Cyber-physical Systems

Wayne Wolf, *Georgia Tech*

Sophisticated control-computing codesign provides unprecedented performance and efficiency levels for cyber-physical systems.

You may not have heard the term "cyber-physical systems" yet, but you probably will hear it some time in the next few years. And chances are, if you're reading this column, you already work on some sort of cyber-physical system. So let's try to figure out what this term means—you can use it to impress your boss.

Developers have already begun work on cyber-physical systems, the next step forward in computing. To a first order, these systems involve control/computing co-design. What does that mean? After all, we've had computers attached to stuff for a long time. Why, you may ask, do we need a new term to describe what we've been doing for years?

THEORETICAL UNDERPINNINGS

Since the dawn of the microprocessor era in the 1970s, embedded computers have been designed into many system types, but developers have done this work mostly with spit and baling wire. We have a surprisingly small amount of theory to tell us how to design computer-based control systems. Cyber-physical systems theory attempts to correct this deficiency.

Some of this theoretical shortfall is understandable. The computers we embed in systems today are vastly more complex and sophisticated than the 8080 I studied in my first computer organization class in 1977. Many of our embedded computers are even more complex than the IBM mainframe I used in my other classes that year.

While practice often runs ahead of theory, now is the time to develop a theory of real-world computing that matches our capabilities. Cyber-physical systems can mean the traditional computer-controlled machine. But we engineers hope to construct a theory that lets us build truly large computer-controlled systems that actually work.

Although today's cars and airplanes contain dozens or hundreds of computers, we must entertain a grander vision and consider a control system that stretches across the entire country. If this idea sounds crazy, think about the national power grid, a handful of networks that supply power to billions of devices. The power grid remains woefully primitive in many respects—as one commentator put it, Thomas Edison would feel right at home with the equipment in the average power plant.

EFFICIENCY BOOST

The goal at both the device and network levels is the same: efficiency. Many embedded computer applications simply replace mechanical controllers. The electronic version might be cheaper or more reliable, but it doesn't do anything a mechanical device couldn't do. Automobile engine controllers do sophisticated things that aren't possible with mechanical distributors and carburetors. But we really need to go beyond these techniques to build next-generation systems.

On the one hand, computing people settled early on for very simple abstractions of the control problem. Real-time computing was a high-priority goal for computer systems research in the 1960s and 1970s. Developers chose as their basic abstraction for timing the periodic deadline, which mandated that all computation must be done by a given time, at which point the whole process would start over. Rate-monotonic scheduling, developed by C.L. Liu and James W. Layland in the early 1970s, exemplifies this approach.

But control systems don't really work this way. The problem with deadlines is that they make the control problem too abstract compared to the physical system. Our design

 Published by the IEEE Computer Society

problem's real goal is to create a stable physical system. As anyone who has tried to balance a pencil on the palm of one hand knows, stability varies over time. In light of this, we are better off adjusting deadlines based on controllability.

We might want to use shorter deadlines when the system is closer to going unstable and longer deadlines when it is more stable. Another approach switches control algorithms based on the plant's condition and possibly the computer's current load. My colleagues—Fumin Zhang, Klementyna Szwaykowska, and Vincent Mooney—and I wrote a paper for the Real-Time Systems Symposium in which we called for a "faster, less accurate/slower, more accurate" tradeoff when controlling physical systems (F. Zhang et al, "Task Scheduling for Control-Oriented Requirements for Cyber-Physical Systems," *Proc. Real-Time Systems Symp.* [RTSS 08], IEEE CS Press, 2008, pp. 47-56).

We have yet to thoroughly understand the nature of timing constraints on physical systems. Rolf Ernst's group at the Technical University of Braunschweig, for example, has developed new timing specifications and algorithms to check the validity of timing constraints. Their work, inspired by automotive electronic networks, performs distributed computations in real time.

CONTROL THEORY ISSUES

Looking at these systems from the control-theory side, we find gaps in our knowledge. Traditional control theory views computing as a numerical device: That the computer performs discrete arithmetic needs to be taken into account, numerical properties must be considered, and so on.

But today's computers have a host of other effects that can be hard to ignore. As anyone who has tried to determine the worst-case execution time for a piece of software can testify, figuring out execution time on a high-end CPU isn't easy. Both the pipeline and any cache on the processor introduce uncertainties that are at best difficult and sometimes intractable. I've referred elsewhere to the physics of software (W. Wolf, *Computers as Components: Principles of Embedded Computing System Design*, 2nd ed., Elsevier, 2008)—the time and energy expended by software as it executes on computers, which are necessarily physical embodiments, not just mathematical abstractions.

As thermal effects become even more important, the physical properties of computers become harder to dismiss. Modern processors include thermal sensors that determine when the CPU is becoming too hot. When these sensors fire, they trigger circuitry that slows down the clock, which is necessary to ensure that the chip doesn't melt. But it might not be obvious to the software when the CPU is in danger of overheating. We need new techniques to minimize the chance that the CPU will overheat. We also need better methods to manage the control system workload when a CPU finally does overheat.

We also need to know more about how to build large-scale cyber-physical systems in which both the plant and computer controller are physically distributed. One challenge is learning how to live with networks' limitations. While small-scale systems can use specialized real-time networks, national-scale systems will inevitably rely on the Internet for part of their operation. We need to understand what parts of a real-time, closed-loop system can be put on Internet protocol networks and what parts cannot.

CYBER-PHYSICAL ROADMAP

In the long term, cyber-physical systems should trickle down to all engineering students and even to high schools. All students will at some point in their careers use computers to build cyber-physical systems, so they should learn some basic principles. These new classes will be substantially different from today's microprocessor-based systems courses. Students should learn about signal processing and control using real-world computers. All too often, today's students use laptop computers to perform their computing, which shields them from dealing with any of the physical constraints they will face in the real world. This approach is akin to trying to learn skiing while standing comfortably in the après ski lounge.

Cyber-physical systems may well become the theory backing up a new wave of computing. These systems should be able to deliver new levels of performance and efficiency thanks to sophisticated control-computing codesign. For this to happen, we must move our understanding of computers beyond information and cyberspace. In the past, we have brought our information to computers in the predigested form of keystrokes and mouse clicks. Cyber-physical systems actively engage with the real world in real time and expend real energy. This requires a new understanding of computing as a physical act—a big change for computing.

Wayne Wolf *is the Rhesa "Ray" P. Farmer Distinguished Chair of Embedded Computing Systems and Georgia Research Alliance Eminent Scholar at the Georgia Institute of Technology. Contact him at wayne.wolf@ece.gatech.edu.*

Editor: Tom Conte, College of Computing, Georgia Institute of Technology; conte@cc.gatech.edu