Challenges In Representing CPS Safety

Philip Koopman
Carnegie Mellon University
ECE Dept., HH A-308
Pittsburgh, PA 15213 USA
+1 (412) 268-5225
Koopman@cmu.edu

ABSTRACT

This position paper describes the challenge of ensuring run-time safety in cyber-physical systems. The overarching problem is ensuring that computer-based systems will maintain safe operations even in the face of design-time and run-time faults. One way to address this problem is by creating an ability to perform run-time safety checks on CPS applications that can be used to record hazards, trigger emergency shutdowns (where doing so is safe), or perform other actions to minimize the consequences of an unsafe system behavior. Existing foundations for creating such a capability exist in the areas of software safety, temporal logic, model based diagnosis, and fault tolerance.

1. INTRODUCTION

A Cyber Physical System (CPS) interfaces computing capability to sensors and actuators that monitor and have effect on the physical world. As such, most CPS applications have some element of safety in terms of a need to avoid harm to the system, users, or the environment resulting from the uncontrolled release of energy from actuators. How much harm an unsafe CPS can cause is a matter of degree. But any level of potential harm means that safety has to be considered at an appropriate level of criticality for the system's design.

2. CHALLENGE: RUN-TIME SAFETY

What are the challenges and possible solutions?

A CPS, like any computer-based system, is vulnerable to design defects and run-time defects. These defects can be in hardware, software, and even at the requirements level. While many useful techniques exist for improving system correctness at design time, no system operating in the field can be expected to be perfect. Even in the unlikely case that a "perfect" specification and implementation are produced, there are still safety problems that can arise from unanticipated operating conditions, maintenance errors, run-time faults, owner neglect, malicious attacks, and other sources. In other words, no system can be expected to be consistently perfect in operation. Therefore, it can be advantageous to have a way to mitigate the effects of unforeseen (and unforeseeable) defects. One way to do this is to have some sort of run-time safety monitoring and recovery approach (a *run-time safety monitor*) to help ensure system safety even in the presence of faults.

The need for run-time safety approaches is especially important for automotive applications. Given the competitive environment of time to market pressures and cost pressures it is unreasonable to expect an automotive product to be "perfect" at anything approaching an affordable cost.

Obviously no such run-time safety monitor can itself be perfect. But, we believe that including such a capability can significantly improve overall CPS safety at modest cost. Ultimately, we believe approaches based on these ideas will reduce overall system costs by reducing the required integrity level of some CPS components, because the safety monitor can assume much of the burden of assuring system-level safety.

3. RESEARCH COMMUNITIES

How can we build and maintain a community of interest in this area?

Run-time safety monitoring can potentially build upon the skills of a number of communities. Creating an interdisciplinary research thrust for this topic could facilitate new advances and encourage researchers from the following areas to work together on this challenging inter-disclipinary problem.

Software and System Safety researchers can help define safety specifications in the context of runtime system behaviors. Doing so will require an ability to define safety in terms of observable run-time parameters without actually knowing the precise operating environment beyond what sensors can detect in a given situation. Also, as a practical matter, run-time safety monitors may need to implement safety thresholds that are slightly short of an actual unsafe condition to permit time for the system to respond before an accident occurs.

Temporal Logic and Formal Methods researchers can create formal definitions of what is and is not safe system behavior to be evaluated at run time. A key challenge will be accommodating the finite amount of run-time state history that can be saved in a low cost system with limited memory. Another challenge will be reconciling the usual temporal concept of “eventually” with a run-time system that must decide whether the system is in a safe state at a particular moment without knowing what the future will be. (In other words, a rethinking of the point of view for time in temporal logic may be required.)

Model Based Diagnosis researchers can create models of safe system behavior that can be used as run-time checks. Much current model based diagnosis is based on detecting behavioral deviations (does the system work as it was built to work?). For run-time safety checking, it may be more important to build a model of safety based on emergent behaviors rather than actual implementation (does the system work in a manner consistent with a model that describes the safe operating envelope for the system, regardless of how the system was actually built?).

Fault tolerance researchers can create a method for building inexpensive run time safety checkers so that they are isolated from the main system to avoid main system defects undermining the operation of the safety monitor (while still observing system information from the main system). Additionally, it will be important to create highly dependable safety monitors at low cost.

Requirements researchers will need to create an approach for defining system-level specifications of safety properties that are as small and succinct as possible. Especially important will be managing risks and availability issues resulting from false negatives and false positives of run-time safety monitors and grappling with the emergent nature of system safety.

4. APPLICATIONS

What are promising applications?

Applications for a run-time safety detector span the development cycle across many different automotive functions.

Within the development cycle, run-time safety monitoring can be used as a debugging monitor for system simulation, prototyping, and field tests. In these uses a safety monitor can ensure that an unsafe or potentially unsafe situation (a hazard) is noticed by testers even though overall system behaviors and the operating environment did not happen to combine to result in an incident or accident. In deployed systems, a run-time safety monitor can trigger safety responses such as system shutdown, failover to degraded operating modes (limp home modes), or shedding of defective functionality.

The most promising application in the near term is automotive active safety functionality (for example, automated braking to avoid collision). The prevalent safety case for such systems is a presumption that the feature can be turned off without compromising vehicle safety. A run-time safety monitor could monitor the behavior of such a feature and disable it if it attempted some unsafe behavior, leaving the driver still in ultimate control of the vehicle.

Over the longer term, a safety monitor could be used as a trigger for reconfiguration (for example, activating a simple backup limp home capability if a primary high-functionality, but complex, subsystem acted in a way that was unsafe). It could also, if made simple and inexpensive enough, be added as a piece of each vehicular subsystem to help ensure the fail-fast/fail-silent behavior assumed by many fault tolerant strategies.

5. INNOVATIONS

What are innovations and abstractions for future automotive cyber-physical systems?

In addition to challenges described in preceding sections, key innovations required to make this approach practical will include:

- A general, flexible, formal way to represent system safety in terms of observable run time properties.
- A way to represent and reason about the safety coverage of run-time monitoring, including understanding fundamental limits to the technique and any synergy that may be available when it is combined with design-time (or even run-time) model checking.
- A way to quantify false negative and false positive risks at run time. Or, if that it is not possible, a way to reason about the fundamental limits of such abilities
- Creation of viable safety failure response mechanisms and policies. Probably this would start with a set of patterns for safety invariants and safety violation responses.
- A methodology for teasing out safety-relevant requirements from system specifications, so that only a small subset of overall system requirements need be safety critical. Alternately, a way to define safety requirements orthogonally to system behavioral requirements.
- Achieving scalability of all the above techniques to complex systems such as an entire automobile.

6. ROADMAP

What are possible milestones for the next 5, 10, and 20 years?

It is difficult to put a strict timetable in place, but here is a potential sequence in which progress might unfold.

Near term: formally defined safety invariants expressed in temporal logic are used to trigger emergency shutdown of appropriate automotive features when they attempt to behave in an unsafe manner.

Mid term: automotive systems routinely have a safety specification at the vehicle level which is independent of the functional specification, directly supporting run-time safety invariants. (And, additionally, enabling use of design time techniques such as model checking based on similar safety specification information.) Automotive systems have layers of safety invariants based on a defined safe operating envelope that trigger warnings, soft shutdowns, and hard shutdowns at the component and system level as the vehicle approaches the boundaries of safe operation.

Long term: automotive manufacturers create per-subsystem safety specifications based on formally stated invariants. This enables subsystem vendors to ensure their components will meet system level safety requirements before the system is integrated. It further allows subsystems to self-monitor and shut down (or trigger other safety-based actions) if they become potentially unsafe, even before their behaviors have a chance to affect system safety. At the system level, vehicles automatically reconfigure based on warnings from safety monitors to gracefully degrade in the event of run-time faults.

7. CONCLUSIONS

The area of run-time safety monitoring has the potential to solve one of the thorniest of CPS software problems: how do we ensure that systems are safe even in the face of design time and run time defects? It appears that the results from a number of research communities can be combined to create an approach that will work in a reasonable research timeframe.