Position Paper for National Transportation CPS Workshop

Aloysius K. Mok
The University of Texas at Austin
October 20, 2008

CPS systems are different from computer systems in one important aspect, namely, that they must interact with physical systems for which we may not have perfect knowledge. An immediate corollary is that there may be failure modes that we cannot anticipate, and yet the CPS must be designed to have at least fail-safe properties. Transportation CPS systems are also likely to be distributed systems. Even if individual vehicles are designed to operate autonomously, the fact that they must share the same transportation infrastructure (highways, sea lanes, air traffic control) means that they are necessarily part of a distributed system. CPS systems are also heterogeneous in that they depend on the interoperability between legacy hardware/software and new and emerging technologies that are more energy-efficiency conscious. This is because of the fact that we cannot afford to rebuild the entire infrastructure to accommodate newer technology and this will be an ongoing process for the future. The above observations suggest a number of open research issues that are by themselves grand challenge problems.

Challenge 1: Failure Semantics

Since we cannot anticipate all the failure modes of complex CPS systems, we must invent new theories and technologies that can mitigate the cost of system failures. A review of the traditional approaches for engineering robust real-time systems may shed light on how we may proceed. The traditional paradigm is to construct a universe of failure modes that can encompass all possible failures and to engineer systems that can withstand specific types of failure with acceptably high probability. The usual failure mode classification consists of the ranking: crash failures, omission failures, timing failures and arbitrary (Byzantine) failures. The higher the rank of the failure mode, the harder it is to mask a failure and the more costly is the recovery. Systems that are designed with crash failure semantics are easier to recover from failures, for example, by duplicating the hardware/software servers, whereas systems that have arbitrary failure semantics are much more difficult to make safe. The key engineering insight is to design systems such that their failures fall into one of the more acceptable failure modes and the design can be accomplished at acceptable cost.

For CPS systems, the traditional failure mode classification is not satisfactory inasmuch as the classification was devised with computer (hardware) systems in mind. For CPS systems, we need a classification system that is more specific with respect to the interaction between the physical components of the system under control and the computer and communications components that control them. For example, we might be able to take advantage of the fact that mechanical components usually fail on a time scale (in seconds) that is large compared with the reaction time of computer systems (much less than a second) to define a CPS-specific failure classification that take into account the amount of time that is available for taking remedial actions. We may for example add a time dimension to the failure mode classification by including explicitly a time-to-failure parameter. There are also other dimensions that a CPS-specific failure mode

classification scheme can exploit by focusing on the application domain specifics. For example, when a traffic light fails, we do not stop all traffic but instead we put the lights in a blinking-red mode. This signals all the cars entering the intersection to follow a pre-agreed protocol, namely, the intersection should be regarded as being regulated by all-way stop signs. The blinking red lights thus impose a specific type of behavior on all traffic. This suggests a way to generalize failure mode semantics. We can define failure semantics in terms of protocols that the failed system must follow. These protocols can be formalized by the plethora of techniques from computer science, hybrid systems and other branches of engineering. In other words, we can design failure modes by the protocols that a failed system must follow and these protocols can be application specific and have a time dimension. By doing so, we can bring to bear the arsenal of techniques in computer science and hybrid systems to understand and build robust CPS systems.

Challenge 2: Legacy Hardware and Software

Future transportation systems will necessarily operate on top of the existing technological infrastructure because the cost of building a new infrastructure from scratch is prohibitive and the legal and political implications are even more daunting. An immediate corollary of this observation is that we need a path of evolutionary improvement from the current to the future systems. For CPS systems, a corresponding problem is how to ensure that the gargantuan hardware/software infrastructure that we have already deployed to operate the current embedded systems can be evolved to enable the computer control systems of the future that must satisfy more demanding functional as well as and non-functional constraints. For example, it has been reported that the high cost of testing the avionics subsystem of the F-18 aircraft has on occasions forced the aircraft's manufacturer to modify the mechanical structure of the plane in order to avoid the high cost of revalidating the operational flight software. With the computerization of civilian aircrafts and air traffic control systems, the interoperability problem between the existing and new subsystems will get worse.

The answer to this question can again be found in a review of the traditional approach. In computer science, a time-tested way to deal with system complexity is the concept of virtualization of resources. The idea is to design systems in a hierarchical fashion so that each layer in the hierarchy delivers the services that are required to implement the service at the next layer, and the description of the services provided by a layer is abstracted by a virtual machine specification. The virtual machine specification provides the only interface to the upper layer; the implementation details of the virtual machine are invisible to the upper layer. This technique of resource virtualization is also applicable to deal with the issue of legacy hardware/software systems. We can in a sense freeze the design of legacy systems by wrapping around it an interface and we make sure to supply enough resources to implement the services as specified by the interface. This way, legacy systems can be reused as components of a new system as long as the requirements specified by the interface virtual machine are satisfied. There are, however, two important problems that will require further research to address. Specifically, the questions concern the virtualization of real-time services and the I/O services.

The virtualization of real-time services presents a uniquely difficult problem for CPS systems. In the traditional concept of resource virtualization, only the functional aspects

of the server machine are of concern. Performance issues, especially the satisfaction of real-time requirements are traditionally ignored in virtualization techniques. For CPS systems, we cannot ignore performance requirements since their satisfaction is often required by safety concerns. For example, the computer system that activates the anti-lock braking system of a car must be certified to be able to react to a command to deploy within a very short time. The virtualization of real-time services is intrinsically difficult inasmuch as the interface to the virtual machine does not provide the details about the implementation that make it possible to meet the timeliness requirements. This has not been a major problem in the past, since real-time embedded systems have been traditionally designed to run on dedicated hardware/software platforms. In such an environment, the engineer has complete knowledge and control of all the hardware and software resources and s/he can schedule the computational resources explicitly by time-multiplexing and by fixing the allocation of the resources at design time. This is typically done in current avionics systems. In the future, however, the situation will change.

Unlike current avionics systems, future systems will likely be operating in an open system environment where the interface to the external world cannot assume that the workload imposed by the environment is fixed, and that the details of the scheduling policy of the resource schedulers in subsystems are available. There has been progress in extending the resource virtualization concept to cover performance requirements, in particular, the concept of real-time virtual servers. However, the problem is complex and we need investment in addressing both the theoretical and engineering problems in this area, especially the virtualization of hierarchical and I/O servers.

Challenge 3: Wireless process control

CPS systems will likely be distributed systems. It is prudent to assume that wireless technology will be a critical component in these systems. While there has been extensive academic research in wireless systems, past work has focused on communications but not the use of wireless to actuate controller devices. Industry on the other hand has proceeded to deploy wireless technology in process control applications without the benefit of the same breadth and depth of academic research. We shall mention two examples here of problem areas that require critical attention. The first area is the compliance testing of wireless protocols for process control. Unlike data communication system where noise from the environment is not an important factor, process control applications often run in very noisy environments because of the presence of high-current industrial machineries. This noisy environment complicates compliance testing, especially when different wireless technologies must co-exist. For example, research is needed for co-existence performance evaluation in the 2.4GHz ISM Band. Many wireless technologies co-exist in the 2.4 GHz ISM radio band such as Wi-Fi, Bluetooth, ZigBee and WirelessHART. The development of effective test suites is a challenge.

Another area is communications scheduling in process control systems such as WirelessHART. Unlike data communications systems, process control applications must respect stringent timing constraints. The need to combine table-driven scheduling that is traditionally adopted for task scheduling in process control systems (e.g., Fieldbus) and a more flexible scheduling regime that is appropriate for wireless transmission scheduling poses challenging research problems.