predictability is essential. Later, Real-Time CORBA has been developed as an extension of CORBA to support temporally predictable end-to-end interactions between client and server objects in a system. It defines a set of mechanisms and interfaces which enable applications to explicitly manage system resources such as synchronization mechanism, thread pool model, scheduling service, and explicit binding.

The ACE ORB (TAO) [165] is an implementation of Real-Time CORBA. It has been used in application areas such as telecommunication, aerospace, medical, and financial services. It is used as a middleware framework for an application development platform, called open control platform (OCP) [166], developed for complex and reconfigurable control system applications under the U.S. DARPA Software Enabled Control research program.

Etherware [133] is a middleware developed for large-scale networked control systems. It is based on the concept of microkernel architecture and supports component-based application development. It also supports runtime reconfiguration, such as component upgrade, and even migration at runtime from one computing node to other node. This is possible through an Etherware component model that is based on several software design patterns [167]. Etherware has been enhanced to support time-critical systems by incorporating quality of service (QoS) in component interaction and a real-time scheduling mechanism for interactions [168]. As an illustrative example of Etherware-based CPS, Fig. 3 shows how a distributed traffic system can be developed over Etherware.

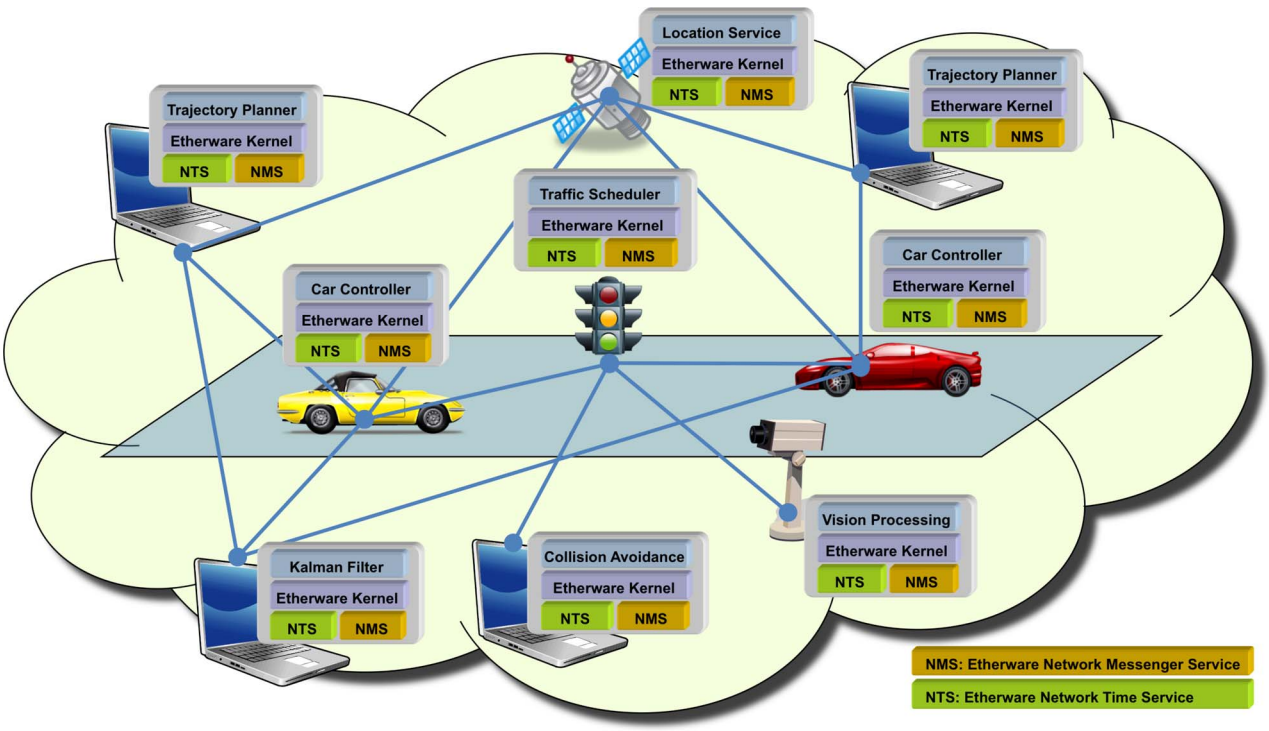

Fig. 3. *An illustrative example of Etherware-based distributed traffic control system.*

A component-based middleware framework for networked embedded systems has been developed under the European Reconfigurable, Ubiquitous, Networked Embedded Systems (RUNES) project. As in Etherware, it is designed to support runtime reconfiguration provided by a middleware service, called the Logical Mobility Service. A number of components for network reconfiguration, localization, and collision avoidance have been developed based on the RUNES middleware component framework [169]. OSA+ [170] is another middleware based on the microkernel architecture for distributed real-time embedded systems. Other real-time middleware frameworks are RTZen [171], an implementation of Real-Time CORBA developed on a real-time Java platform, and ARMADA [172], a set of communication and middleware services for real-time embedded distributed applications.

Another approach to implementation of real-time CPSs is the development of programming languages. Giotto [173] provides platform-independent high-level abstractions that can be used for specifying time-triggered sensor readings, task invocations, actuator updates, and mode switches of control systems. Platform-specific issues such as schedulability analysis of a program on a specific platform are handled by the Giotto compiler. Giotto thereby decouples high-level real-time programming of real-time embedded systems from low-level real-time scheduling of computation and communication. Other programming languages for real-time systems that have been used successfully, especially in industry, are the synchronous languages such as Esterel [174] and Signal [175].

A discussion on the importance of time in computing abstractions of every layer of the computing system and possible approaches for the development of repeatable and predictable CPS can be found in [176].

C. Real-Time Wireless Networking

CPSs rely on an underlying communication network to transport data packets between sensors, computational units, and actuators. For actions to be taken on time, these packets need to be delivered within a time deadline. Nodes may require a certain minimum throughput of such packets. Thus, CPSs need a *real-time* communication network that can provide guarantees on both the throughputs and delays of flows. The current Internet does not provide such QoS guarantees, a significant challenge.

As motivation, current automobiles have about 75 sensors and 100 switches connected by a wired network. The wiring harness is heavy, complex, hard to assemble, expensive, and subject to failures. There is significant motivation for replacing the wiring harness by a wireless access point. This can potentially lead to savings in fuel economy and reduced manufacturing cost, as well as making it possible to perform software upgrades, and add or remove devices. Packets will then have to be delivered within timing constraints.

Such a system can be modeled as an access point serving n several clients [177]. Similar to Section IV-A [6], suppose that packets arrive, one for each client, at the beginning of each common period of T slots. Suppose that each packet takes one slot to transmit, and in each slot the access point can attempt a transmission for one of the clients. The deadline for each packet is the end of the period. The throughput of each client is the long-term average of the number of packets delivered per period. Each client c has a throughput requirement of q_c packets/slot, called the *timely* throughput since it only considers packets delivered by their deadline. The distinction from the deterministic model for real-time computation is that the wireless channels are unreliable. When the access point transmits a packet for client c, it only succeeds with probability p_c. (This model of channel reliability can be generalized [178].)

There are two fundamental questions concerning the QoS requirements $\{(q_c, p_c, \tau) : c \in \{1, 2, \ldots, C\}\}$: 1) Are they feasible; and, if so, 2) what is an appropriate scheduling policy? The first item is the problem of *admission control*.

Let γ_c be a geometrically distributed random variable with mean $1/p_c$ representing the number of transmissions needed to successfully deliver client c's packet, and let $I(S) := E[[\tau - \sum_{c=1}^{n} \gamma_c]^+]$ be the expected *unavoidable* idle time when the access point has to serve only the subset $S \subseteq \{1, 2, \ldots, C\}$, of clients. With $z^+ := \max\{z, 0\}$, the necessary and sufficient condition for feasibility [177] is

$$\sum_{c \in S} \frac{q_c}{p_c} + I(S) \leq \tau, \qquad \text{for every } S \subseteq \{1, 2, \ldots, C\}. \quad (6)$$

The following weighted delivery debt policy fulfills any set of feasible clients: give priority to clients according to the expected number of packets that ought to have been delivered minus the number of packets that have been actually delivered, weighted by some positive constant.

In some situations, task frequencies can be optimally tuned to support control systems [179]. Suppose that the throughputs $\{q_c\}$ are not prespecified, but there is a strictly concave and increasing utility function $U_c(q_c)$ for each client, and the goal is to maximize the sum of the utilities: $\max_{(q_1, q_2, \ldots, q_n)} \sum_{c=1}^{n} U_c(q_c)$. This problem is difficult because the number of constraints (6) is exponential in n. One can decompose the problem into two subproblems, as in [180], by decoupling clients from the access point by using a price per unit throughput ψ_c for each client, and the amount ρ_c paid by client c. Then, client c's problem is $\max_{\rho_c}[U_c(\rho_c/\psi_c) - p_c]$, subject to $0 \leq \rho_c \leq \psi_c$. The access point's optimization problem is to determine how much timely throughput to provide to each client, given that the client is willing to pay ρ_c: $\max_{(q_1, q_2, \ldots, q_c)} \sum_{c=1}^{n} \rho_c \log q_c$, subject to the constraints (6). Surprisingly, the access point's problem is solved [181] by simply giving higher priority to

clients with lower value of the ratio. (Number of slots provided to c so far)/ρ_c. Also interesting is the consequence that neither the access point nor the clients need to know the channel reliabilities $(p_1, p_2, \ldots, p_n)$.

This formulation of the problem of real-time wireless communication can be extended to handle random arrivals [182], model fading and rate adaptation [178], provide a minimum specified throughput to each client while maximizing the total utility even when the clients are strategic and noncooperative in revealing their true utilities [183]. There has also been work on simultaneous existence of flows with delay constraints as well as flows without delay constraints [184]. A major open problem is that of delay constraints in multihop networks.

For sensor networks, protocols have been developed to support real-time applications [185]. A protocol to support timeliness is presented in [186] that exploits cellular structure for the network architecture, the periodic nature of communication, and uses EDF to support real-time messages. The SPEED protocol [187] attempts to ensure that end-to-end delay is proportional to the distance traveled by the flow. RAP [188] is a communication architecture for supporting high-level query and event services. Nano Resource Kernel (Nano-RK) is a real-time operating system for sensor networks [189].

V. WIRELESS SENSOR NETWORKS

Wireless networks allow nodes to communicate with each other over the wireless medium, possibly by using other nodes as relays or cooperating in other more information theoretic ways. By attaching sensors to nodes and providing them with computational capability, one obtains wireless *sensor* networks. They can be deployed to monitor their environment, e.g., monitoring facilities for anomalies or monitoring wildlife [190]–[192], or to conduct physics-in-the-large by offering scientists the means to deploy large number of sensors in the field and wirelessly gather information from them, as at the Center for Embedded Networked Sensing [193]. By using active sensors they can estimate distances between nodes and thus their relative positions [194].

A. Connectivity of Sensor Networks

Two nodes not in range of each other may need to communicate over several hops. Therefore, a multihop wireless network will need to ensure such a path between any two nodes, i.e., it is connected. The range of a wireless transmission will depend on its power, for the same data rate. If nodes do not employ adequate power, there may not be enough links in the network to produce a connected graph, while using too much power is wasteful. It is of interest to determine what is an appropriate range that ensures that a wireless network is connected.

A simple model is when n nodes are randomly scattered, say uniformly in a square, and employ a common range r_n that depends on n. When the nodes are few and therefore sparse, they need to choose a large range to form a connected graph, while if there are many nodes, and hence dense, then they can each choose a small range. The network is connected with a probability approaching one as the number of nodes $n \to \infty$, if and only if $r_n = \sqrt{(\log n + \gamma_n)/\pi n}$, where $\lim_{n\to+\infty} \gamma_n = +\infty$ [195], [196]. When nodes are more regularly spread, one can reduce the range while still being connected.

A related problem is that of coverage by sensors. If each node has a sensor that can detect events within a distance r_n, one is interested in how large r_n has to be so that every point in the entire domain is covered by some sensor [197].

B. Energy-Efficient Networking

The overall sensor network may often be deployed untended over a long duration, with the nodes drawing energy from their batteries or from renewable energy sources, such as solar cells, and one is interested in ensuring that the networks can survive a long duration in the field before requiring attention, e.g., replacing batteries or other maintenance [198]. All protocols used will therefore have to be energy efficient.

Clearly, any collision of packets leads to packet loss and is wasteful. Nodes will need to coordinate their wireless transmissions to avoid interfering with each other. This needs a medium access protocol. It must efficiently use the transmission medium and avoid wasting the communication spectrum that is a common resource of the network, and also be energy efficient. In contrast to wireless local area networks, ensuring fairness to all nodes is not important since sensor networks are often deployed for a specific purpose. Thus, one can design a medium access control protocol specifically for sensor networks. Also, a node wastes energy if its radio is "on" listening to packets that are not intended for it, or just "on" when there is no nearby ongoing transmission. One of the most significant ways to save power is to turn off a node's radio and put it to sleep. The protocol Sensor-MAC (S-MAC) [199] takes advantages of such sleep to save power. Sleeping can be initiated on the basis of time, i.e., by scheduled sleep, or by implicit signaling that occurs due to a neighbor's transmission. The former requires clock synchronization. Collision can be avoided by using control packets, e.g., "request-to-sent" (RTS) and "clear-to-send" (CTS) as in IEEE 802.11 [13]. Long packets can be fragmented into smaller packets, so that not all is lost when a long packet is corrupted. However, transmission of the short packets can be done in a single burst, after only a single RTS and CTS, thus amortizing their overhead. Through such strategies, MAC protocols can be made to be specifically energy efficient for sensor network deployment [200]. The protocol B-MAC is motivated by the goal of simple implementation, and aims at only providing link-layer functionality, relegating other functionalities like task synchronization and organization to higher layers, which

can then employ the mechanisms exposed by B-MAC so as to adapt to changing network or channel conditions. It employs carrier sensing and adaptive preamble sampling to design an efficient MAC for sensor network monitoring applications.

C. Routing in Sensor Networks

A routing protocol is needed for two nodes that are not neighbors to communicate. Peer-to-peer routing, multicast and all-cast may be needed by sensor network applications. Very commonly, the data gathered, or the information that is extracted, may need to be communicated to a designated "sink" or "collector" or "fusion" node, which may also possibly serve as a gateway for exfiltrating the desired information out of the sensor network. This is called "ConvergeCast" [201]. It may need to be done in an energy-efficient manner, or with low delay, depending on the application. In some applications, the identity of a node may not be important; only its data may be relevant. This can simplify ID or address management schemes. Nodes may be limited in their processing or storage capabilities. The data collected from nearby nodes may have considerable redundancy, which can also be exploited in designing an efficient protocol. The protocol may be query based, responding to particular information that is sought, or the dissemination may be content based. The protocol itself may be flat, hierarchical, or even location based [202].

D. Protocols and Operating Systems for Sensor Networks

TinyOS [203], [204], an open source operating system developed for sensor networks, has triggered much experimental and deployment activities in sensor networks. The challenges in the networking, operating system, and middleware layers are surveyed in [205]. The IEEE 802.15.4 standard [206] specifies the physical layer and medium-access control for wireless personal area networks. The Zigbee alliance [207] builds upon IEEE 802.15.4 to specify high level protocols for low data rate and low energy consumption applications. WirelessHART [208], [209] is an open communication standard for process control. So is ISA100.11a that has been developed by the International Society of Automation (ISA) [210]. An Internet Engineering Task Force Working Group has developed 6LoWPAN [211]–[213] to use Internet Protocol version 6 over IEEE 802.15.4. It allows interoperability with Internet Protocol (IP) links, while still being energy efficient, reliable, adaptable to applications, and allowing management of a large number of nodes. Interoperability with IP also allows use of established security mechanisms, network management tools, transport protocols, and services for naming, addressing, discovery, etc.

E. Clock Synchronization in Sensor Networks

In many applications, it is important that sensor measurements be time stamped. In fact, this is an important aspect of CPSs because the physical world's evolution does depend on time. Different nodes in the network may however have different clocks, and so it is necessary to synchronize them. Another reason is that in order to save energy, it is important that nodes go to "sleep" most of time, and "wake up" only when necessary to hear or send a transmission, or take a sensor measurement. When a node wakes up and transmits, it is necessary that the receiving node also be in an awake state. The more accurately their sleep–wake times are coordinated, the less is the energy wastage in an awake but idle state.

When clocks are linear, they can be described by their skew (rate) and offset. Two neighboring nodes can exchange time-stamped packets. If there is a constant but unknown time delay in such packet exchanges that is symmetric, i.e., the same for transmissions in both directions, then the nodes can determine all three quantities—offset, skew, and time delay [214], [215].

In a network, one can multiply multiple skews over successive links in a path to determine the skew between two remote nodes, and likewise one can also estimate offsets. The Flooding Time Synchronization Protocol [216] time stamps packets at the MAC layer and uses linear regression to smooth noisy time stamps and delays. When the synchronization error at each link is independent with a certain standard deviation, then summed over the links along the path, the error grows as $O(\sqrt{d})$, where d is the diameter of the network. In a grid topology where n nodes are located at, say, points with integer x and y coordinates in a square, the synchronization error grows like $O(n^{1/4})$. If nodes are uniformly and randomly located in a square of side 1, then the critical range at which the network gets connected is $O(\sqrt{(\log n)/n})$, as noted earlier. Then, the synchronization error grows like $O((n/\log n)^{1/4})$ [217]. All these errors grow polynomially with the number of nodes in the network. However, one can do much better by combining estimates over different paths [218], [219]. The error is then related to the resistance distance of the graph, i.e., the resistance between two nodes when each link is replaced by a 1-Ω resistance [218]. The resulting error in a critically connected random wireless network is then only $O(1)$, showing that error can indeed be kept bounded even in random wireless networks with large number of nodes [217].

F. In-Network Information Processing in Sensor Networks

The *raison d'etre* for sensor networks is that they can provide *information* about the environment, which may be exfiltrated through a designated gateway to an external entity. To do this, the *data* gathered by the sensor nodes has to be processed to determine relevant information. One strategy is to send all data from all nodes to the sink or gateway node, where it is centrally processed. However, this may be very wasteful of energy and communication bandwidth due to the large amount of data. An alternative

is for all nodes to conduct processing, and only send along to other nodes what is relevant. This strategy is feasible because individual nodes in sensor networks have computation capabilities. Nodes can thereby trade off computation for communication. This strategy is called "in-network computation," and how it is to be best done is an important issue.

An early precursor is the communication complexity problem in distributed computing [220]. The goal is to exchange the minimum number of bits between two nodes which each possess the value of one variable, so that they can determine the value of a function of the two variables. Similar to block communication, one can compute several instances of the function, giving rise to the direct sum problem [221]. In information theory, a similar problem is source coding with side information. One variant is to require zero error for finite block lengths [222], [223].

The problem of computing a function corresponds to a rate-distortion problem with a particular choice of distortion measure, and the required capacity is the conditional graph entropy [224]. The problem of computing some symmetric functions, i.e., invariant to permutations of their arguments, has been considered in the context of a wireless sensor network in [225]. For a random wireless network with n randomly located nodes, the shared aspect of the wireless medium is modeled by each wireless transmission consuming a certain interference footprint. The rate at which the *Average* of nodal values can be computed is $\Theta(1/\log n)$, when each node chooses a communication range that leads to a connected graph. Interestingly, this problem does not benefit, up to order, from allowing block computation. In contrast, computing the *Maximum* does significantly benefit from allowing block computation; the computational rate is $\Theta(1/(\log\log n))$. Such symmetric functions are of interest because many statistical functions are symmetric, and because they embody the data-centric paradigm where only nodal values are relevant, and not nodal identities. The problem of computing divisible functions is addressed in [225], while the problem of computing divisible functions that are amenable to divide and conquer is considered in [225]–[227].

When data are random, one can consider *optimal* function computation to minimize the expected number of bits communicated. This has been considered for symmetric functions that are Boolean valued [228], [229], and some specific problems are solved optimally or near-optimally when nodes are collocated within one hop of each other.

One can also consider the problem of in-network computation from an information theoretic point of view; this has been done for two nodes in [230] and for collocated nodes in [231]. The problem of computing in noisy networks is considered in [232]–[239]. Related information theoretic problems are studied in [240].

G. Self-Calibration in Sensor Networks

There are also interesting issues at the sensing end [241]. For example, sensing nodes may provide erroneous measurements about the environment, and it is important to detect that based on correlated sensor measurements from neighboring nodes. More generally, there is the problem of how nodes in a network can self-calibrate themselves [242].

VI. SECURITY OF CYBER–PHYSICAL SYSTEMS

Security is a critical aspect of any safety-critical system, i.e., one where physical harm can be caused. Much remains to be done for security of CPSs. The case of an attack on a Supervisory Control and Data Acquisition system is described in [243]. There have been attacks on natural gas pipeline systems [244], trams [245], power utilities [246], and water systems [247]. Recently, there has been the Stuxnet worm that attacked control systems [248]–[250]. There has been much work on security at the computational and communication layers, but CPSs have additional challenges since they involve not only the communication and computation layers, but also the control layer and the physical system itself. At the same time, one can also exploit the features of the CPS system to develop approaches to security.

Several new challenges and a research roadmap are presented in [251]. Due to the feedback processes between the physical and cyber parts, there are new communication "channels" that need to be secured. Some large-scale systems, e.g., power grid, are federated. The systems are real time, yet can be geographically distributed. There are a multiplicity of time-scales and the overall system is a system-of-systems.

The vulnerability of CPSs is increased because controllers are computers prone to bugs and attacks, the communication networks are open and of potentially large scale, increasing use of commodity solutions so that systems are susceptible to the flaws of components, protocols for control are becoming more open and accessible, and increasing functionality provided by CPS opens new vulnerabilities [252]. There are challenges and security mechanisms for prevention, detection and recovery, resilience, and deterrence of attacks [253]. Computer attacks can be detected by incorporating knowledge of the physical system under control [254]. Other results for detecting attacks can be found in [255] and [256].

Standardization efforts underway include North American Electric Reliability Corporation [257], National Institute of Standards and Technology [258], and ISA-SP99 [259].

VII. DESIGN AND DEVELOPMENT

The importance of the computing system, especially software, in control system applications has been increasing.

Since its first introduction in automobiles around 30 years ago, the amount of software has increased. Computing systems including software can take up almost half of the production costs of today's automobiles [260]. The same is true for many other control systems such as airplanes and factory automation systems. This trend is anticipated to continue due to the significant benefits provided by software technologies in control applications, with respect to functionalities, performance, and flexibility. Simultaneously, it is becoming more challenging to develop such control systems since the overall complexity of the system also increases. In fact, the performance, reliability, and production costs of control systems are becoming more dependent on those of computing systems, especially software systems. Hence, an important research issues in software technology is how to manage complexity to make it easy to design and implement software systems for reliable CPSs.

From past experience, it has been observed that one of the most effective approaches in managing complexity, and accordingly increasing productivity in software development, is to raise the level of abstraction. In the early stages of computing, assembler technology allowed us to step up from machine code to assembler code. Later in the 1970s, compiler technology raised the level of abstraction a step further from assembly language to high level programming languages such as C and Fortran, which make it significantly easier to write and understand software programs. We now have object-oriented programming languages such as C++ and Java, which allow us to develop software at even higher levels of abstraction than procedural programming languages such as C and Fortran.

The next level of abstraction beyond today's component and object-based programming can be model-driven development (MDD), as emphasized in [261]–[264]. One of the important visions of MDD is that software developers can develop software systems through designing models in the application domain instead of writing computer programs at the implementation level, and can then transform the application domain design models into real implementation. MDD can thereby significantly improve the productivity of the software development process. Another important benefit is to improve productivity in the long term by supporting developers to build a software system that is less sensitive to changes in personnel, requirements, and implementation platforms [262].

Broadly, a model is a description of some aspect of a system for some purposes such as communication, analysis, or implementation. In principle, models relevant for software systems can be in any form depending on the purpose. For example, in the traditional software development process, the requirement and functionalities of a software system are typically described in text and picture format, resulting in documents for software developers to use. At the next stage, the system is designed based on requirements typically in the form of diagrams, e.g., class diagrams and activity diagrams of Unified Modeling Language. Finally, the design is implemented and tested by software developers in the form of computer programs. One of the issues in this process is that the models at various stages are only loosely connected and information contained in a model might not be correctly captured during the transition from one form of model to another. As an example, whenever there is some change in requirements, lower level models have to be manually updated to maintain consistency between models, and *vice versa*. Another important concern is that whether there are some errors at the design stage might not be determined until the test stage of the implemented code. Thus, it requires much cost and effort to maintain consistency between models in the traditional software development process.

To fully exploit benefits of MDD such as automatic generation of complete programs from application domain models and automatic verification of a system at design time, models, especially those at the application layer, should possess properties that enable seamless usage throughout the development process [261]. Key to MDD are that a model should be 1) an appropriate abstraction of the system, hiding irrelevant details; 2) represented so that it is easily understandable for improving productivity in design and maintenance; and 3) executable, so that it can help to predict the modeled system's properties at an early stage of development process. Building such models is itself a great challenge in MDD. Major challenges in realizing the vision of MDD are categorized into three different aspects [263]: 1) modeling language to support creating well-defined models; 2) separation of concerns to support modeling a system from multiple viewpoints; and 3) model manipulation and management, such as transformation between models, maintaining consistency between models, and model-level execution and debugging.

Model-driven architecture (MDA) [265] is a conceptual framework for software development defined by OMG, and is supported by standards for modeling and transformation between models such as UML, XML Metadata Interchange, Meta-Object Facility. In particular, to improve flexibility for better support of evolving software systems, MDA models a system in three different types: 1) computation independent model to capture system requirements; 2) platform independent model to represent a system with high-level designs that are independent of any forms of implementation technologies; and 3) platform-specific model to represent a system in terms of some specific platform implementation technologies.

Model-integrated computing (MIC) [264] is another well-known software development framework which supports the development paradigm envisioned by MDD. As in MDA, models are the main artifacts for software development and used in each stage of the development process, such as design, analysis, and test. However, while MDA adopts UML as one of its primary modeling languages, MIC emphasizes the framework for designing

modeling languages, called domain-specific modeling language (DSML) [266]. DSML tool suites developed based on such an MIC concept are the Platform-Independent Component Modeling Language for component-based software system development and the Embedded Control Systems Language for distributed embedded automotive system development [267]. Another approach to MDD is Software Factories [268], which provides a software framework that can be used to create software development environments for rapid development of applications. The Architecture Analysis & Design Language (AADL) [269] is a Society of Automotive Engineers (SAE) standard model-based language that can be used for designing and analyzing structure and runtime behavioral properties such as performance, schedulability, and reliability of complex real-time embedded systems.

VIII. APPLICATIONS AND CHALLENGES

As can be seen, the research spectrum related to CPS is indeed quite broad, ranging from theories in various areas for analysis and design, to technologies for implementation. The impact of CPS research can be significant enough to bring revolutionary changes in how to design and develop engineering systems to meet societal needs in several domains such as energy, environment, and healthcare. In this section, we attempt to anticipate benefits that CPS research can potentially provide in some representative application areas. We also outline some of the challenges that need to be overcome.

A. Energy Systems

Energy generation, transmission, and distribution for a clean and sustainable society are high-priority issues that need immediate research attention in many disciplines for the global public interest. Smart grid [270] is a next-generation infrastructure for electric power systems that can help to produce, distribute, and use electricity in a more clean, efficient, and cost-effective manner through the integration of computing, communication, and control technologies. The production and distribution of electric energy can be made more responsive and reliable through real-time distributed sensing, measurement, and analysis. Furthermore, communication and information technology can contribute to improving efficiency of overall electric energy consumption by encouraging consumers to avoid consumption at peak times through dynamic pricing mechanisms and by providing useful real-time price information to consumers. Thanks to the infrastructure and mechanisms for bidirectional exchange of information and electricity, smart grid also allows traditional electric energy consumers to become providers. Electric energy that is stored or generated at residential and industrial facilities from renewable energy sources such as wind and solar can be sold to other consumers in the neighborhood or electric power providers.

Computing, communication, and control technologies can play an important role in improving efficiency in home and office building energy consumption. Electric energy used in the buildings sector is approximately 70% of total electricity consumption in the United States [271]. Energy consumption for lighting, heating/cooling, and computing can be made more efficient through distributed sensing and intelligent management of energy consumption by dynamically reacting to circumstances such as human activities and weather conditions.

B. Transportation Systems

The development of vehicles, mass transit, and traffic systems to address sustainability, efficiency, congestion, and safety is an important research issue for the benefit of our environment, economy, and safety. Next-generation transportation systems can potentially integrate intelligent vehicles and intelligent infrastructures. Intelligent vehicles can be equipped with seamlessly integrated embedded computing systems and in-vehicle networking systems. Vehicles can exchange information through wireless communication between vehicle-to-vehicle and vehicle-to-infrastructure. Intelligent mass transit systems can be more adaptive to the needs of users. Through these capabilities, vehicles can assist drivers or even drive autonomously by monitoring and estimating traffic conditions, planning ahead their behavior, and implementing the plan through drive-by-wire functionalities such as stability control, speed control, braking, and steering. Intelligent traffic infrastructures can be operated to manage the throughput of entire traffic systems. Intelligent mass transit systems can be better adaptive to the needs of users.

C. Healthcare and Medical Systems

It is an important challenge to design and develop medical devices and systems with better efficiency, reliability, intelligence, and interoperability. Medical devices need to be highly reliable, and moreover should be operated in a patient-specific manner since patients have different physiological characteristics. Formal models of patient physiological dynamics, and the hardware and software systems of medical devices, and their interactions, can play an important role in designing and verifying safety properties of devices. The integration of wireless networking and distributed sensing and computing infrastructure for interconnectivity and interoperability with medical devices enables the development of medical systems by which patient physiological conditions can be diagnosed and treated in a more integrated and intelligent manner.

D. Research Challenges

The high level of complexity of CPSs in both structural and behavioral aspects poses many challenges for researchers in realizing the benefits envisioned in many application areas.

Fundamental theoretical frameworks that can address the dynamics of CPSs in an integrated manner need to be developed. Further development of theoretical foundations is needed to better understand and predict complex dynamical behaviors caused by tight interactions between cyber and physical domains. Significant further advancement is needed to develop theories which enable us to capture and analyze the dynamics of the communications, computation, control, and applications in a unified theoretical framework.

Much research remains to be done to address complexity and productivity issues in the design and development of CPSs. Languages to model various aspects of a system at different levels of abstraction for various application domains need a fuller development. Further advances are also required to support automatic transformation between models in different semantic domains, model-level execution and debugging capabilities, composition of models to build an application, and incorporation of verification and validation capabilities.

Software platforms with well-defined and appropriate levels of abstractions and architecture are essential for the development of reliable, scalable, and evolvable CPSs in various application domains. They should hide unnecessary complexities inherent to CPSs, such as heterogeneity and distribution, and support rapid implementation of application and runtime reconfiguration and resource management to meet functional and nonfunctional requirements of an application.

Control methodologies need to be extended to much broader contexts since next-generation CPSs will be operated in much larger scales and in open environments. Algorithms and theories for high-level decision making based on information collected from different sources at different spatial and temporal scales are necessary for system-wide reliability, efficiency, security, robustness, and autonomy of CPSs.

Much important work remains to be done. ■

Acknowledgment

The authors would like to thank M. Caccamo, M. Franceschetti, S. Mitra, and P. Tabuada for their careful reading of the paper and valuable comments.

REFERENCES

[1] *Proceedings of the IEEE*. [Online]. Available: http://www.ieee.org/publications_standards/publications/proceedings/index.html

[2] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: The next computing revolution," in *Proc. 47th Design Autom. Conf.*, 2010, pp. 731–736.

[3] D. A. Mindell, *Between Human and Machine: Feedback, Control, and Computing Before Cybernetics*. Baltimore, MD: Johns Hopkins Univ. Press, 2002.

[4] S. Bennett, "A brief history of automatic control," *IEEE Control Syst. Mag.*, vol. 16, no. 3, pp. 17–25, Jun. 1996.

[5] J. P. Eckert and J. Mauchly, *Outline of Plans for Development of Electronic Computers*, 1946. [Online]. Available: http://archive.computerhistory.org/resources/access/text/2010/08/102660910-05-01-acc.pdf

[6] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *J. ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[7] L. Sha, T. Abdelzaher, K.-E. Årzén, A. Cervin, T. Baker, A. Burns, G. Buttazzo, M. Caccamo, J. Lehoczky, and A. K. Mok, "Real time scheduling theory: A historical perspective," *Real-Time Syst.*, vol. 28, no. 2, pp. 101–155, 2004.

[8] T. A. Henzinger and S. Sastry, Eds., *Proceedings of the First International Workshop on Hybrid Systems: Computation and Control*. New York: Springer-Verlag, 1998.

[9] A history of the ARPANET: The first decade. [Online]. Available: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA115440

[10] N. Abramson, "The ALOHA system: Another alternative for computer communications," in *Proc. Joint Comput. Conf.*, 1970, pp. 281–285.

[11] J. Jubin and J. Tornow, "The DARPA packet radio network protocols," *Proc. IEEE*, vol. 75, no. 1, pp. 21–32, Jan. 1987.

[12] Air Land Sea Application Center, Army, Marine Corps, Navy, Combat Air Forces, Talk II—SINCGARS, Tech. Rep., 1996.

[13] B. Crow, I. Widjaja, L. Kim, and P. Sakai, "IEEE 802.11 wireless local area networks," *IEEE Commun. Mag.*, vol. 35, no. 9, pp. 116–126, Sep. 1997.

[14] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Mobile networking for smart dust," in *Proc. ACM/IEEE Int. Conf. Mobile Comput. Netw.*, 1999.

[15] H. S. Black, "Stabilized feed-back amplifiers," *Trans. Amer. Inst. Electr. Eng.*, vol. 53, no. 1, pp. 114–120, 1934.

[16] H. Nyquist, "Regeneration theory," *Bell Syst. Tech. J.*, vol. 11, pp. 126–147, 1932.

[17] H. W. Bode, *Network Analysis and Feedback Amplifier Design*. Bell Telephone Laboratories, New York: Van Nostrand, 1952.

[18] W. R. Evans, "Control system synthesis by root locus method," *Trans. Amer. Inst. Electr. Eng.*, vol. 69, no. 1, pp. 66–69, 1950.

[19] J. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 64, pp. 759–768, 1942.

[20] S. Bennett, "Control and the digital computer: The early years," in *Proc. Int. Fed. Autom. Control*, 2002.

[21] R. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957.

[22] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishechenko, *The Mathematical Theory of Optimal Processes*. New York: Wiley, 1963.

[23] R. E. Kalman, "Design of a self-optimizing control system," *Trans. ASME*, vol. 80, pp. 468–478, 1958.

[24] R. E. Kalman, "On the general theory of control systems," in *Proc. 1st Int. Conf. Autom. Control*, 1960, pp. 481–492.

[25] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *Trans. ASME*, vol. 83, pp. 95–108, 1961.

[26] R. E. Kalman, "Canonical structure of linear dynamical systems," *Proc. Nat. Acad. Sci. USA*, vol. 48, no. 4, pp. 596–600, 1962.

[27] R. E. Kalman, "Mathematical description of linear dynamical systems," *J. Soc. Ind. Appl. Math. Ser.*, vol. 1, no. 2, pp. 152–192, 1963.

[28] J. A. E. Bryson and Y.-C. Ho, *Applied Optimal Control: Optimization, Estimation and Control*. London, U.K.: Taylor & Francis, 1975.

[29] A. N. Michel, L. Hou, and D. Liu, *Stability of Dynamical Systems: Continuous, Discontinuous, and Discrete Systems*. Boston, MA: Birkhäuser, 2007.

[30] S. P. Bhattacharyya, A. Datta, and L. H. Keel, *Linear Control Theory: Structure, Robustness, and Optimization*. Boca Raton, FL: CRC Press, 2009.

[31] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 2001.

[32] P. R. Kumar and P. Varaiya, *Stochastic Systems: Estimation, Identification and Adaptive Control*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[33] K. J. Åström and B. Wittenmark, *Adaptive Control*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1994.

[34] G. E. Dullerud and F. Paganini, *A Course in Robust Control Theory: A Convex Approach*. New York: Springer-Verlag, 2010.

[35] H. O. Fattorini, *Infinite Dimensional Linear Control Systems: The Time Optimal and Norm Optimal Problems*. Amsterdam, The Netherlands: North Holland, 2005.

[36] D. D. Šiljak, *Decentralized Control of Complex Systems*. New York: Academic, 1991.

[37] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. New York: Springer-Verlag, 1999.

[38] M. A. Arbib, "Automata theory and control theory—A rapprochement," *Automatica*, vol. 3, pp. 161–189, 1966.

[39] CAN History. [Online]. Available: http://www.can-cia.org/index.php?id=systemdesign-can-history

[40] A. Chatha, "Fieldbus: The foundation for field control systems," *Control Eng.*, vol. 41, no. 6, pp. 77–80, 1994.

[41] K. Astrom and B. Bernhardsson, "Comparison of Riemann and Lebesgue sampling for first order stochastic systems," in *Proc. 41st IEEE Conf. Decision Control*, 2002, vol. 2, pp. 2011–2016.

[42] M. Velasco, J. Fuertes, and P. Marti, "The self triggered task model for real-time control systems," presented at the Work-in-Progress Session 24th IEEE Real-Time Syst. Symp., Cancun, Mexico, Dec. 2003.

[43] X. Wang and M. Lemmon, "Self-triggered feedback control systems with finite-gain $\mathcal{L}_2$ stability," *IEEE Trans. Autom. Control*, vol. 54, no. 3, pp. 452–467, Mar. 2009.

[44] A. Anta and P. Tabuada, "To sample or not to sample: Self-triggered control for nonlinear systems," *IEEE Trans. Autom. Control*, vol. 55, no. 9, pp. 2030–2042, Sep. 2010.

[45] G. C. Walsh and H. Ye, "Scheduling of networked control systems," *IEEE Control Syst. Mag.*, vol. 21, no. 1, pp. 57–65, Feb. 2001.

[46] G. C. Walsh, H. Ye, L. G. Bushnell, J. Nilsson, and B. Bernhardsso, "Stability analysis of networked control systems," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 3, pp. 438–446, May 2002.

[47] M. S. Branicky, S. M. Phillips, and W. Zhang, "Stability of networked control systems: Explicit analysis of delay," in *Proc. Amer. Control Conf.*, 2000, pp. 2352–2357.

[48] M. Branicky, "Stability of hybrid systems: State of the art," in *Proc. 36th IEEE Conf. Decision Control*, 1997, vol. 1, pp. 120–125.

[49] L. A. Montestruque and P. J. Antsaklis, "On the model-based control of networked systems," *Automatica*, vol. 39, no. 10, pp. 1837–1843, 2003.

[50] D. Nesic and A. Teel, "Input-output stability properties of networked control systems," *IEEE Trans. Autom. Control*, vol. 49, no. 10, pp. 1650–1667, Oct. 2004.

[51] A. Hassibi, S. P. Boyd, and J. P. How, "Control of asynchronous dynamical systems with rate constraints on events," in *Proc. 38th IEEE Conf. Decision Control*, 1999, pp. 1345–1351.

[52] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems," *IEEE Control Syst. Mag.*, vol. 20, no. 1, pp. 84–99, Feb. 2001.

[53] N. Elia, "Remote stabilization over fading channels," *Syst. Control Lett.*, vol. 54, no. 3, pp. 237–249, 2005.

[54] D. Snyder and P. Fishman, "How to track a swarm of fireflies by observing their flashes (corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-21, no. 6, pp. 692–695, Nov. 1975.

[55] O. C. Imer, S. Yüksel, and T. Başar, "Optimal control of LTI systems over unreliable communication links," *Automatica*, vol. 42, pp. 1429–1439, 2006.

[56] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry, "Foundations of control and estimation over lossy networks," *Proc. IEEE*, vol. 95, no. 1, pp. 163–187, Jan. 2007.

[57] H. S. Witsenhausen, "A counterexample in stochastic optimum control," *SIAM J. Control Optim.*, vol. 6, no. 1, pp. 131–147, 1968.

[58] H. Witsenhausen, "Separation of estimation and control for discrete time systems," *Proc. IEEE*, vol. 59, no. 11, pp. 1557–1566, Nov. 1971.

[59] C. L. Robinson and P. R. Kumar, "Optimizing controller location in networked control systems with packet drops," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 4, pp. 661–671, May 2008.

[60] V. Borkar and S. Mitter, "LQG control with communication constraints," in *Communications, Computation, Control, and Signal Processing: A Tribute to Thomas Kailath*. Norwell, MA: Kluwer, 1997.

[61] W. S. Wong and R. Brockett, "Systems with finite communication bandwidth constraints—Part II: Stabilization with limited information feedback," *IEEE Trans. Autom. Control*, vol. 44, no. 5, pp. 1049–1053, May 1999.

[62] J. Baillieul, "Feedback designs for controlling device arrays with communication channel bandwidth constraints," in *Proc. ARO Workshop Smart Structures*, 1999.

[63] R. Brockett and D. Liberzon, "Quantized feedback stabilization of linear systems," *IEEE Trans. Autom. Control*, vol. 45, no. 7, pp. 1279–1289, Jul. 2000.

[64] N. Elia and S. Mitter, "Stabilization of linear systems with limited information," *IEEE Trans. Autom. Control*, vol. 46, no. 9, pp. 1384–1400, Sep. 2001.

[65] S. Tatikonda and S. Mitter, "Control under communication constraints," *IEEE Trans. Autom. Control*, vol. 49, no. 7, pp. 1056–1068, Jul. 2004.

[66] S. Tatikonda and S. Mitter, "Control over noisy channels," *IEEE Trans. Autom. Control*, vol. 49, no. 7, pp. 1196–1201, Jul. 2004.

[67] G. N. Nair and R. J. Evans, "Stabilization with data-rate-limited feedback: Tightest attainable bounds," *Syst. Control Lett.*, vol. 41, no. 1, pp. 49–56, 2000.

[68] G. N. Nair and R. J. Evans, "Stabilizability of stochastic linear systems with finite feedback data rates," *SIAM J. Control Optim.*, vol. 43, no. 2, pp. 413–436, 2004.

[69] N. Martins, M. Dahleh, and N. Elia, "Feedback stabilization of uncertain systems in the presence of a direct link," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 438–447, Mar. 2006.

[70] P. Minero, M. Franceschetti, S. Dey, and G. Nair, "Data rate theorem for stabilization over time-varying feedback channels," *IEEE Trans. Autom. Control*, vol. 52, no. 2, pp. 243–255, Feb. 2009.

[71] K. You and L. Xie, "Minimum data rate for mean square stabilizability of linear systems with Markovian packet losses," *IEEE Trans. Autom. Control*, vol. 56, no. 4, pp. 772–785, Apr. 2011.

[72] L. Coviello, P. Minero, and M. Franceschetti, "Stabilization over Markov feedback channels: The general case," in *Proc. 50th IEEE Conf. Decision Control*, 2011, pp. 3776–3782.

[73] A. Sahai and S. Mitter, "The necessity and sufficiency of anytime capacity for stabilization of a linear system over a noisy communication link—Part I: Scalar systems," *IEEE Trans. Inf. Theory*, vol. 52, no. 8, pp. 3369–3395, Aug. 2006.

[74] L. J. Schulman, "Coding for interactive communications," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pt. 1, pp. 1745–1756, Nov. 1996.

[75] T. Simsek, R. Jain, and P. Varaiya, "Scalar estimation and control with noisy binary observations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1598–1603, Sep. 2004.

[76] R. Ostrovsky, Y. Rabani, and L. Schulman, "Error correcting codes for automatic control," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 2931–2941, Jul. 2009.

[77] R. T. Sukhavasi and B. Hassibi, "Anytime reliable codes for stabilizing plants over erasure channels," in *Proc. 50th IEEE Conf. Decision Control*, 2011, pp. 5254–5259.

[78] N. Elia, "When bode meets Shannon: Control-oriented feedback communication schemes," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1477–1488, Sep. 2004.

[79] V. Gupta, B. Hassibi, and R. M. Murray, "Optimal LQG control across packet-dropping links," *Syst. Control Lett.*, vol. 56, pp. 439–446, 2007.

[80] N. Elia and J. N. Eisenbeis, "Limitations of linear remote control over packet drop networks," in *Proc. IEEE Conf. Decision Control*, 2004, pp. 5152–5157.

[81] N. Elia and J. N. Eisenbeis, "Limitations of linear control over packet drop networks," *IEEE Trans. Autom. Control*, vol. 56, no. 4, pp. 826–841, Apr. 2011.

[82] J. Nilsson, B. Bernhardsson, and B. Wittenmark, "Stochastic analysis and control of real-time systems with random time delays," *Automatica*, vol. 34, no. 1, pp. 57–64, 1998.

[83] J. Nilsson and B. Bernhardsson, "Analysis of real-time control systems with time delays," in *Proc. 35th IEEE Conf. Decision Control*, 1996, vol. 3, pp. 3173–3178.

[84] P. Naghshtabrizi and J. P. Hespanha, "Designing an observer-based controller for a network control system," in *Proc. 44th IEEE Conf. Decision Control*, 2005, pp. 848–853.

[85] D. Yue, Q.-L. Han, and C. Peng, "State feedback controller design of networked control systems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 51, no. 11, pp. 640–644, Nov. 2004.

[86] M. Pajic, S. Sundaram, J. Le Ny, G. Pappas, and R. Mangharam, "The wireless control network: Synthesis and robustness," in *Proc. 49th IEEE Conf. Decision Control*, 2010, pp. 7576–7581.

[87] J. Sandell, N. P. Varaiya, M. Athans, and M. Safonov, "Survey of decentralized control methods for large scale systems," *IEEE Trans. Autom. Control*, vol. AC-23, no. 2, pp. 108–128, Apr. 1978.

[88] M. Rotkowitz, R. Cogill, and S. Lall, "Convexity of optimal control over networks with delays and arbitrary topology," *Int. J. Syst. Control Commun.*, vol. 2, pp. 30–54, 2010.

[89] J. Baillieul and P. Antsaklis, "Control and communication challenges in networked real-time systems," *Proc. IEEE*, vol. 95, no. 1, pp. 9–28, Jan. 2007.

[90] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proc. IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[91] D. Liberzon, *Switching in Systems and Control (Systems & Control: Foundations & Applications)*. Boston, MA: Birkhäuser, 2003.

[92] H. Lin and P. J. Antsaklis, "Stability and stabilizability of switched linear systems: A survey of recent results," *IEEE Trans. Autom. Control*, vol. 54, no. 2, pp. 308–322, Feb. 2009.

[93] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho, "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems," in *Proc. ACM Hybrid Syst., Comput. Control*, 1993, pp. 250–271.

[94] T. A. Henzinger, "The theory of hybrid automata," in *Proc. 11th Annu. IEEE Symp. Logic Comput. Sci.*, 1996, pp. 278–292.

[95] N. Lynch, R. Segala, and F. Vaandrager, "Hybrid I/O automata," *Inf. Comput.*, vol. 185, no. 1, pp. 105–157, 2003.

[96] D. K. Kaynar, N. Lynch, R. Segala, and F. Vaandrager, *The Theory of Timed I/O Automata*, San Francisco, CA: Morgan Claypool, 2005.

[97] S. Mitra, "A verification framework for hybrid systems," Ph.D. dissertation, Dept. Electr. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, 2007.

[98] A. Platzer, *Logical Analysis of Hybrid Systems: Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. New York: Springer-Verlag, 2010.

[99] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi, "HyTech: A model checker for hybrid systems," *Int. J. Softw. Tools Technol. Transfer*, vol. 1, no. 1–2, pp. 110–122, 1997.

[100] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theor. Comput. Sci.*, vol. 138, no. 1, pp. 3–34, 1995.

[101] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" in *Proc. ACM Symp. Theory Comput.*, 1995, pp. 373–382.

[102] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, pp. 183–235, 1994.

[103] D. Park, "Concurrency and automata on infinite sequences," in *Proc. 5th GI-Conf. Theor. Comput. Sci.*, 1981, vol. 104, pp. 167–183.

[104] R. Milner, *Communication and Concurrency*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[105] E. M. C. Jr., O. Grumberg, and D. A. Peled, *Model Checking*. Cambridge, MA: MIT Press, 1999.

[106] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 782–798, May 2007.

[107] G. Lafferriere, G. J. Pappas, and S. Sastry, "O-minimal hybrid systems," *Math. Control Signal Syst.*, vol. 13, no. 1, pp. 1–21, 2000.

[108] L. van den Dries, *Tame Topology and O-Minimal Structures*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[109] L. van den Dries and C. Miller, "Geometric categories and o-minimal structures," *Duke Math. J.*, vol. 84, no. 2, pp. 497–540, 1996.

[110] R. Alur, T. A. Henzinger, and P.-H. Ho, "Automatic symbolic verification of embedded systems," *IEEE Trans. Softw. Eng.*, vol. 22, no. 3, pp. 181–201, Mar. 1996.

[111] A. Chutinan and B. H. Krogh, "Computational techniques for hybrid system verification," *IEEE Trans. Autom. Control*, vol. 48, no. 1, pp. 64–75, Jan. 2003.

[112] E. Asarin, O. Bournez, T. Dang, and O. Maler, "Approximate reachability analysis of piecewise-linear dynamical systems," in *Proc. ACM Hybrid Syst., Comput. Control*, 2000, vol. 1790, pp. 21–31.

[113] D. Angeli, "A Lyapunov approach to incremental stability properties," *IEEE Trans. Autom. Control*, vol. 47, no. 3, pp. 410–421, Mar. 2002.

[114] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508–2516, 2008.

[115] P. Tabuada, "An approximate simulation approach to symbolic control," *IEEE Trans. Autom. Control*, vol. 53, no. 6, pp. 1406–1416, Jun. 2008.

[116] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear systems," *IEEE Trans. Autom. Control*, vol. 51, no. 12, pp. 1862–1877, Dec. 2006.

[117] C. Belta and L. Habets, "Controlling a class of nonlinear systems on rectangles," *IEEE Trans. Autom. Control*, vol. 51, no. 11, pp. 1749–1759, Nov. 2006.

[118] K. G. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *Int. J. Softw. Tools Technol. Transfer*, vol. 1, pp. 134–152, 1997.

[119] G. Frehse, "PHAVer: Algorithmic verification of hybrid systems past HyTech," *Int. J. Softw. Tools Technol. Transfer*, vol. 10, no. 3, pp. 263–279, 2008.

[120] G. Frehse, C. L. Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, "Spaceex: Scalable verification of hybrid systems," in *Proc. 23rd Int. Conf. Comput. Aided Verif.*, 2011.

[121] C. L. Guernic and A. Girard, "Reachability analysis of linear systems using support functions," *Nonlinear Anal., Hybrid Syst.*, vol. 4, no. 2, pp. 250–262, 2010.

[122] M. Mazo, A. Davitian, and P. Tabuada, "PESSOA: A tool for embedded controller synthesis," in *Proc. 22nd Int. Conf. Comput. Aided Verif.*, 2010, pp. 566–569.

[123] A. A. Kurzhanskiy and P. Varaiya, "Ellipsoidal techniques for reachability analysis of discrete-time linear systems," *IEEE Trans. Autom. Control*, vol. 52, no. 1, pp. 26–38, Jan. 2007.

[124] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *Proc. ACM Hybrid Syst., Comput. Control*, 2005, vol. 3414, pp. 291–305.

[125] M. Ian and C. Tomlin, "Level set methods for computation in hybrid systems," in *Proc. ACM Hybrid Syst., Comput. Control*, 2000, pp. 310–323.

[126] E. C. A. Bayen and C. Tomlin, "Guaranteed overapproximations of unsafe sets for continuous and hybrid systems: Solving the Hamilton-Jacobi equation using viability techniques," in *Proc. ACM Hybrid Syst., Comput. Control*, 2002, vol. 2289, pp. 90–104.

[127] K.-D. Kim, S. Mitra, and P. R. Kumar, "Computing bounded ϵ-reach set with finite precision computations for a class of linear hybrid automata," in *Proc. ACM Hybrid Syst., Comput. Control*, 2011, pp. 113–122.

[128] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. New York: Springer-Verlag, 2009.

[129] E. Clarke, A. Fehnker, Z. Han, B. Krogh, O. Stursberg, and M. Theobald, "Verification of hybrid systems based on counterexample guided abstraction refinement," in *Proc. 9th Int. Conf. Tools Algorithms Constr. Anal. Syst.*, 2003, pp. 192–207.

[130] R. Alur, T. Dang, and F. Ivančić, "Predicate abstraction for reachability analysis of hybrid systems," *ACM Trans. Embedded Comput. Syst.*, vol. 5, no. 1, pp. 152–199, 2006.

[131] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid systems," *Proc. IEEE*, vol. 88, no. 7, pp. 971–984, Jul. 2000.

[132] A. Tiwari, "Abstractions for hybrid systems," *Formal Methods Syst. Design*, vol. 32, no. 1, pp. 57–83, 2008.

[133] G. Baliga, "A middleware framework for networked control systems," Ph.D. dissertation, Dept. Comput. Sci., Univ. Illinois Urbana-Champaign, Urbana, IL, 2005.

[134] S. Graham, G. Baliga, and P. Kumar, "Abstractions, architecture, mechanisms, and a middleware for networked control," *IEEE Trans. Autom. Control*, vol. 54, no. 7, pp. 1490–1503, Jul. 2009.

[135] A. Giridhar and P. Kumar, "Scheduling automated traffic on a network of roads," *IEEE Trans. Veh. Technol.*, vol. 55, no. 5, pp. 1467–1474, Sep. 2006.

[136] H. Kowshik, D. Caveney, and P. Kumar, "Provable systemwide safety in intelligent intersections," *IEEE Trans. Veh. Technol.*, vol. 60, no. 3, pp. 804–818, Mar. 2011.

[137] S. Gilbert, N. Lynch, S. Mitra, and T. Nolte, "Self-stabilizing robot formations over unreliable networks," *ACM Trans. Autonom. Adapt. Syst.*, vol. 4, no. 3, pp. 1–29, 2009.

[138] T. Johnson, S. Mitra, and M. S. Karthikeyan, "Safe and stabilizing distributed cellular flows," in *Proc. 30th IEEE Int. Conf. Distrib. Comput. Syst.*, 2010, pp. 577–586.

[139] S. M. Loos, A. Platzer, and L. Nistor, "Adaptive cruise control: Hybrid, distributed, and now formally verified," in *Proc. 17th Int. Symp. Formal Methods*, 2011, pp. 42–56.

[140] A. Platzer, "Quantified differential invariants," in *Proc. ACM Hybrid Syst., Comput. Control*, 2011, pp. 63–72.

[141] J. A. Stankovic, "Misconceptions about real-time computing: A serious problem for next-generation systems," *Computer*, vol. 21, no. 10, pp. 10–19, 1988.

[142] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. New York: Springer-Verlag, 2004.

[143] E. Bini, G. C. Buttazzo, and G. Buttazzo, "Rate monotonic scheduling: The hyperbolic bound," *IEEE Trans. Comput.*, vol. 52, no. 7, pp. 933–942, Jul. 2003.

[144] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings, "Applying new scheduling theory to static priority pre-emptive scheduling," *Softw. Eng. J.*, vol. 8, no. 5, pp. 284–292, 1993.

[145] S. K. Baruah, R. R. Howell, and L. Rosier, "Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor," *Real-Time Syst.*, vol. 2, pp. 301–324, 1990.

[146] L. Sha, J. P. Lehoczky, and R. Rajkumar, "Solutions for some practical problems in prioritized preemptive scheduling," in *Proc. IEEE Real-Time Syst. Symp.*, 1986, pp. 181–191.

[147] J. Strosnider, J. Lehoczky, and L. Sha, "The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments," *IEEE Trans. Comput.*, vol. 44, no. 1, pp. 73–91, Jan. 1995.

[148] J. P. Lehoczky, L. Sha, and J. K. Strosnider, "Enhanced aperiodic responsiveness in hard real-time environments," in *Proc. IEEE Real-Time Syst. Symp.*, 1987, pp. 261–270.

[149] M. Spuri, G. Buttazzo, and S. S. S. Ann, "Scheduling aperiodic tasks in dynamic priority systems," *Real-Time Syst.*, vol. 10, pp. 179–210, 1996.

[150] L. Abeni and G. Buttazzo, "Resource reservations in dynamic real-time systems," *Real-Time Syst.*, vol. 27, no. 2, pp. 123–165, 2004.

[151] T. Abdelzaher, V. Sharma, and C. Lu, "A utilization bound for aperiodic tasks and priority driven scheduling," *IEEE Trans. Comput.*, vol. 53, no. 3, pp. 334–350, Mar. 2004.

[152] L. Sha, R. Rajkumar, and J. Lehoczky, "Priority inheritance protocols: An approach to real-time synchronization," *IEEE Trans. Comput.*, vol. 39, no. 9, pp. 1175–1185, Sep. 1990.

[153] T. P. Baker, "Stack-based scheduling for realtime processes," *Real-Time Syst.*, vol. 3, no. 1, pp. 67–99, 1991.

[154] S. Lu and P. Kumar, "Distributed scheduling based on due dates and buffer priorities," *IEEE Trans. Autom. Control*, vol. 36, no. 12, pp. 1406–1416, Dec. 1991.

[155] J. Lehoczky, "Real-time queueing theory," in *Proc. IEEE Real-Time Syst. Symp.*, 1996, pp. 186–195.

[156] B. Doytchinov, J. Lehoczky, and S. Shreve, "Real-time queues in heavy traffic with earliest-deadline-first queue discipline," *Ann. Appl. Probab.*, vol. 11, no. 2, pp. 332–378, 2001.

[157] A. Mok, X. Feng, and D. Chen, "Resource partition for real-time systems," in *Proc. IEEE Real-Time Technol. Appl. Symp.*, 2001, pp. 75–84.

[158] I. Shin and I. Lee, "Compositional real-time scheduling framework with periodic model," *ACM Trans. Embedded Comput. Syst.*, vol. 7, no. 3, pp. 30:1–30:39, 2008.

[159] R. Rajkumar, K. Juvva, A. Molano, and S. Oikawa, "Resource kernels: A resource-centric approach to real-time and multimedia systems," in *Proc. SPIE/ACM Conf. Multimedia Comput. Netw.*, 1998, pp. 150–164.

[160] T. Abdelzaher, K. Shin, and N. Bhatti, "Performance guarantees for web server end-systems: A control-theoretical approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, no. 1, pp. 80–96, Jan. 2002.

[161] R. Zhang, C. Lu, T. Abdelzaher, and J. Stankovic, "Controlware: A middleware architecture for feedback control of software performance," in *Proc. 22nd Int. Conf. Distrib. Comput. Syst.*, 2002, pp. 301–310.

[162] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, "Adaptive control of virtualized resources in utility computing environments," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 4, pp. 289–302, 2007.

[163] P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," *SIGOPS Oper. Syst. Rev.*, vol. 35, no. 5, pp. 89–102, 2001.

[164] CORBA, The Object Management Group (OMG). [Online]. Available: http://www.corba.org/

[165] D. C. Schmidt, D. L. Levine, and S. Mungee, "The design of the TAO real-time object request broker," *Comput. Commun.*, vol. 21, no. 4, pp. 294–324, 1998.

[166] L. Wills, S. Kannan, S. Sander, M. Guler, B. Heck, J. Prasad, D. Schrage, and G. Vachtsevanos, "An open platform for reconfigurable control," *IEEE Control Syst. Mag.*, vol. 21, no. 3, pp. 49–64, Jun. 2001.

[167] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA: Addison-Wesley, 1994.

[168] K.-D. Kim and P. R. Kumar, "Design and experimental verification of real-time mechanisms for middleware for networked control," in *Proc. Amer. Control Conf.*, 2010, pp. 2119–2124.

[169] K.-E. Årzén, A. Bicchi, G. Dini, and S. Hailes, "A component-based approach to the design of networked control systems," *Eur. J. Control*, vol. 13, no. 2–3, pp. 261–279, 2007.

[170] U. Brinkschulte, A. Bechina, F. Picioroaga, E. Schneider, T. Ungerer, J. Kreuzinger, and M. Pfeffer, "A microkernel middleware architecture for distributed embedded real-time systems," in *Proc. 20th Symp. Reliable Distrib. Syst.*, 2001, pp. 218–226.

[171] K. Raman, Y. Zhang, M. Panahi, J. A. Colmenares, R. Klefstad, and T. Harmon, "RTZen: Highly predictable, real-time Java middleware for distributed and embedded systems," in *Proc. Int. Middleware Conf.*, 2005, pp. 225–248.

[172] T. Abdelzaher, S. Dawson, W. C. Feng, F. Jahanian, S. Johnson, A. Mehra, T. Mitton, A. Shaikh, K. Shin, Z. Wang, and H. Zou, "ARMADA middleware and communication services," *Real-Time Syst.*, vol. 16, pp. 127–153, 1999.

[173] T. A. Henzinger, B. Horowitz, and C. M. Kirsch, "Giotto: A timetriggered language for embedded programming," *Proc. IEEE*, vol. 91, no. 1, pp. 84–99, Jan. 2003.

[174] G. Berry and G. Gonthier, "The ESTEREL synchronous programming language: Design, semantics, implementation," *Sci. Comput. Programm.*, vol. 19, no. 2, pp. 87–152, 1992.

[175] P. LeGuernic, T. Gautier, M. Le Borgne, and C. Le Maire, "Programming real-time applications with SIGNAL," *Proc. IEEE*, vol. 79, no. 9, pp. 1321–1336, Sep. 1991.

[176] E. A. Lee, "Computing needs time," *Commun. ACM*, vol. 52, no. 5, pp. 70–79, 2007.

[177] I.-H. Hou, V. Borkar, and P. R. Kumar, "A theory of QoS in wireless," in *Proc. IEEE INFOCOM*, 2009, pp. 486–494.

[178] I.-H. Hou and P. R. Kumar, "Scheduling heterogeneous real-time traffic over fading wireless channels," in *Proc. IEEE INFOCOM*, 2010, DOI: 10.1109/INFCOM.2010.5462090.

[179] D. Seto, J. Lehoczky, L. Sha, and K. Shin, "On task schedulability in real-time control systems," in *Proc. IEEE Real-Time Syst. Symp.*, 1996, pp. 13–21.

[180] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237–252, 1998.

[181] I.-H. Hou and P. R. Kumar, "Utility maximization for delay constrained QoS in wireless," in *Proc. IEEE INFOCOM*, 2010, DOI: 10.1109/INFCOM.2010.5462070.

[182] I.-H. Hou and P. R. Kumar, "Admission control and scheduling for QoS guarantees for variable-bit-rate applications on wireless channels," in *Proc. ACM MobiHoc*, 2009, pp. 175–184.

[183] I.-H. Hou and P. R. Kumar, "Utility-optimal scheduling in time varying wireless networks with delay constraints," in *Proc. ACM MobiHoc*, 2010, pp. 31–40.

[184] J. Jaramillo and R. Srikant, "Optimal scheduling for fair resource allocation in ad hoc networks with elastic and inelastic traffic," *IEEE/ACM Trans. Netw.*, vol. 19, no. 4, pp. 1125–1136, Aug. 2011.

[185] Y. Li, C. S. Chen, Y.-Q. Song, and Z. Wang, "Real-time QoS support in wireless sensor networks: A survey," in *Proc. 7th IFAC Int. Conf. Fieldbuses Netw. Ind. Embedded Syst.*, 2007.

[186] M. Caccamo, L. Zhang, L. Sha, and G. Buttazzo, "An implicit prioritized access protocol for wireless sensor networks," in *Proc. IEEE Real-Time Syst. Symp.*, 2002, pp. 39–48.

[187] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A stateless protocol for real-time communication in sensor networks," in *Proc. Int. Conf. Distrib. Comput. Syst.*, 2003, pp. 46–55.

[188] C. Lu, B. Blum, T. Abdelzaher, J. Stankovic, and T. He, "RAP: A real-time communication architecture for large-scale wireless sensor networks," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp.*, 2002, pp. 55–66.

[189] A. Eswaran, A. Rowe, and R. Rajkumar, "Nano-RK: An energy-aware resource-centric RTOS for sensor networks," in *Proc. IEEE Real-Time Syst. Symp.*, 2005, pp. 256–265.

[190] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Comput.*, vol. 10, no. 2, pp. 18–25, Mar.-Apr. 2006.

[191] Y. Li, Z. Wang, and Y. Song, "Wireless sensor network design for wildfire monitoring," in *Proc. 6th World Congr. Intell. Control Autom.*, 2006, pp. 109–113.

[192] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. 1st ACM Int. Workshop Wireless Sensor Netw. Appl.*, 2002, pp. 88–97.

[193] University of California Los Angeles, Center for Embedded Networked Sensing (CENS), Los Angeles, CA. [Online]. Available: http://research.cens.ucla.edu/

[194] W.-P. Chen, J. Hou, and L. Sha, "Dynamic clustering for acoustic target tracking in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 3, no. 3, pp. 258–271, Jul.-Aug. 2004.

[195] P. Gupta and P. R. Kumar, "Critical power for asymptotic connectivity in wireless networks *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W. H. Fleming*. Boston, MA: Birkhäuser, 1998, pp. 547–566.

[196] M. Penrose, *Random Geometric Graphs (Oxford Studies in Probability)*. Oxford, U.K.: Oxford Univ. Press, 2003.

[197] L. Booth, J. Bruck, M. Franceschetti, and R. Meester, "Covering algorithms, continuum percolation and the geometry of wireless networks," *Ann. Appl. Probab.*, vol. 13, no. 2, pp. 722–741, 2003.

[198] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh, "VigilNet: An integrated sensor network system for energy efficient surveillance," *ACM Trans. Sensor Netw.*, vol. 2, no. 1, pp. 1–38, 2006.

[199] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. 21st Annu. Joint Conf. IEEE Comput. Commun. Soc.*, 2003, vol. 3, pp. 1567–1576.

[200] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM*, 2002, pp. 1567–1576.

[201] H. Zhang, A. Arora, Y.-R. Choi, and M. G. Gouda, "Reliable bursty convergecast in wireless sensor networks," in *Proc. 6th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2005, pp. 266–276.

[202] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Commun.*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[203] P. Levis, S. Madden, J. Polastre, R. Szewczyk, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler, "TinyOS: An operating system for sensor networks," in *Ambient Intelligence*, W. Weber, J. Rabaey, and E. Aarts, Eds. New York: Springer-Verlag, 2004.

[204] D. E. Culler, "TinyOS: Operating system design for wireless sensor networks," *Sensors*, 2006. [Online]. Available: http://www.sensorsmag.com/networking-communications/tinyos-operating-system-design-wireless-sensor-networks-918

[205] J. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou, "Realtime communication and coordination in embedded sensor networks," *Proc. IEEE*, vol. 91, no. 7, pp. 1002–1022, Jul. 2003.

[206] IEEE 802.15.4. [Online]. Available: http://standards.ieee.org/getieee802/download/802.15.4d-2009.pdf

[207] Zigbee alliance. [Online]. Available: http://www.zigbee.org/

[208] WirelessHART. [Online]. Available: http://www.hartcomm.org/protocol/wihart/wireless_technology.html

[209] J. Song, S. Han, A. Mok, D. Chen, M. Lucas, and M. Nixon, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp.*, 2008, pp. 377–386.

[210] ISA100.11a. [Online]. Available: http://www.isa.org/source/ISA100.11a_Release1_Status.ppt

[211] The Wireless Embedded Internet. [Online]. Available: http://6lowpan.net/

[212] D. E. Culler and J. Hui, *6loWPAN Tutorial: IP on IEEE 802.15.4 Low-Power Wireless Networks*. [Online]. Available: http://robotics.eecs.berkeley.edu/~pister/290Q/Handouts/6LoWPAN-tutorial.pdf

[213] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, *Transmission of IPv6 Packets Over IEEE 802.15.4 Networks*. [Online]. Available: http://tools.ietf.org/pdf/rfc4944.pdf

[214] S. Graham and P. R. Kumar, "Time in general-purpose control systems: The control time protocol and an experimental evaluation," in *Proc. 43rd IEEE Conf. Decision Control*, 2004, pp. 4004–4009.

[215] O. Gurewitz, I. Cidon, and M. Sidi, "One-way delay estimation using network-wide measurements," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2710–2724, Jun. 2006.

[216] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proc. 2nd Int. Conf. Embedded Netw. Sensor Syst.*, 2004, pp. 39–49.

[217] A. Giridhar and P. Kumar, "Distributed clock synchronization over wireless networks: Algorithms and analysis," in *Proc. 45th IEEE Conf. Decision Control*, 2006, pp. 4915–4920.

[218] R. M. Karp, J. Elson, C. H. Papadimitriou, and S. Shenker, "Global synchronization in sensornets," in *Proc. 6th Latin Amer. Symp. Theor. Inf.*, 2004, pp. 609–624.

[219] A. Giridhar and P. R. Kumar, "The spatial smoothing method of clock synchronization in wireless networks," in *Theoretical Aspects of Distributed Computing in Sensor Networks*, S. Nikoletseas and J. Rolim, Eds. Berlin, Germany: Springer-Verlag, 2011, pp. 227–256.

[220] A. C.-C. Yao, "Some complexity questions related to distributive computing (preliminary report)," in *Proc. 11th Annu. ACM Symp. Theory Comput.*, 1979, pp. 209–213.

[221] M. Karchmer, R. Raz, and A. Wigderson, "Super-logarithmic depth lower bounds via direct sum in communication complexity," in *Proc. 6th Annu. Conf. Structure Complexity Theory*, 1991, pp. 299–304.

[222] H. Witsenhausen, "The zero-error side information problem and chromatic numbers," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 5, pp. 592–593, Sep. 1979.

[223] N. Alon and A. Orlitsky, "Source coding and graph entropies," *IEEE Trans. Inf. Theory*, vol. 42, no. 5, pp. 1329–1339, Sep. 1996.

[224] A. Orlitsky and J. Roche, "Coding for computing," *IEEE Trans. Inf. Theory*, vol. 47, no. 3, pp. 903–917, Mar. 2001.

[225] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 755–764, Apr. 2005.

[226] R. Appuswamy, M. Franceschetti, N. Karamchandani, and K. Zeger, "Network coding for computing: Cut-set bounds," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 1015–1030, Feb. 2011.

[227] H. Kowshik and P. R. Kumar, "Optimal computation of symmetric boolean functions in tree networks," in *Proc. IEEE Int. Symp. Inf. Theory*, 2010, pp. 1873–1877.

[228] H. Kowshik and P. Kumar, "Optimal strategies for computing symmetric boolean functions in collocated networks," in *Proc. IEEE Inf. Theory Workshop*, 2010, DOI: 10.1109/ITWKSPS.2010.5503141.

[229] H. Kowshik and P. R. Kumar, "Optimal ordering of transmissions for computing boolean threshold functions," in *Proc. IEEE Int. Symp. Inf. Theory*, 2010, pp. 1863–1867.

[230] N. Ma and P. Ishwar, "Two-terminal distributed source coding with alternating messages for function computation," in *Proc. IEEE Int. Symp. Inf. Theory*, 2008, pp. 51–55.

[231] N. Ma, P. Ishwar, and P. Gupta, "Information-theoretic bounds for multiround function computation in collocated networks," in *Proc. IEEE Int. Symp. Inf. Theory*, 2009, pp. 2306–2310.

[232] A. E. Gamal, *Reliable Communication of Highly Distributed Information*. New York: Springer-Verlag, 1987, pp. 60–62.

[233] R. Gallager, "Finding parity in a simple broadcast network," *IEEE Trans. Inf. Theory*, vol. IT-34, no. 2, pp. 176–180, Mar. 1988.

[234] E. Kushilevitz and Y. Mansour, "Computation in noisy radio networks," in *Proc. 9th Annual ACM-SIAM Symp. Discrete Algorithms*, 1998, pp. 236–243.

[235] U. Feige and J. Kilian, "Finding OR in a noisy broadcast network," *Inf. Process. Lett.*, vol. 73, pp. 69–75, 2000.

[236] L. Ying, R. Srikant, and G. Dullerud, "Distributed symmetric function computation in noisy wireless sensor networks," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4826–4833, Dec. 2007.

[237] Y. Kanoria and D. Manjunath, "On distributed computation in noisy random planar networks," in *Proc. IEEE Int. Symp. Inf. Theory*, 2007, pp. 626–630.

[238] C. Dutta, Y. Kanoria, D. Manjunath, and J. Radhakrishnan, "A tight lower bound for parity in noisy communication networks," in *Proc. 19th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2008, pp. 1056–1065.

[239] N. Karamchandani, R. Appuswamy, and M. Franceschetti, "Time and energy complexity of function computation over networks," *IEEE Trans. Inf. Theory*, vol. 57, no. 12, pp. 7671–7684, Dec. 2011.

[240] B. Nazer and M. Gastpar, "Computation over multiple-access channels," *IEEE Trans. Inf. Theory*, vol. 53, no. 10, pp. 3498–3516, Oct. 2007.

[241] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," *Wireless Sensor Netw.*, vol. 2920, pp. 307–322, 2004.

[242] K. Whitehouse and D. Culler, "Calibration as parameter estimation in sensor networks," in *Proc. 1st ACM Int. Workshop Wireless Sensor Netw. Appl.*, 2002, pp. 59–67.

[243] J. Slay and M. Miller, "Lessons learned from the Maroochy water breach," in *Critical Infrastructure Protection*, vol. 253, E. Goetz and S. Shenoi, Eds. Boston, MA: Springer-Verlag, 2007, pp. 73–82.

[244] S. E. Schechter, J. Jung, and A. W. Berger, "Fast detection of scanning worm infections," in *Proc. 7th Int. Symp. Recent Adv. Intrusion Detection*, 2004, pp. 59–81.

[245] J. Leyden, "Polish teen derails tram after hacking train network," *The Register*, 2008. [Online]. Available: http://www.theregister.co.uk/2008/01/11/tram_hack/

[246] A. Greenberg, "Hackers cut cities' power," *Forbes*, 2008. [Online]. Available: http://www.forbes.com/2008/01/18/cyber-attack-utilities-tech-intel-cx_ag_0118attack.html

[247] R. Esposito, "Hackers penetrate water system computers," *ABC News*, 2006. [Online]. Available: http://abcnews.go.com/blogs/headlines/2006/10/hackers_penetra/

[248] N. Falliere, L. O. Murchu, and E. Chien, *W32.Stuxnet Dossier*, Symantec, 2011. [Online]. Available: http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32 stuxnet_dossier.pdf

[249] R. McMillan, "Siemens: Stuxnet worm hit industrial systems," *Computerworld*, 2010. [Online]. Available: http://www.computerworld.com/s/article/9185419/Siemens_Stuxnet_worm_hit_industrial_systems

[250] S. Cherry, "How Stuxnet is rewriting the cyberterrorism playbook," *Computerworld*, 2010. [Online]. Available: http://spectrum.ieee.org/podcast/telecom/security/how-stuxnet-is-rewriting-the-cyberterrorism-playbook

[251] C. Neuman, "Challenges in security for cyber-physical systems," in *Proc. S&T Workshop Future Directions Cyber-Physical Syst. Security*, 2009.

[252] A. A. Cárdenas, S. Amin, and S. S. Sastry, "Secure control: Towards survivable cyber-physical systems," in *Proc. 28th Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2008, pp. 495–500.

[253] A. Cárdenas, S. Amin, B. Sinopoli, A. Giani, A. Perrig, and S. Sastry, "Challenges for securing cyber physical systems," in *Proc. Workshop Future Directions Cyber-Physical Syst. Security*, 2009.

[254] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry, "Attacks against process control systems: Risk assessment, detection, and response," in *Proc. 6th ACM Symp. Inf. Comput. Commun. Security*, 2011, pp. 355–366.

[255] F. Pasqualetti, F. Dörfler, and F. Bullo, "Cyber-physical attacks in power networks:

Models, fundamental limitations and monitor design," in *Proc. 50th IEEE Conf. Decision Control*, 2011, pp. 2195–2201.

[256] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure state-estimation for dynamical systems under active adversaries," in *Proc. 49th Annu. Allerton Conf. Commun. Control Comput.*, 2011.

[257] North American Electric Reliability Corporation, *Critical Infrastructure Protection*. [Online]. Available: http://www.nerc.com/

[258] K. Stouffer, J. Falco, and K. Scarfone, *Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control Systems Security*, National Institute of Standards and Technology. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-82/SP800-82-final.pdf

[259] ISA, ISA99, Industrial Automation and Control Systems Security. [Online]. Available: http://www.isa.org/isa99

[260] M. Broy, "Challenges in automotive software engineering," in *Proc. 28th Int. Conf. Softw. Eng.*, 2006, pp. 33–42.

[261] B. Selic, "The pragmatics of model-driven development," *IEEE Software*, vol. 20, no. 5, pp. 19–25, Sep.-Oct. 2003.

[262] C. Atkinson and T. Kühne, "Model-driven development: A metamodeling foundation," *IEEE Software*, vol. 20, no. 5, pp. 36–41, Sep.-Oct. 2003.

[263] R. France and B. Rumpe, "Model-driven development of complex software: A research roadmap," in *Proc. IEEE Comput. Soc. Future Softw. Eng.*, 2007, pp. 37–54.

[264] J. Sztipanovits and G. Karsai, "Model-integrated computing," *Computer*, vol. 30, no. 4, pp. 110–111, 1997.

[265] OMG, *Model-Driven Architecture*. [Online]. Available: http://www.omg.org/mda/

[266] G. Karsai, J. Sztipanovits, A. Ledeczi, and T. Bapty, "Model-integrated development of embedded software," *Proc. IEEE*, vol. 91, no. 1, pp. 145–164, Jan. 2003.

[267] K. Balasubramanian, A. Gokhale, G. Karsai, J. Sztipanovits, and S. Neema, "Developing applications using model-driven design environments," *Computer*, vol. 39, no. 2, pp. 33–40, 2006.

[268] J. Greenfield, K. Short, S. Cook, S. Kent, and J. Crupi, *Software Factories: Assembling Applications With Patterns, Models, Frameworks, and Tools*. New York: Wiley, 2004.

[269] *Architecture Analysis & Design Language (AADL)*. [Online]. Available: http://www.aadl.info/

[270] Smart Grid. [Online]. Available: http://www.smartgrid.gov/

[271] DOE Building Energy Data Book. [Online]. Available: http://buildingsdatabook.eren.doe.gov/

ABOUT THE AUTHORS

Kyoung-Dae Kim received the B.S. and the M.S. degrees in mechanical engineering from Hanyang University, Seoul, Korea in 1995 and in 1998, respectively and the M.S. degree in computer science and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, Urbana, in 2011.

Currently, he is a Postdoctoral Research Associate in the Department of Electrical and Computer Engineering, Texas A&M University, College Station. His research interests include robotics, autonomous systems, distributed and cooperative control systems, real-time embedded systems, and hybrid systems.

P. R. Kumar (Fellow, IEEE) received the B.Tech. degree in electrical engineering (electronics) from the Indian Institute of Technology (IIT), Madras, India, in 1973 and the M.S. and D.Sc. degrees in systems science and mathematics from Washington University at St. Louis, St. Louis, MO, in 1975 and 1977, respectively.

From 1977 to 1984, he was a faculty member in the Department of Mathematics, University of Maryland Baltimore County, and from 1985 to 2011, in the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory at the University of Illinois. He is currently at Texas A&M University, College Station, where he holds the College of Engineering Chair in Computer Engineering. He has worked on problems in game theory, adaptive control, stochastic systems, simulated annealing, neural networks, machine learning, queueing networks, manufacturing systems, scheduling, wafer fabrication plants, and information theory. His research is currently focused on wireless networks, sensor networks, cyber–physical systems, and the convergence of control, communication, and computation.

Dr. Kumar is a member of the National Academy of Engineering of the USA, as well as the Academy of Sciences of the Developing World. He was awarded an honorary doctorate by the Swiss Federal Institute of Technology (Eidgenossische Technische Hochschule), Zurich, Switzerland. He received the IEEE Field Award for Control Systems, the Donald P. Eckman Award of the American Automatic Control Council, and the Fred W. Ellersick Prize of the IEEE Communications Society. He is a Guest Chair Professor and Leader of the Guest Chair Professor Group on Wireless Communication and Networking at Tsinghua University, Beijing, China. He is also an Honorary Professor at IIT Hyderabad. He was awarded the Daniel C. Drucker Eminent Faculty Award from the College of Engineering at the University of Illinois, the Alumni Achievement Award from Washington University in St. Louis, and the Distinguished Alumni Award from IIT Madras.